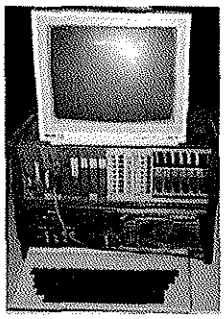
0800-666 5722

Artikel: 321085582796



Computer System MAT 85 Lehrsystem 19" 8085 Prozessor

winterzwerg78 (312 ☆) mich

Artikelzustand:	**Gebraucht**
Verkaufsdatum:	12. Mär. 2013 10:27:20 MEZ
Gebotsübersicht:	2 Gebote
Erfolgreiches Gebot:	**EUR 2,28**

Sie haben was zu verkaufen? Selbst verkaufen

Verpackung und Versand

Versand nach: Selbstabholung (kein Versand).

Abholung

Sie können sich mit dem Verkäufer in Verbindung setzen und die Einzelheiten der Bezahlung und Abholung des Artikels besprechen.

Versandversicherung

Wird nicht angeboten

Widerrufs- oder Rückgabebelehrung

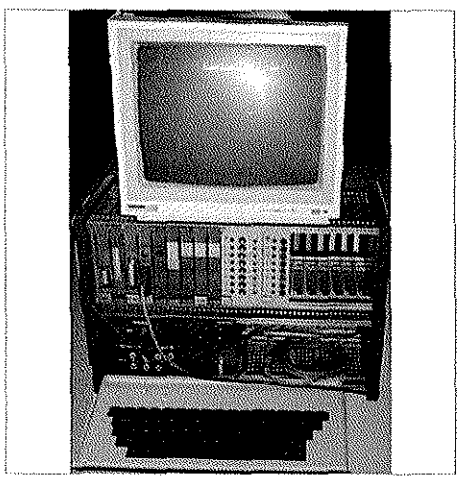
Rücknahmebedingungen: Weitere Angaben

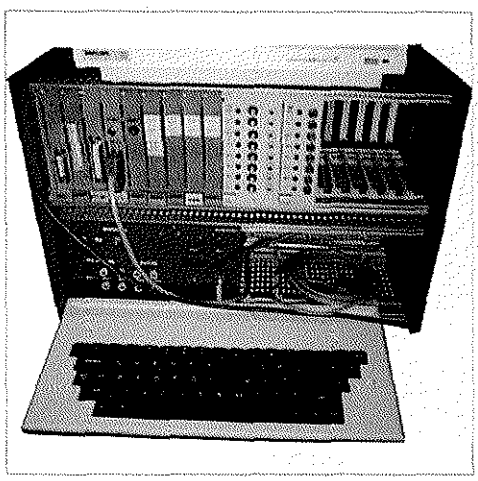
Der Verkäufer nimmt diesen Artikel nicht zurück.

Einzelheiten zur Bezahlung

Zahlungsmethode	Bevorzugt / Akzeptiert
Barzahlung bei Abholung	Akzeptiert

Angebotsbilder





Diese Seite ist speziell für Ausdrucke formatiert und enthält nicht alle Informationen des Angebots. Sie müssen alle Optionen auswählen, wenn alle Informationen im Angebot (Zusammenfassung, Artikelbeschreibung,

Hinweise zu den Übungen in diesem Band

Die in diesem Band enthaltenen Übungen sind nicht durchgehend, sondern einzeln paginiert, und zwar oben rechts auf jeder Seite. Dabei ist jeweils auch die Systemnummmer der betreffenden Übung angegeben (z.B. Ein- und Ausgabe-Einheiten BFZ/MFA 10.2.). Hierdurch ist es möglich, den Band auseinanderzunehmen und die Übungen einzeln einzusetzen.

Dieser Band enthält folgende Übungen:

Aufbau von DV-Anlagen und Bus-Systemen	BFZ/MFA 10.1.
Ein- und Ausgabe-Einheiten	BFZ/MFA 10.2.
Speicher-Einheiten	BFZ/MFA 10.3.
Mikroprozessor-Mikrocomputer	BFZ/MFA 10.4.
Steuerung einer Paketwendeanlage	BFZ/MFA 20.1.
MAT 85 (Betriebsprogramm)	BFZ/MFA 7.1.
Softwarepaket SP 1 (Betr.Prog.Erweiterung)	BFZ/MFA 7.2.
BFZ-Monitor-Listing	Version 1.8

Übersicht über die Fachpraktischen Übungen:

Der Band 1 der FPÜ enthält:

Baugruppenträger mit Busverdrahtung	BFZ/MFA 0.1.
Busabschluß	BFZ/MFA 0.2.
Trafo-Einschub	BFZ/MFA 1.1.
Spannungsregelung	BFZ/MFA 1.2.
Prozessor 8085	BFZ/MFA 2.1.
8-K-RAM/EPROM	BFZ/MFA 3.1.
8-Bit-Parallel-Ausgabe	BFZ/MFA 4.1.
8-Bit-Parallel-Eingabe	BFZ/MFA 4.2.
Bus-Signalgeber	BFZ/MFA 5.1.
Bus-Signalanzeige	BFZ/MFA 5.2.
Inbetriebnahme 8085-System	BFZ/MFA 6.1.
MAT 85 (Betriebsprogramm)	BFZ/MFA 7.1.
ASCII-Tastatur	BFZ/MFA 8.1.
Video-Interface	BFZ/MFA 8.2.

Der Band 2 der FPÜ enthält:

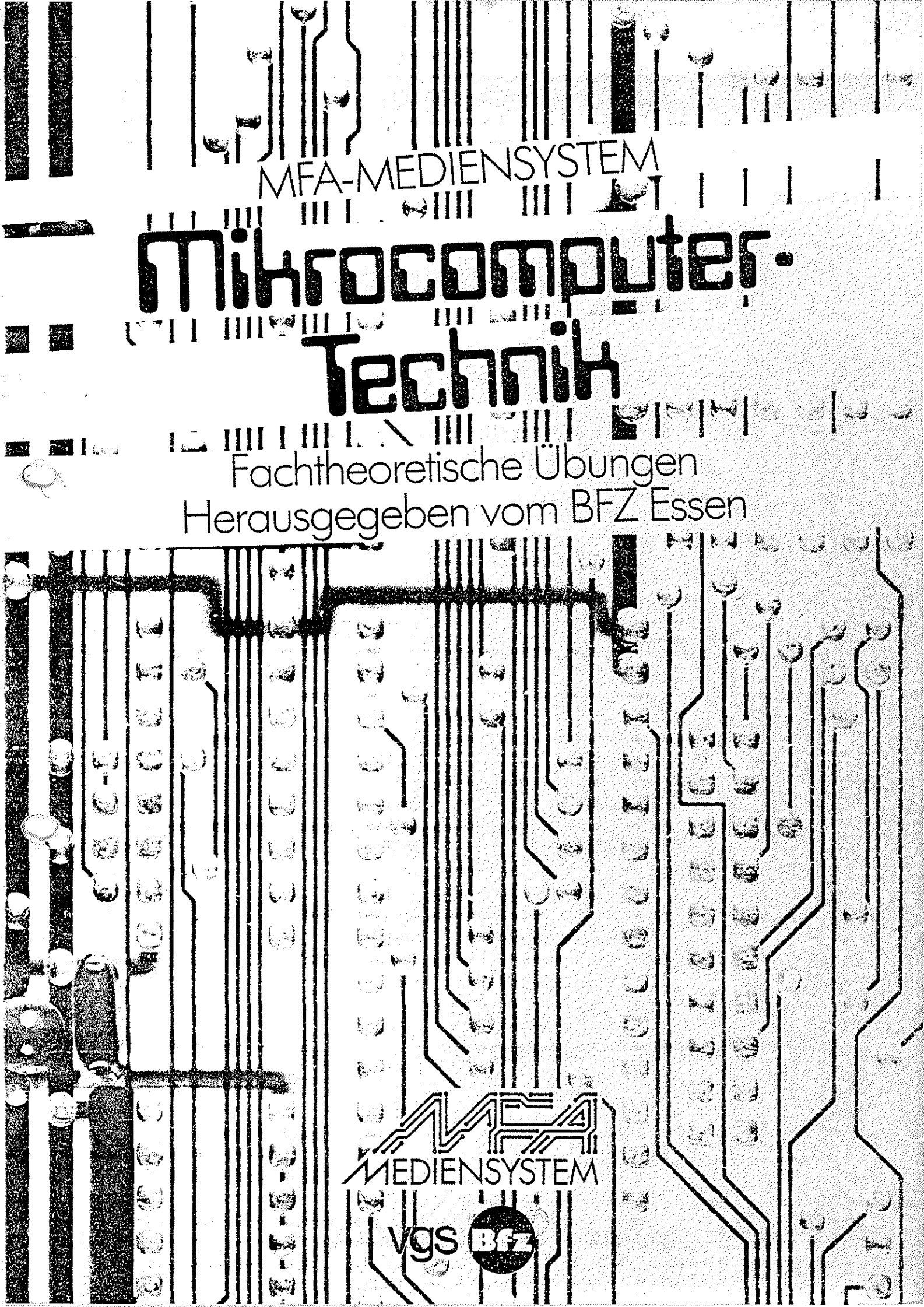
16-K-RAM/EPROM	BFZ/MFA 3.2.
Programmierbare Parallelschnittstelle	BFZ/MFA 4.3.
EPROM-Programmierer	BFZ/MFA 4.3.a
Drucker-Interface	BFZ/MFA 4.3.b
Zeitwerk (4-fach)	BFZ/MFA 4.3.c
Programmierbare Seriellschnittstelle	BFZ/MFA 4.4.
Kassetten-Interface	BFZ/MFA 4.4.a
Analoge Ein-/Ausgabe (2-kanalig)	BFZ/MFA 4.5.
Zähler und Zeitgeber	BFZ/MFA 4.6.
Adapterkarte	BFZ/MFA 5.3.
Fehlersimulation	BFZ/MFA 5.4.
Demonstrationsmodell	BFZ/MFA 5.5.

MFA-MEDIENSYSTEM

Mikrocomputer-Technik

Fachtheoretische Übungen

Herausgegeben vom BFZ Essen



MEDIENSYSTEM

vgs BFZ

MFA-Mediensystem
Mikrocomputer-Technik

Fachtheoretische Übungen

MFA-MEDIENSYSTEM

Mikrocomputer-Technik

Fachtheoretische Übungen
Herausgegeben vom BFZ Essen

MFA MEDIENSYSTEM
vgs BFZ

CIP-Kurztitelaufnahme der Deutschen Bibliothek

MFA-Mediensystem Mikrocomputer-Technik/hrsg. vom BFZ Essen. (Red./MFA-Projektgruppe: N. Meyer...).- Köln: Verlagsgesellschaft Schulfernsehen
NE: Meyer, Norbert (Red.); Berufsförderungszentrum Essen
Fachtheoretische Übungen. - 1. Aufl. - 1984.
ISBN 3-8025-1233-2

Herausgeber: Berufsförderungszentrum Essen e.V. (BFZ)
Altenessener Str. 80/84
4300 Essen 12
Tel.: 0201/3204-1

Redaktion/
MFA-Projektgruppe: F. Derriks, A. Dix, H. Gregel, C. Handel, M. Hüllweg, F. Lindemann, E. Matl, N. Meyer, K. Michaely, Fr. H. Milde, L. Refardt, Fr. G. Roßmanek, S. Sagawe, W. Schmit, F.J. Senicar, K.D. Strelow, H. Storbek, H. Schwieters, S. Wirtgen

© 1984 Berufsförderungszentrum Essen e.V.
Diese Publikation ist urheberrechtlich geschützt.
Alle Rechte sind vorbehalten.

Verlag: Verlagsgesellschaft Schulfernsehen, Köln

1. Auflage 1984 (überarbeitete Fassung)

Satz und Zeichnungen: BFZ Essen

Druck und Binden: Beltz Offsetdruck, Hemsbach

Berufsförderungszentrum Essen e.V.

Das Berufsförderungszentrum Essen e.V. (BFZ) ist eine Berufsbildungsstätte für Erwachsene in Trägerschaft der Bundesregierung (BMBW), der Landesregierung NW (MAGS), der Bundesanstalt für Arbeit, der Stadt Essen, verschiedener Arbeitgeber- und Arbeitnehmerorganisationen sowie der Kammern und Kirchen. Mit einem breit gefächerten Berufs- und Fortbildungsangebot wird hier den Anforderungen und Entwicklungen von Arbeitsmarkt und Technik Rechnung getragen.

Durch den ständigen Kontakt mit Fachleuten der Wirtschaft, der Bundesanstalt für Arbeit, des Bildungssystems und der Sozialorganisationen sowie durch wissenschaftliche Begleituntersuchungen ist sichergestellt, daß sowohl die Bildungsziele als auch die vermittelten Inhalte den Anforderungen der Arbeitsplätze entsprechen.

Seit April 1972 bis Ende 1983 haben nahezu 4000 Teilnehmer und Teilnehmerinnen an Umschulungsmaßnahmen des BFZ aus dem gesamten Bundesgebiet ihre Abschlußprüfung vor den Prüfungsausschüssen der IHK Essen bzw. der Landwirtschaftskammer Bonn mit Erfolg abgelegt. Das BFZ führt Umschulungsmaßnahmen in den folgenden Berufsbereichen durch:

- Elektrotechnik,
- Meß- und Regeltechnik,
- Metall,
- kaufmännische und datenverarbeitende Berufe,
- Gartenbau.

Daneben beinhaltet das Berufsbildungsprogramm des BFZ u.a. eine Reihe von Maßnahmen im Vorfeld der Umschulung und Fortbildung (Fernvorförderung, Grundstufe, Bildungserprobung, Informationsseminare für Arbeitslose) sowie Seminare zur Anpassungsfortbildung.

Diese Anpassungsfortbildungsseminare werden in unterschiedlichen Seminarformen sowohl für Arbeitslose (Vollzeit) als auch für Berufstätige (berufsbegleitend, externe Seminare) durchgeführt. Sie sollen Facharbeiter bzw. andere Fachkräfte mit entsprechender Berufspraxis in die Lage versetzen, den veränderten Qualifikationsanforderungen, die die Einführung neuer Technologien mit sich bringt, gerecht zu werden.

Anpassungsfortbildungsseminare werden vom BFZ in zwei Bereichen durchgeführt:

- Digital-/Mikrocomputer-Technik
- NC-Technik (CNC-Drehen/CNC-Fräsen)

Als Modelleinrichtung der beruflichen Erwachsenenbildung hat das BFZ in der Vergangenheit darüber hinaus eine Reihe von Modellprojekten durchgeführt sowie moderne Medien für den Bereich der beruflichen Bildung entwickelt. In dieser Tradition steht auch der Modellversuch "Einsatz der Mikrocomputer-Technik in der Facharbeiterausbildung (MFA)", dessen Träger das BFZ seit 1980 ist.

Vorwort

Der vorliegende Band "Fachtheoretische Übungen (FTÜ)" ist Teil des MFA-Mediensystems für die Aus- und Weiterbildung von Fachkräften auf dem Gebiete der Hard- und Software von Mikrocomputern. Es wurde im Rahmen des Modellversuchs zum

"Einsatz der Mikrocomputer-Technik
in der Facharbeiterausbildung (MFA)"

entwickelt. Dieser Modellversuch, der vom Bundesministerium für Bildung und Wissenschaft (BMBW), dem Bundesministerium für Forschung und Technologie (BMFT) und der Bundesanstalt für Arbeit (BA) finanziert wird, hat u.a. das Ziel, auf dem Gebiete der Mikrocomputer-Technik Aus- und Weiterbildungskonzepte einschließlich der erforderlichen Medien bereitzustellen. Damit soll der durch die Entwicklung des Mikroprozessors bedingten technologischen Veränderung Rechnung getragen werden.

Im ersten Teil dieses Bandes wird die Hardware, das heißt der Aufbau und die Wirkungsweise eines Mikrocomputers behandelt. Der zweite Teil vermittelt grundlegende Kenntnisse zur Software.

Hardware-Vermittlung

Der Hardware-Teil bietet die Möglichkeit, die Fachkräfte auf zwei unterschiedlichen Vermittlungsebenen in den Aufbau und die Wirkungsweise eines Mikrocomputers einzuführen. Dabei beginnt die Vermittlung bei den einfachen Funktionseinheiten und führt über den Mikroprozessor zum Mikrocomputer hin.

- Verbal-anschauliche Ebene

Die erste Ebene basiert auf der Vermittlung von Funktionsschemata, die weitgehend von der technischen Realisierung eines Mikrocomputers und den dafür erforderlichen Bauelementen losgelöst sind. Diese Ebene, verbal-anschauliche Ebene genannt, setzt für die Bearbeitung lediglich die Kenntnis des digitaltechnischen Prinzips voraus und kann auf breiter Basis vermittelt werden.

- Prinzipschaltbild-Ebene

Die zweite Ebene orientiert sich an Fachkräften, die über den allgemeinen Aufbau und die allgemeine Funktionsweise hinaus die grundlegenden Schaltungstechniken der Funktionseinheiten eines Mikrocomputers kennenlernen müssen. Sie sollen damit in die Lage versetzt werden, mittels meßtechnischer Untersuchungen die Funktionsweise eines Mikrocomputers überprüfen zu können. Diese Ebene wird Prizipschaltbild-Ebene genannt.

Software-Vermittlung

Im Software-Teil dieses Bandes wird anhand einer Problemstellung zur Steuerung einer Paketwendeanlage in die Programmierung eines Mikrocomputers eingeführt. Die programmiertechnische Lösung wird anhand unterschiedlicher Programmiersprachen aufgezeigt. Neben einem Assembler-Programm wird auch ein SPS- und ein BASIC-Programm erarbeitet. Gerade die "Speicherprogrammierbare Steuerungstechnik (SPS)" gewinnt zunehmend an Bedeutung, da sie die Möglichkeit bietet, auf einer einfachen Weise Steuerungsprobleme programmiertechnisch zu lösen.

MFA-Mikrocomputer

Die Bearbeitung der Übungsaufgaben setzt ein Lehrsystem voraus, das es ermöglicht, die einzelnen Funktionseinheiten eines Mikrocomputers wie Eingabe-, Ausgabe-, Speicher- oder Prozessor-Einheit getrennt zu betreiben. Ein solcher Mikrocomputer ist in einem weiteren Band des MFA-Mediensystems, dem Band I "Fachpraktische Übungen (FPÜ)" beschrieben. Dieser Band enthält alle Informationen über den Aufbau und die Inbetriebnahme des MFA-Mikrocomputers, einschließlich detaillierter Bauelemente- und Schaltungsbeschreibungen und stellt damit eine dritte, sehr tiefgehende Vermittlungs-Ebene (Schaltbild-Ebene) dar.

Die ergänzende Beschreibung zum Betriebsprogramm MAT 85 einschließlich der Erweiterung SP 1 (MAT 85+, SPS, Steuer-BASIC, EPROM-Programmierer) soll die Handhabung und Bedienung des Mikrocomputers bei der Software-Vermittlung erleichtern.

Speziell für die Vertiefung und die Lösung eigener Probleme in Verbindung mit dem Betriebsprogramm MAT 85, befindet sich im Anhang das komplette kommentierte Assembler-Listing des Monitors.

Norbert Meyer, Projektleiter
Franz Derriks, Entwicklungsleiter
Christian D. Handel, Stellv. Projektleiter

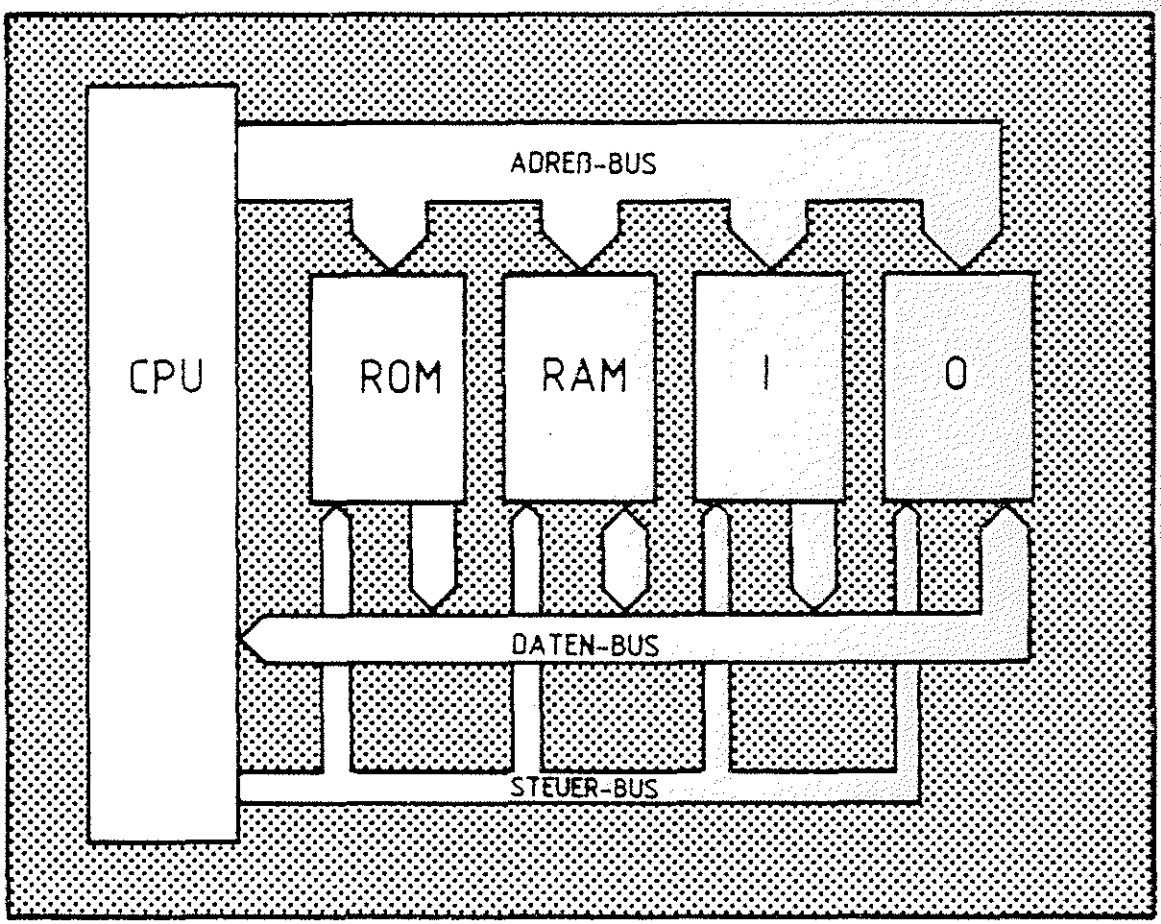
Das gesamte MFA-Mediensystem (Hardware und Begleitbücher) wird von der vgs, Breite Str. 118/120, 5000 Köln 1, vertrieben.

Im regelmäßig erscheinenden BFZ/MFA-Info werden Ergänzungen, Korrekturen, Anwendungen etc. veröffentlicht. Dieses "Info" ist kostenlos beim BFZ Essen, Postfach 12 00 11, 4300 Essen 12, zu beziehen.

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

AUFBAU VON DV-ANLAGEN UND BUS-SYSTEMEN

BFZ/MFA 10.1.



BFZ
MFA

Diese Übung ist Bestandteil eines Mediensystems, das im Rahmen eines vom Bundesminister für Bildung und Wissenschaft, vom Bundesminister für Forschung und Technologie sowie der Bundesanstalt für Arbeit geförderten Modellversuches zum Einsatz der "Mikrocomputer-Technik in der Facharbeiterausbildung" vom BFZ-Essen e.V. entwickelt wurde.

Inhaltsverzeichnis

Theorieteil 1

1.1. Einleitung
1.2. Die Komponenten eines Computers
1.3. Anwendungsbeispiele für Mikrocomputer
1.4. Prinzip der Bus-Technik
1.5. Beschreibung von Bus-Signalen
1.6. Darstellung der Bus-Verbindungen

Übungsteil 1

A1 Messen der Adreß-Signale auf dem Adreß-Bus
A2 Messen der Datensignale auf dem Daten-Bus
A3 Überprüfen der Steuersignale
A4 Überprüfen der Wirkung der Steuersignale auf der Bus-Signalanzeige

Theorieteil 2

2.1. Tristate-Technik
2.2. Kurzschlüsse auf Bus-Leitungen
2.3. Unterbrechungen auf Bus-Leitungen
2.4. Signal-Zeit-Diagramme für Bus-Systeme

Übungsteil 2

A1 Messungen am Daten-Bus bei einer unterbrochenen Datenleitung
A2 Messungen am Daten-Bus bei einem Kurzschluß zwischen zwei Datenleitungen

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

AUFBAU VON DV-ANLAGEN UND BUS-SYSTEMEN

BFZ/MFA 10.1.

THEORIETEIL 1

1.1. Einleitung

Über Mikrocomputer hört man die unterschiedlichsten Dinge, die einen bezeichnen sie als Jobkiller, andere als Jobknüller. Viele verstehen diese Entwicklung als technische Revolution. Im folgenden soll versucht werden, die dahinterstehende Technik durchschaubar und damit beurteilbar zu machen.

Man versteht unter dem englischen Begriff "Computer" nichts anderes als einen "Rechner" und unter einem Mikrocomputer (mikro = sehr klein) einen Computer, der aus einem oder wenigen kleinen Bausteinen aufgebaut ist. Möglich wurde dies durch eine fortschreitende Entwicklung des Herstellungsprozesses für "Integrierte Schaltkreise" (IC's). Sämtliche Elemente einer elektronischen Schaltung wie Widerstände, Dioden, Transistoren usw. werden dabei in ein kleines Plättchen aus Halbleitermaterial, meist Silizium, eingebracht. Heute ist es möglich, in einem Plättchen von der Größe 5 mm x 5 mm ca. 150.000 Transistoren zu integrieren. Die erreichte Packungsdichte ändert sich zur Zeit noch von Jahr zu Jahr.

Computer = Rechner

Integrierter Schaltkreis

Ein Computer ist in der Technik nichts Neues. Aufbau, Arbeitsweise und die möglichen Anwendungen sind seit langem bekannt. Jedoch verhinderten Anschaffungskosten, Baugröße, Wartungskosten usw. den Einsatz auf breiter Basis. Die Herstellung eines Computers aber auf wenigen kleinen Chips, sowie die Massenproduktion dieser kleinen Bauteile, führte schnell zu einem erheblichen Preisverfall und damit zur verstärkten Anwendung. Taschenrechner, CNC-Maschinen, Computerspiele usw. sind nur einige Beispiele dafür.

CNC = Computer Numeric Control, rechnergeführte numerische Steuerung

Was aber macht den Computer so anwendungsfreundlich und so flexibel in der Anpassung an die verschiedensten Aufgaben und Anwendungen?

1.2. Die Komponenten eines Computers

Die am Ende der Einführung gestellte Frage kann beantwortet werden, wenn Aufbau und prinzipielle Arbeitsweise eines Computers bekannt sind.

Wie bearbeitet der Mensch eine Aufgabe?

Die Arbeitsweise eines Computers und das Vorgehen eines Menschen beim Lösen einer Aufgabe sind einander sehr ähnlich. Daher soll im folgenden unser Tun z.B. beim Lösen einer mathematischen Aufgabe analysiert werden. Für die Lösung einer solchen Aufgabe müssen einige Voraussetzungen erfüllt sein. Neben einer Arbeitsanweisung (oder Formelbuch) müssen die zu verarbeitenden Zahlen, auch Daten genannt, vorliegen. Da es sich um eine mathematische Aufgabe handelt, wird noch ein Taschenrechner benötigt. Für Zwischenergebnisse und Zahlen soll ein Notizblock verwendet werden. Das Ergebnis der Aufgabe soll anschließend auf einer Schreibmaschine protokolliert werden. Im Bild 1 sind die Randbedingungen und Hilfsmittel schematisch dargestellt.

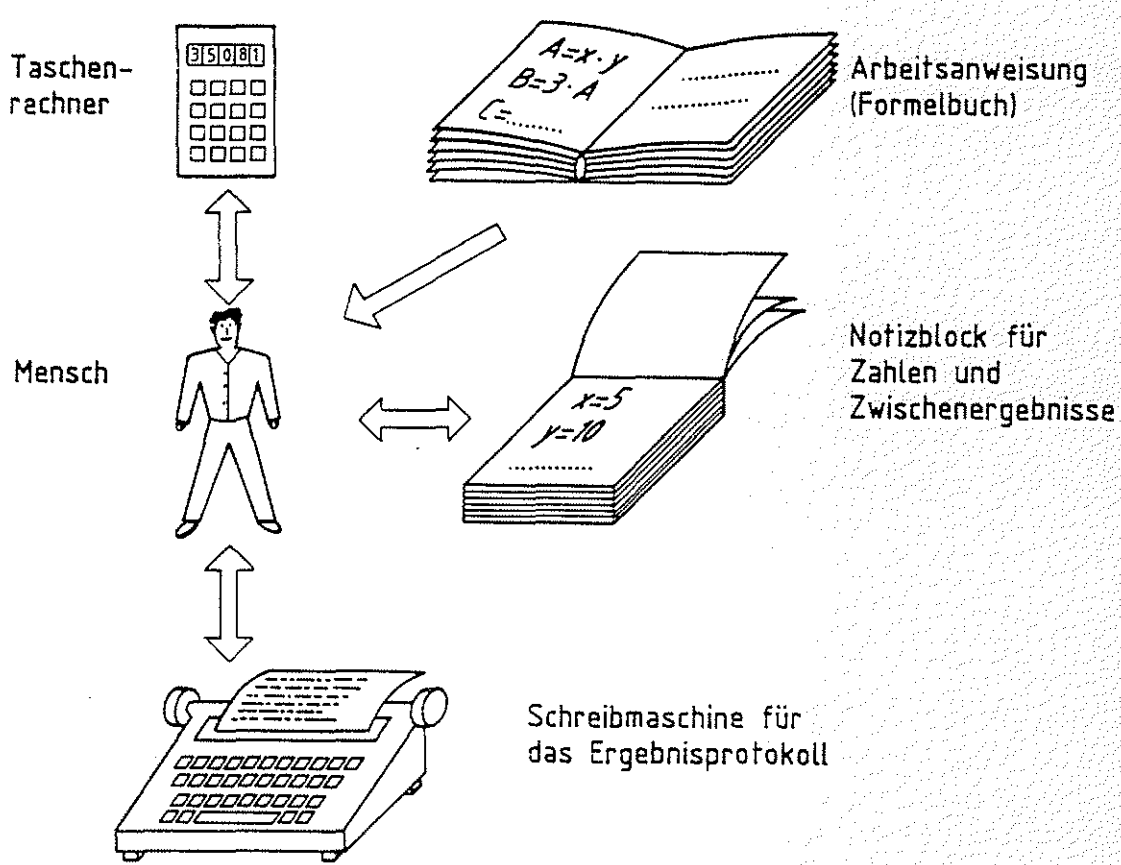


Bild 1: Hilfsmittel zur Bearbeitung einer Rechenaufgabe

Der Mensch wird bei Bearbeitung dieser Aufgabe folgende Arbeitsschritte durchführen:

- ● Erste Arbeitsanweisung lesen, deuten und im Gedächtnis speichern.
- — Notwendige Zahlen (Daten) vom Notizblock lesen und in den Taschenrechner eingeben.
- — Rechenoperation durch Tastendruck am Taschenrechner auslösen.
- — Zwischenergebnis auf dem Notizblock notieren.
- ● Nächste Arbeitsanweisung lesen, deuten und im Gedächtnis speichern.
- — Notwendige Zahlen/Zwischenergebnisse vom Notizblock lesen und in den Taschenrechner eingeben.
- ● ... Arbeitsanweisung lesen, deuten ...
- ● ... Arbeitsanweisung lesen, deuten ...
- ● Ergebnis auf der Schreibmaschine protokollieren. Fertig.

Notwendige Arbeitsschritte zur Lösung einer Aufgabe

Diese Vorgehensweise des Menschen läßt sich fast unverändert auf den Computer übertragen. Die beteiligten Komponenten (Mensch, Taschenrechner, ...) findet man auch in einem Computer wieder, jedoch mit anderen Namen.

Die Komponenten des Computers

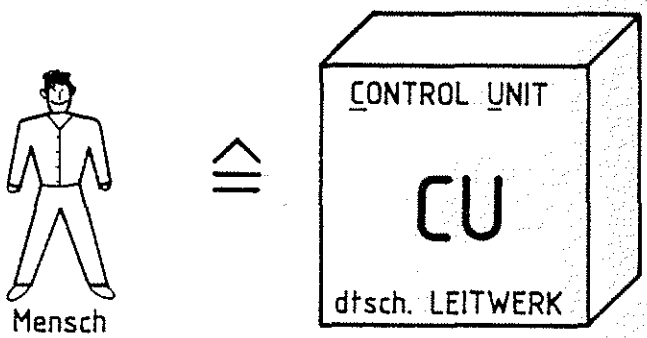


Bild 1a: Mensch und Leitwerk

Dem Menschen, der Schritt für Schritt die Arbeitsanleitung liest und den Arbeitsablauf entsprechend steuert, entspricht im Computer das Leitwerk (englisch Control Unit) (Bild 1a).

Leitwerk = Control Unit

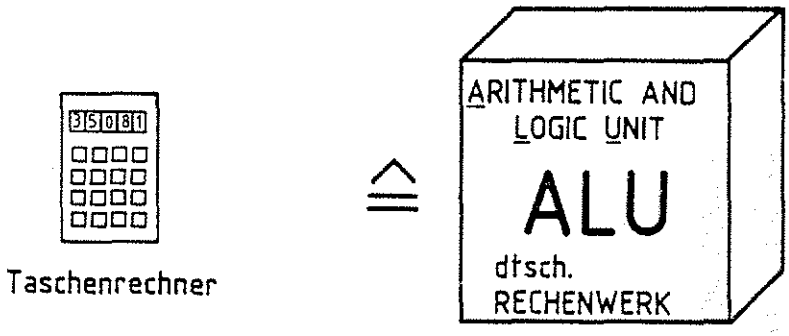


Bild 1b: Taschenrechner und ALU

Dem Taschenrechner, mit dem Rechenoperationen ausgeführt werden, entspricht in einem Computer das Rechenwerk (englisch Arithmetic und Logic Unit, kurz ALU) (Bild 1b).

Rechenwerk = Arithmetic and Logic Unit, ALU

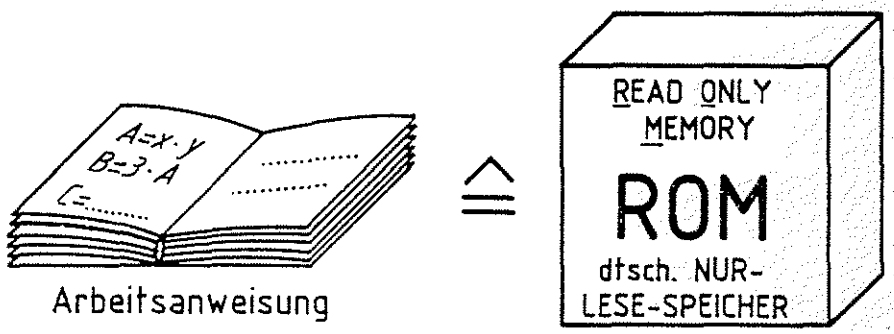


Bild 1c: Arbeitsanweisung und ROM

Das Buch, in dem die auszuführenden Arbeitsschritte stehen, nennt man im Computer Speicher (englisch Memory). Da die Arbeitsanleitung nur gelesen wird, legt man sie häufig in besonderen Speichern ab, die ebenfalls nur gelesen werden können. Diese Speicher heißen Nur-Lese-Speicher (englisch Read Only Memory, kurz ROM) (Bild 1c).

Nur-Lese-Speicher = Read Only Memory, ROM

Die einzelnen Arbeitsschritte nennt man auch Befehle oder Instruktionen (englisch instruction) und die gesamte Befehlsfolge zur Lösung der Aufgabe heißt Programm.

Befehl = Instruction
Befehlsfolge = Programm

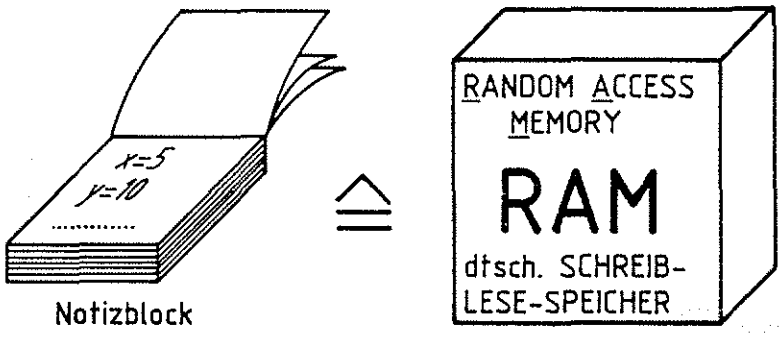


Bild 1d: Notizblock und RAM

Auf dem Notizblock können Zahlen und Zwischenergebnisse notiert werden, die nach Lösung der Aufgabe nicht mehr benötigt werden. Der Notizblock entspricht im Computer einem Speicher, in den man Daten hineingeben und, falls erforderlich, wieder herausholen kann. Ein solcher Speicher heißt Schreib-Lese-Speicher (englisch Random Access Memory = Speicher mit wahlfreiem Zugriff) oder kurz RAM (Bild 1d).

Schreib-Lese-Speicher = Random Access Memory, RAM

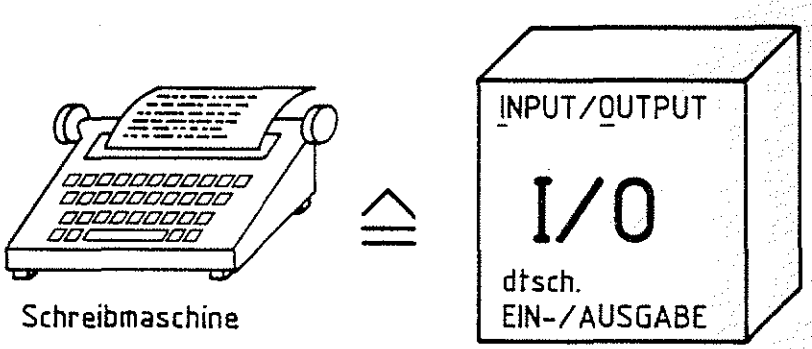


Bild 1e: Schreibmaschine und I/O-Geräte

Die Schreibmaschine für das Ergebnisprotokoll gehört zu den sogenannten Ein- und Ausgabe-Einheiten des Computers (englisch Input/Output, I/O) (Bild 1e).

Ein-Ausgabe = Input-Output

Das Besondere bei dem Menschen im obigen Beispiel ist, daß er durch Ändern der Arbeitsanweisung eine neue, vollkommen andere Aufgabe ausführen kann. Das aber ist auch das Besondere eines Computers, den man durch Ändern der Befehle im Speicher, d.h. durch ein anderes Programm, ebenfalls an eine neue Aufgabenstellung anpassen kann, ohne den gerätetechnischen Aufbau zu verändern.

Ein anderes Programm löst eine andere Aufgabe

Geräte, deren Funktion durch ein Programm veränderbar sind, nennt man "programmierbare" Geräte.
Die Bauteile und Geräte eines Computers nennt man die "HARDWARE". Das Programm, d.h. die Folge der Anweisungen im Speicher, ist veränderbar und heißt "SOFTWARE" des Computers.
Ein Computer arbeitet nicht in gewünschter Weise, wenn Fehler in der Software, d.h. im Programm vorliegen. Hieran kann man schon die Grenzen eines Computers erkennen, der eben nur die vom Menschen vorgegebenen Arbeitsanweisungen in Form des Programms ausführen kann. Die von ihm zu bearbeitende Aufgabe muß also zunächst vom Menschen analysiert und in kleine Arbeitsschritte zerlegt werden. Fehler bei der Festlegung der Arbeitsschritte kann der Computer nicht erkennen, da er stets Anweisung für Anweisung ausführt. Denkvermögen und Kreativität beherrscht er nicht. Dafür aber führt er die vorgegebenen Arbeitsschritte sehr präzise und schnell aus.

programmierbar = durch ein Programm veränderbar
Hardware
Software

Im folgenden soll nun der Begriff (Mikro-)Prozessor erläutert werden. Betrachtet man die Komponenten des Computers, so erkennt man, daß die eigentliche Arbeit von der Control Unit und der ALU ausgeführt wird. Beide Komponenten werden daher auch wie in Bild 2 dargestellt, gerätetechnisch eng miteinander verbunden. Den Verbund von Control Unit und ALU nennt man auch Control Processing Unit (CPU), zu deutsch Zentraleinheit. Der Mikroprozessor ist somit nichts anderes, als eine CPU auf einem (oder wenigen) Chip, d.h. in einem integrierten Schaltkreis (IC).

Control Processing Unit = Zentraleinheit
Mikroprozessor

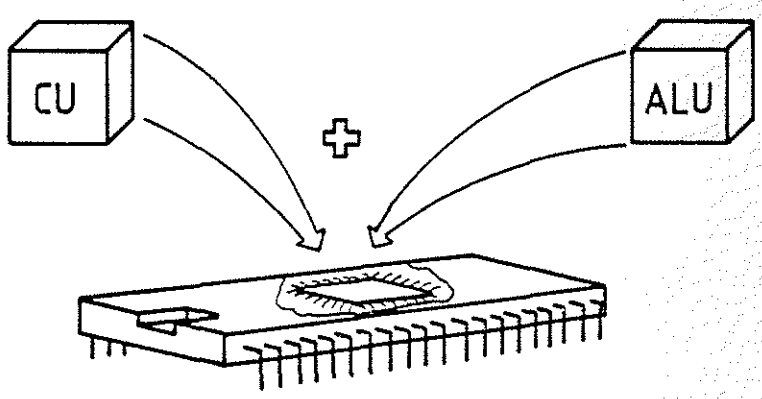


Bild 2: Die Verbindung von CU und ALU zum Mikroprozessor in einem IC

Mikrocomputer

In Bild 3 ist das Blockschaltbild eines Mikrocomputers mit den Komponenten Mikroprozessor, Speicher und Ein-/Ausgabe dargestellt. Der Mikroprozessor, kurz Prozessor genannt, ist über eine Vielzahl von Leitungen mit Speicher und Ein-/Ausgabe verbunden. Über diese Leitungen werden Daten und Informationen in binär-verschlüsselter Form ("0"- und "1"-Signale) ausgetauscht.
Die einzelnen Leitungen sind in Bild 3 durch einen gemeinsamen Strang dargestellt. Pfeile geben dabei die Signalflußrichtungen an.

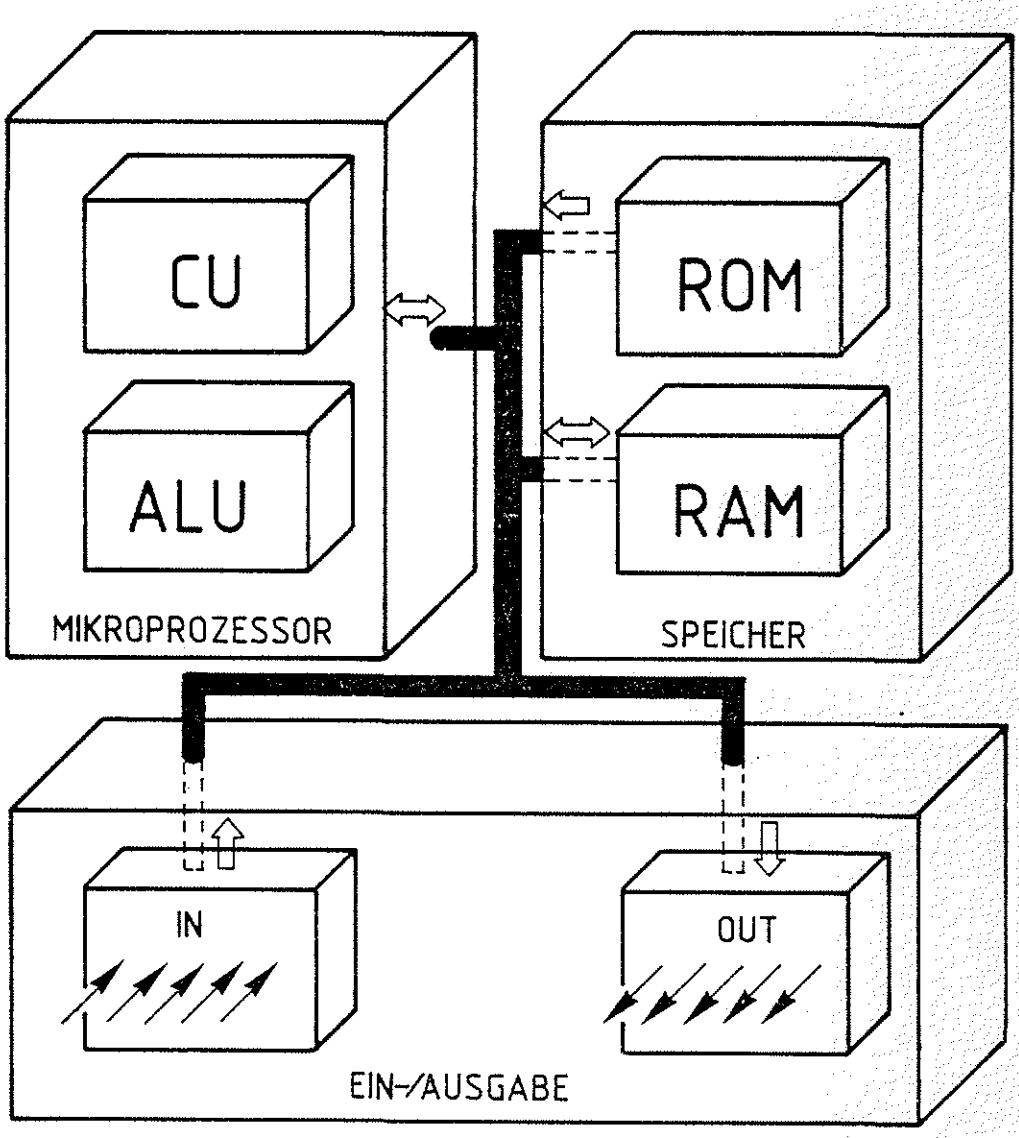


Bild 3: Die Komponenten eines Mikrocomputers verbunden durch ein Leitungsbündel

1.3. Anwendungsbeispiele für Mikrocomputer

Im Gegensatz zu den klassischen Einsatzgebieten der Groß-EDV (Elektronische Datenverarbeitung) wie z.B. der Lohnabrechnung, Personalverwaltung, Auftragsabwicklung usw. werden Mikrocomputer insbesondere für meß-, steuerungs- und regelungstechnische Aufgaben eingesetzt. Zwei typische Anwendungsbeispiele sind in den Bildern 4 bis 6 dargestellt. Im ersten Beispiel handelt es sich um eine Prozeßdaten-Erfassung und -Überwachung.

Prozeßdaten; Erfassung und Überwachung

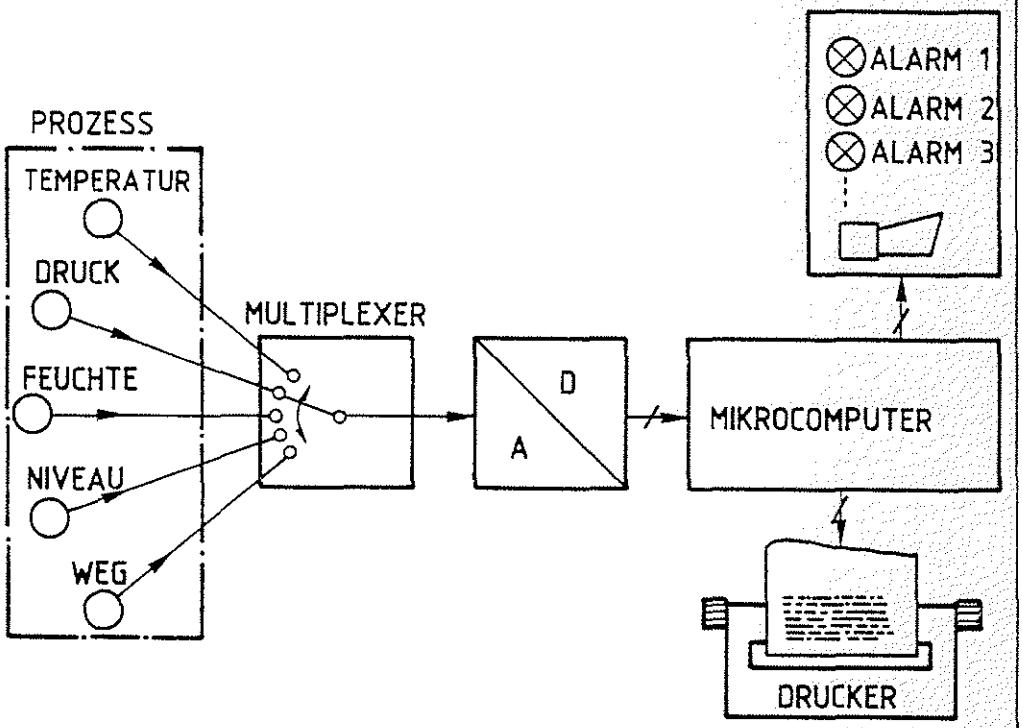


Bild 4: Prozeßdaten; Erfassung- und Überwachung

Der Mikrocomputer erfaßt in vorgegebenen Zeitintervallen über den Meßstellenumschalter (Multiplexer) die gewünschte Prozeßgröße, veranlaßt die Digitalisierung über den Analog-Digital-Wandler (A/D), vergleicht die Meßwerte mit vorgegebenen Grenzwerten, gibt entsprechende Alarme aus und protokolliert die gemessenen Werte auf einem Drucker.

Im zweiten Beispiel (Bild 5) steuert ein Mikrocomputer einen Handhabungsautomaten, auch Roboter genannt, der unbearbeitete Drehteile (Rohlinge) aus einem Magazin dem Drehautomaten zuführt und sie nach Bearbeitung im Magazin für Fertigteile ablegt.

Handhabungsautomat, Roboter

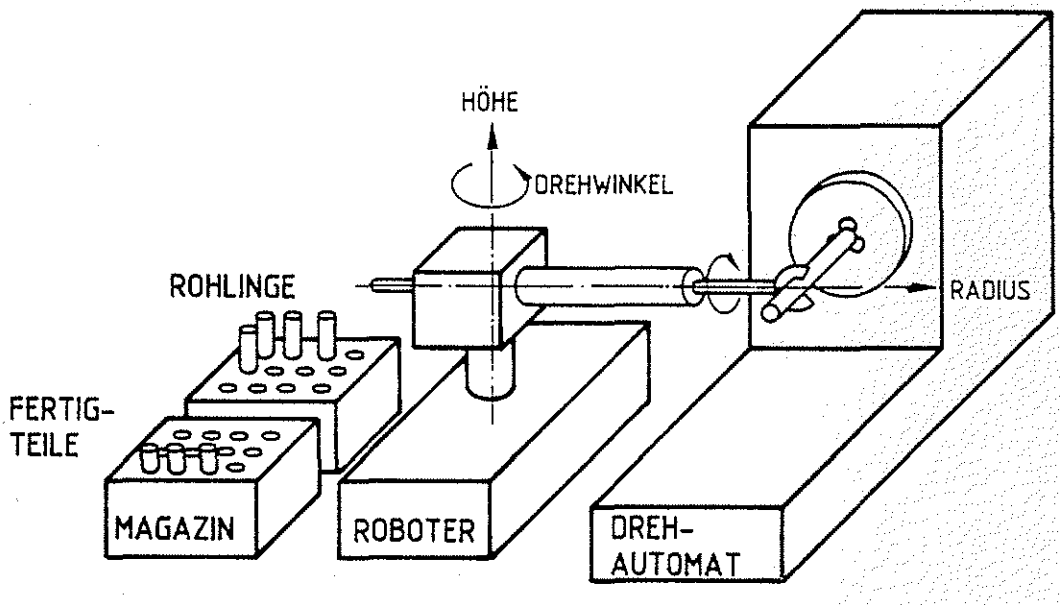


Bild 5: Handhabungsautomat

Dazu muß der Mikrocomputer den Greifarm in die gewünschten Positionen steuern, indem er Stellsignale für die Antriebe und Motoren ausgibt, die Bewegungen z.B. über Impulsgeber kontrolliert und Fertigmeldungen an den Drehautomaten übermittelt, wie "Futter spannen" oder "Ruhelage eingenommen". Der Bewegungsablauf, der sich mit Werkstückgröße, Art des Magazins usw. ändert, wird dem Mikrocomputer im sogenannten "Teach-In-Verfahren" (engl. teach = lehren) eingegeben. Dabei steuert ein Bediener die später automatisch anzufahrenden Positionen des Greifarms zunächst von Hand über Tastersignale an und veranlaßt den Computer, diese Positionen abzuspeichern. Ebenso werden die gewünschten Reaktionen wie "Greifer schließen" usw. eingegeben. Der Name dieses Verfahrens ist demnach dem Vorgehen des Bedieners angepaßt, der den Handhabungsautomaten die auszuführenden Arbeitsschritte lehrt. In Bild 6 ist das Blockschaltbild für diese Mikrocomputer-Steuerung dargestellt.

Teach-In-Verfahren

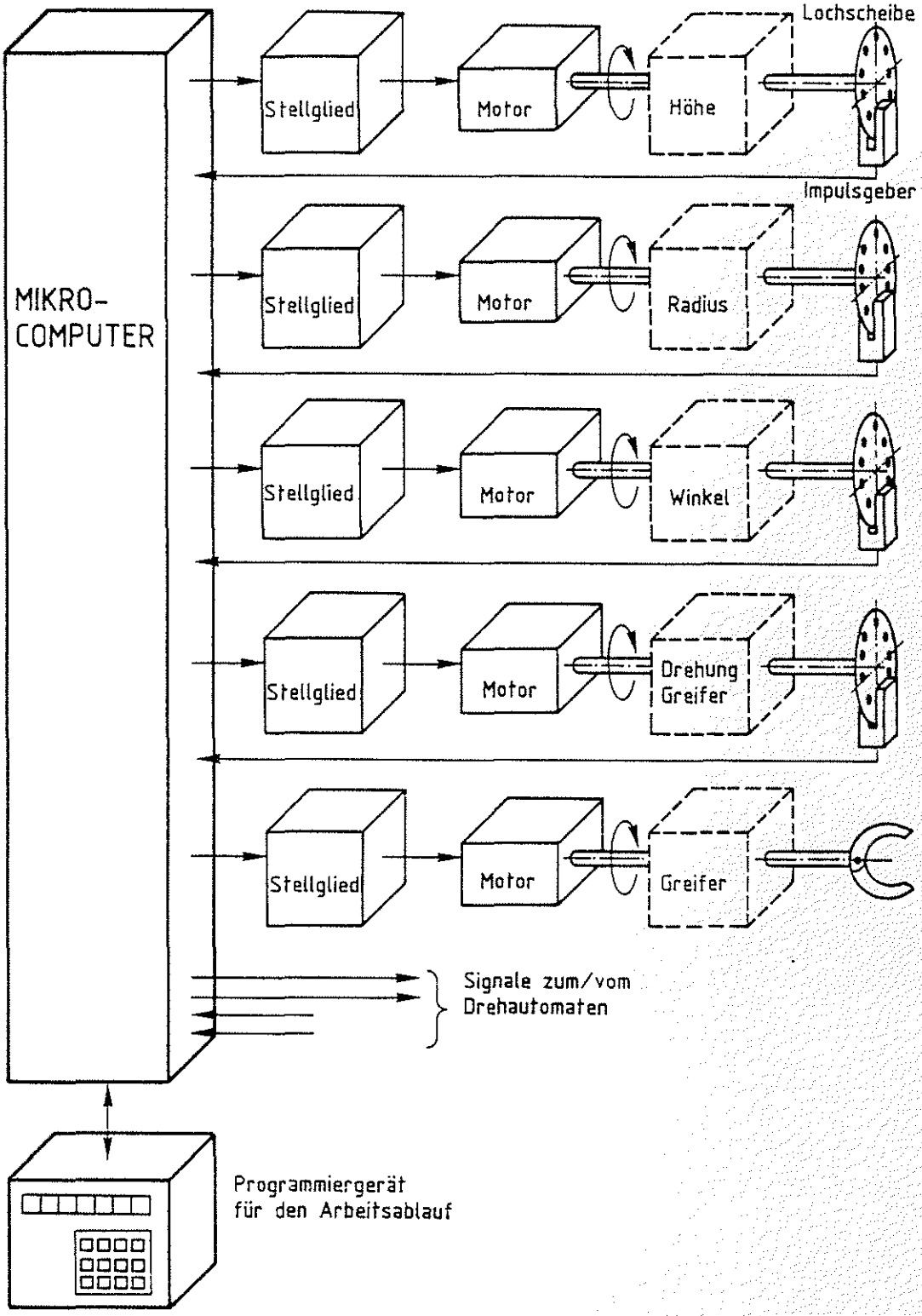


Bild 6: MC- Steuerung eines Handhabungsautomaten

1.4. Prinzip der Bus-Technik

Bus-Technik

Ein besonderes Merkmal von Computern ist die Art und Weise, wie der Prozessor mit dem Speicher und Ein-/Ausgabe-Einheiten verbunden ist und wie der umfangreiche Daten- und Signalfluß vonstatten geht. Ausschließlich angewandt wird die sogenannte Bus-Technik. Dahinter verbirgt sich nichts anderes, als eine Parallelverdrahtung aller Komponenten untereinander. In Bild 7 sind die Komponenten des Computers beispielsweise über vier Busleitungen miteinander verbunden.

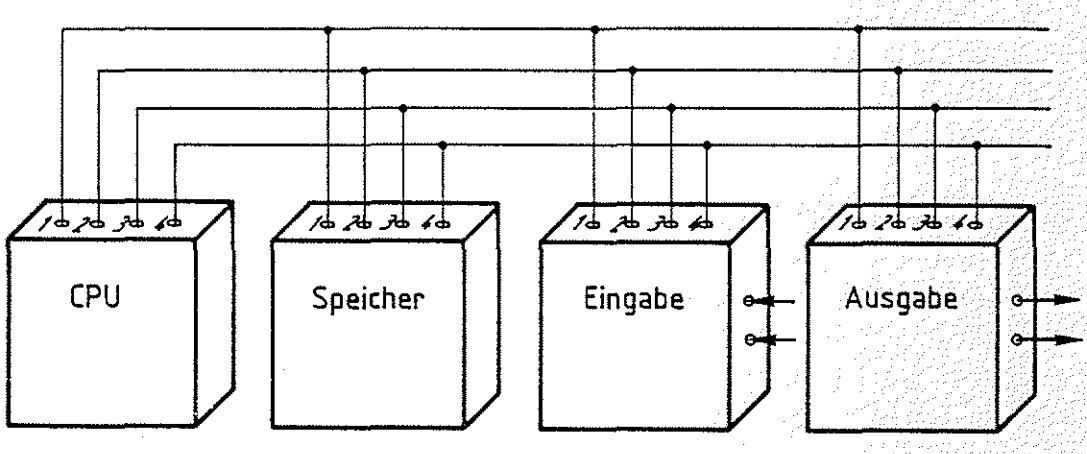


Bild 7: Parallelverdrahtung als Bus- Technik

Der Name "Bus", der in Anlehnung an die von uns gemeinsam benutzten öffentlichen Verkehrsmittel gewählt wurde, soll andeuten, daß der Informationsaustausch über gemeinsam benutzte Leitungen erfolgt.
Damit die über die Busleitungen miteinander verbundenen Geräte nun nicht wahllos durcheinander Informationen auf die Busleitungen schalten, muß ein Gerät die Kontrolle über den Informationsaustausch übernehmen. Diese Aufgabe kommt bei einem Mikrocomputer der CPU zu. Um diese Aufgabe ausführen zu können, sind noch weitere Busleitungen erforderlich. In Bild 8 ist ein Mikrocomputer mit erweitertem Bus-System dargestellt, sein Funktionsablauf wird anschließend erklärt.

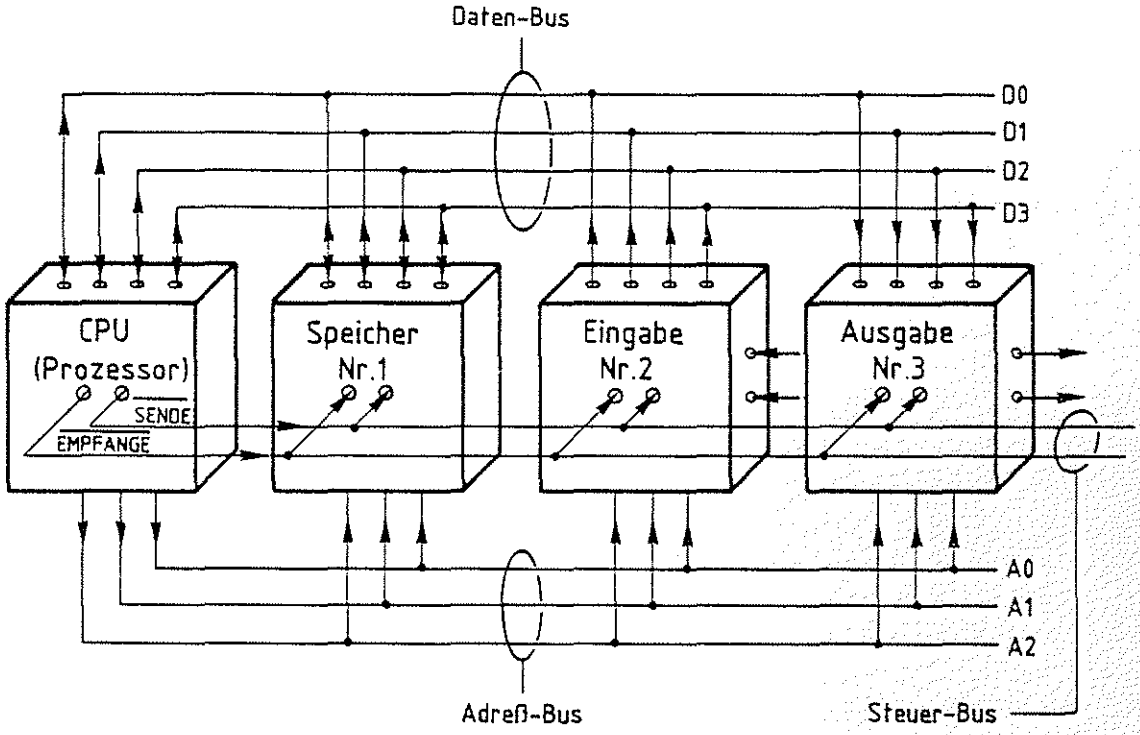


Bild 8: Mikrocomputer mit einfachem Bus-System

Daten-Bus

Adreß-Bus

Dieses Bus-System besteht aus einem vier Bit breiten Daten-Bus, einem drei Bit breiten Adreß-Bus und aus einem zwei Bit breiten Steuer-Bus.
Über den Adreß-Bus zeigt der Prozessor an, welches Gerät sich für die Abgabe bzw. den Empfang von Daten bereithalten soll. Über die drei Adreß-Leitungen können $2^3 = 8$ verschiedene Geräte ausgewählt werden. Bild 9 zeigt eine Tabelle mit den möglichen Signalzuständen auf den Adreß-Leitungen.

Gerätenummer

Jedes am Bus-System angeschlossene Gerät besitzt daher eine Gerätenummer und eine elektronische Schaltung, die das Gerät in Bereitschaft versetzt, sobald die zugehörige Gerätenummer am Adreß-Bus ansteht.

Steuer-Bus

Mit den Steuersignalen "$\overline{\text{SENDE}}$" bzw. "$\overline{\text{EMPFANGE}}$" aktiviert der Prozessor über den Steuer-Bus dann das ausgewählte Gerät, damit es entweder Daten auf den Daten-Bus schaltet (sendet) oder Daten übernimmt (empfängt). Die Pfeile in den Bus-Leitungen zeigen an, in welche Richtung die Informationen auf diesen Leitungen "fließen".

Adreß-Bus			
A2	A1	A0	Geräte-Nr.
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Bild 9: Mögliche Signalzustände auf drei Adreß-Leitungen zur Geräte-Auswahl

Benötigt der Prozessor beispielsweise Daten vom Eingabegerät mit der Nummer 2, so führt er folgende Arbeitsschritte aus: Der Prozessor ...

Lesen von Daten

- wählt Gerät 2 aus, indem er den Signalzustand "010" auf den Adreß-Leitungen aussendet;
- veranlaßt Gerät 2 seine Daten auf den Daten-Bus zu schalten, indem er das Steuer-Signal "$\overline{\text{SENDE}}$" auf "0"-Signal *) schaltet;
- liest den Signal-Zustand vom Daten-Bus;
- beendet den Vorgang, indem er die Steuerleitung "$\overline{\text{SENDE}}$" auf "1"-Signal schaltet.

*) Eine Steuerleitung, die mit 0-Signal (od. Low-Pegel) eine Aktivität auslöst, bezeichnet man als "aktiv Low" und kennzeichnet sie durch eine Überstreichung ihres Namens.

aktiv Low

Sollen die empfangenen Daten weitergeleitet werden, z.B. an das Gerät mit der Nummer 3, so müssen zusätzlich noch folgende Arbeitsschritte ausgeführt werden: Der Prozessor...

Schreiben von Daten

- wählt Gerät 3 aus, indem er den Signalzustand "011" auf den Adreß-Leitungen aussendet;

- schaltet die Daten auf den Daten-Bus;
- veranlaßt Gerät 3 die Daten zu übernehmen, indem er die Steuerleitung "$\overline{\text{EMPFANGE}}$" auf "0"-Signal schaltet;
- beendet den Vorgang, indem er die Steuerleitung "$\overline{\text{EMPFANGE}}$" auf "1"-Signal schaltet.

Da die Adreß- und Steuer-Signale ausgehend vom Prozessor die Leitungen nur in einer Richtung durchlaufen, nennt man Adreß- und Steuer-Bus unidirektional (eine Signalrichtung möglich). Im Gegensatz dazu können Daten-Signale den Daten-Bus in beiden Richtungen durchlaufen. Daher bezeichnet man den Daten-Bus als bidirektional (zwei Richtungen). Der wesentliche Vorteil der Bus-Technik ist, daß über eine relativ geringe Anzahl von Leitungen eine Vielzahl von Geräten/Komponenten miteinander Daten austauschen können. Ohne Anwendung der Bus-Technik wäre es auch nicht möglich, den Mikroprozessor in einem Gehäuse mit häufig nur 40 Anschlüssen unterzubringen. Getrennte Verbindungen (Punkt-zu-Punkt-Verbindungen) zum Speicher und den Ein-/Ausgabe-Einheiten würden erheblich mehr Leitungen erfordern. Bus-Systeme sind außerdem leicht erweiterbar und ermöglichen insbesondere einen modularen (modular = bausteinartig) Aufbau. Der Hauptnachteil ist, daß der Datenaustausch zwischen mehreren Komponenten nur nacheinander (sequentiell) erfolgen kann, weshalb für die Verarbeitung mehr Zeit beansprucht wird.

unidirektional

bidirektional

modularer Aufbau

Das in Bild 8 dargestellte einfache Bus-System reicht für Aufgaben der Praxis nicht aus. Mit den drei Adreß-Leitungen lassen sich nur 8 (2^3) Geräte anwählen (adressieren), praktisch müssen jedoch wesentlich mehr Adressen erzeugt werden können. Bild 10 zeigt einen Mikrocomputer mit einem Bus-System, das den Erfordernissen der Praxis gerecht wird.

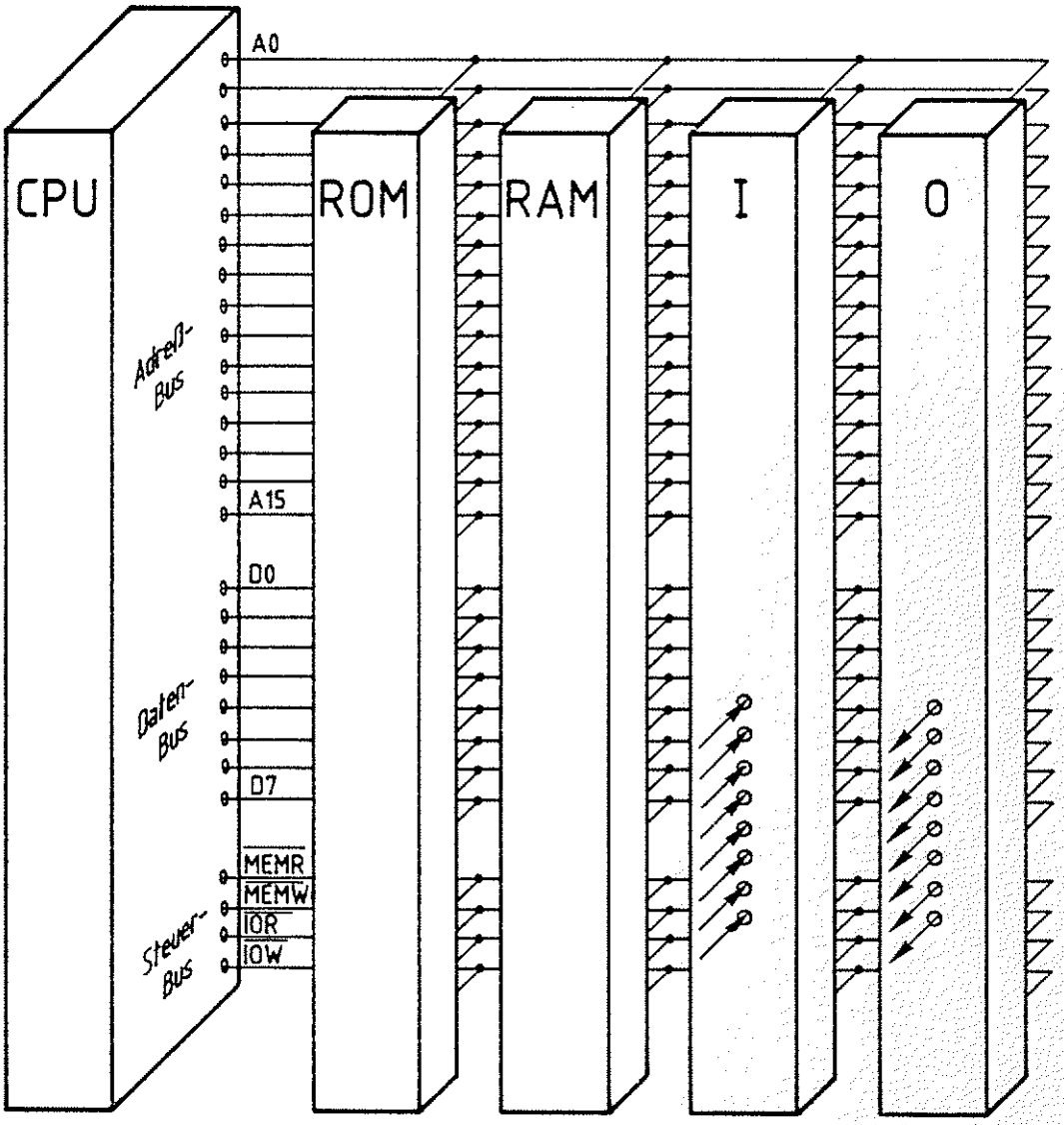


Bild 10: Mikrocomputer mit erweitertem Bus-System

Der Adreßbus dieses Mikrocomputers verfügt über 16 Leitungen. Damit wäre es möglich, 2^{16} = 65536 Geräte an dieses Bus-System anzuschließen. Bezeichnet man nun den Speicherplatz für eine Information als Gerät, so lassen sich also 2^{16} (65536) verschiedene Speicherplätze für die Arbeitsanweisungen und Daten ansprechen. Über die acht Datenleitungen können 2^{8} (256) verschiedene Signalzustände übertragen werden. Häufig unterscheidet man bei den Steuersignalen solche, die entweder nur die Speicher oder nur die Ein-/Ausgabe-Einheiten aktivieren.

Funktionen der Steuersignale

Bezeichnungen und zugeordnete Funktionen der in Bild 10 dargestellten Steuersignale:

$\overline{\text{MEMR}}$ (MEMORY READ, dtsch. Speicher lesen), veranlaßt den Speicher, Daten auf den Daten-Bus zu schalten. Aktiv Low-Signal.

$\overline{\text{MEMW}}$ (MEMORY WRITE, dtsch. Speicher schreiben), veranlaßt den Speicher, Daten vom Daten-Bus zu übernehmen. Aktiv-Low-Signal.

$\overline{\text{IOR}}$ (INPUT/OUTPUT READ, dtsch. Ein-/Ausgabe-Einheit lesen), veranlaßt die Eingabe-Einheit, den Signalzustand an den Eingangsleitungen auf den Daten-Bus zu schalten. Aktiv-Low-Signal.

$\overline{\text{IOW}}$ (INPUT/OUTPUT WRITE, dtsch. Ein-/Ausgabe-Einheit schreiben), veranlaßt die Ausgabe-Einheit, den Signalzustand vom Daten-Bus zu übernehmen und zu den Ausgangsleitungen durchzuschalten. Aktiv-Low-Signal.

Darüberhinaus gibt es oft noch eine Vielzahl anderer Steuersignale, die aber zunächst für das Verständnis der Funktion unbedeutend und je nach Hersteller des Mikroprozessors unterschiedlich sind.

Erinnern wir uns noch einmal an die Arbeitsweise eines Menschen. Wenn er einen Arbeitsschritt ausgeführt hat, liest er die nächste Anweisung in der Folgezeile der Arbeitsanleitung. Hat der Prozessor eine Anweisung ausgeführt, so sendet er die nächst höhere Speicherplatznummer auf dem Adreß-Bus aus, aktiviert das Steuersignal "$\overline{\text{MEMR}}$" und liest die vom Speicher bereitgestellte Anweisung vom Daten-Bus. Dann führt er die Anweisung aus und wiederholt den Lesevorgang, nachdem er erneut die Speicherplatznummer um eins erhöht hat usw.. Es gibt die verschiedensten Anweisungen, maximal jedoch "nur" 256 (2^8). Dies ist abhängig von der Anzahl der Datenleitungen. Z.B. gibt es sogenannte Verarbeitungsbefehle, die den Prozessor veranlassen, Zahlen zu addieren, zu subtrahieren oder zu vergleichen.

Arbeitsanweisung = Befehl

Verarbeitungsbefehle

Transportbefehle

Andere, die Transportbefehle, veranlassen den Prozessor, Daten im RAM zwischenzuspeichern oder abzurufen, den Signalzustand an den Eingangsleitungen der Eingabe-Einheit zu ermitteln oder einen ganz bestimmten Signalzustand an den Ausgangsleitungen der Ausgabe-Einheit einzustellen.
Die Mikroprozessoren bezeichnet man in Abhängigkeit von der Anzahl der Datenleitungen als 1-Bit-, 2-Bit-, 4-Bit-, 8-Bit- oder 16-Bit-Mikroprozessoren. Mittlerweile gibt es schon 32-Bit-Mikroprozessoren.

1.5. Beschreibung von Bus-Signalen

Ein kleines Problem beim Umgang mit Computern ist die Beschreibung des Signalzustandes auf den Bus-Leitungen. Soll z.B. im Fehlerfall überprüft werden, ob der Prozessor auf dem Adreß-Bus die Speicherplatznummer 16783 aussendet, so muß zunächst die zugehörige Dualzahl z.B. mit Hilfe des Divisionsverfahrens ermittelt werden:

Dezimal-zu-Dual-wandlung durch Divisionsverfahren

```
16783 : 2 = 8391 R 1 <---- niederwertigste Stelle,
 8391 : 2 = 4195 R 1       engl. least significant
 4195 : 2 = 2097 R 1       bit, kurz LSB
 2097 : 2 = 1048 R 1
 1048 : 2 =  524 R 0
  524 : 2 =  262 R 0
  262 : 2 =  131 R 0
  131 : 2 =   65 R 1
   65 : 2 =   32 R 1
   32 : 2 =   16 R 0
   16 : 2 =    8 R 0
    8 : 2 =    4 R 0
    4 : 2 =    2 R 0
    2 : 2 =    1 R 0
    1 : 2 =    0 R 1 <---- höchstwertigste Stelle,
                           engl. most significant
                           bit, kurz MSB
```

Nach einer langen Rechnung ergibt sich der folgende Signalzustand auf den Adreß-Leitungen:

Adreß-Leitung	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Signal-Zustand	0	1	0	0	0	0	0	1	1	0	0	0	1	1	1	1

Einen solchen Signalzustand kann man sich aber sehr schlecht merken, so daß man dafür eine besondere Art der Verschlüsselung eingeführt hat. Betrachten wir dazu zunächst alle möglichen Signalzustände auf vier Leitungen und fassen diese in einer Tabelle (Bild 11) zusammen. Zur Kennzeichnung der möglichen Zustände werden sie einfach durchnummeriert. Um alle Signalzustände mit nur einer Kennziffer zu versehen, werden für die letzten sechs Zustände die Buchstaben A bis F verwendet.

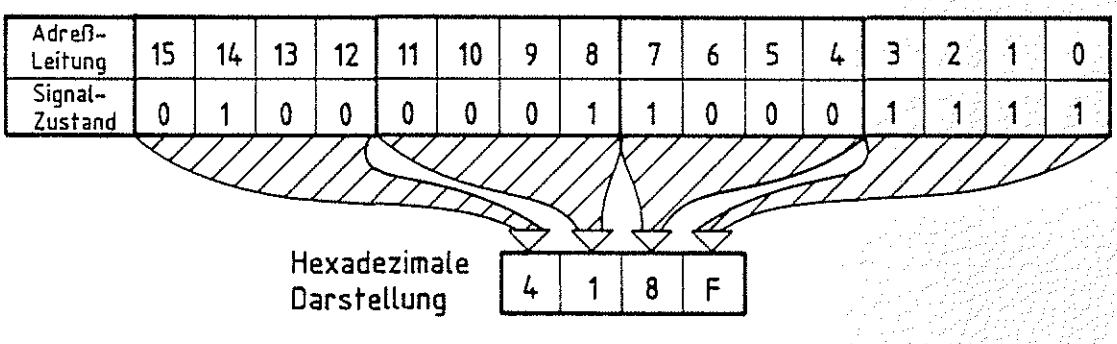
A3	A2	A1	A0	Kennziffer
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	A
1	0	1	1	B
1	1	0	0	C
1	1	0	1	D
1	1	1	0	E
1	1	1	1	F

Bild 11: Die möglichen Signalzustände auf vier Bus-Leitungen

In Anlehnung an unser übliches Zahlensystem mit der Basis 10 (10 Ziffern, 0...9) entspricht die Durchnummerierung der Signalzustände mit 16 Ziffern (0...9, A...F) einem Zahlensystem mit der Basis 16, so daß man diese Zahlen auch Hexadezimalzahlen (hexa = sechs, dezi = zehn) oder Sedezimalzahlen (sedezi = sechzehn) nennt.

Hexadezimal- oder Sedezimal-Zahlen

Faßt man nun die sechzehn Adreß-Leitungen zu vier Tetraden (Viererbündel) zusammen, so kann der Signalzustand der Leitungen wesentlich kürzer beschrieben werden:



Für die Beschreibung des Signalzustandes auf den 16 Adreß-Leitungen genügt so eine vierstellige und für die acht Daten-Leitungen eine zweistellige Hexadezimalzahl.

1.6. Darstellung von Bus-Verbindungen

Die Leitungen eines Bus-Systems werden in Schaltbildern meist nicht einzeln gezeichnet, sondern wie in Bild 12 gezeigt, schematisch durch breite Verbindungsbalken dargestellt. In die Balken schreibt man, welchen Teil des Busses sie symbolisieren. Die Richtung des Datenflusses auf den Bus-Leitungen wird zusätzlich durch Pfeile gekennzeichnet.

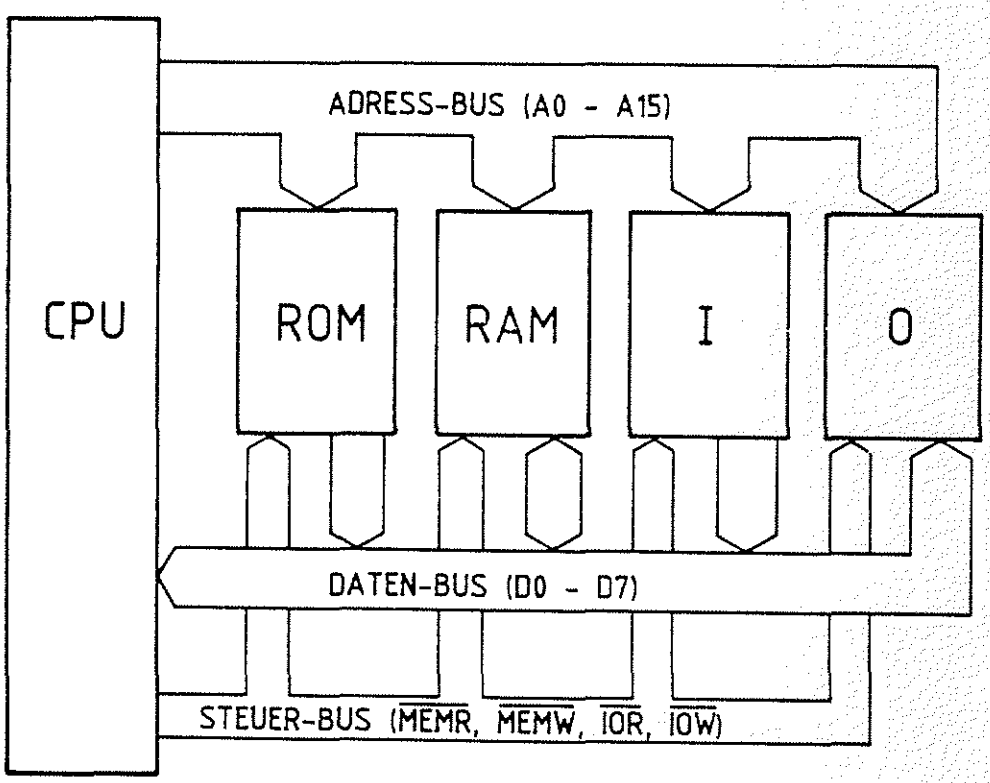


Bild 12: Darstellung der Bus-Leitungen

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

AUFBAU VON DV-ANLAGEN UND BUS-SYSTEMEN

BFZ/MFA 10.1.

ÜBUNGSTEIL 1

Übungsteil 1

In den folgenden Arbeitsschritten werden Sie Messungen am Bus-System eines Mikrocomputers durchführen.

Dazu benötigen Sie:

- 1 Baugruppenträger mit Busverdrahtung (BFZ/MFA 0.1.)
- 1 Bus-Abschluß (BFZ/MFA 0.2.)
- 1 Trafo-Einschub (BFZ/MFA 1.1.)
- 1 Spannungsregelung (BFZ/MFA 1.2.)

} zusammengebaut und geprüft nach FPÜ BFZ/MFA 1.2. Arbeitsblatt A7

- 1 Bus-Signalgeber (BFZ/MFA 5.1.)
- 1 Bus-Signalanzeige (BFZ/MFA 5.2.)
- 1 Adapterkarte 64polig (BFZ/MFA 5.3.)
- 1 Logik-Tester oder Vielfach-Meßinstrument
- 2 Meßleitungen und Meßklips

Allgemeine Hinweise zur Durchführung der Übungen:

- Die Einschübe dürfen nur bei abgeschalteter Betriebsspannung gesteckt oder gezogen werden
- Aufgrund der Busverdrahtung können die Baugruppen in beliebige Steckplätze gesteckt werden
- Messungen an den Bus-Leitungen sollten mit Hilfe der Adapterkarte durchgeführt werden
- Den logischen Signalen "0" und "1" sind die folgenden Pegel zugeordnet:

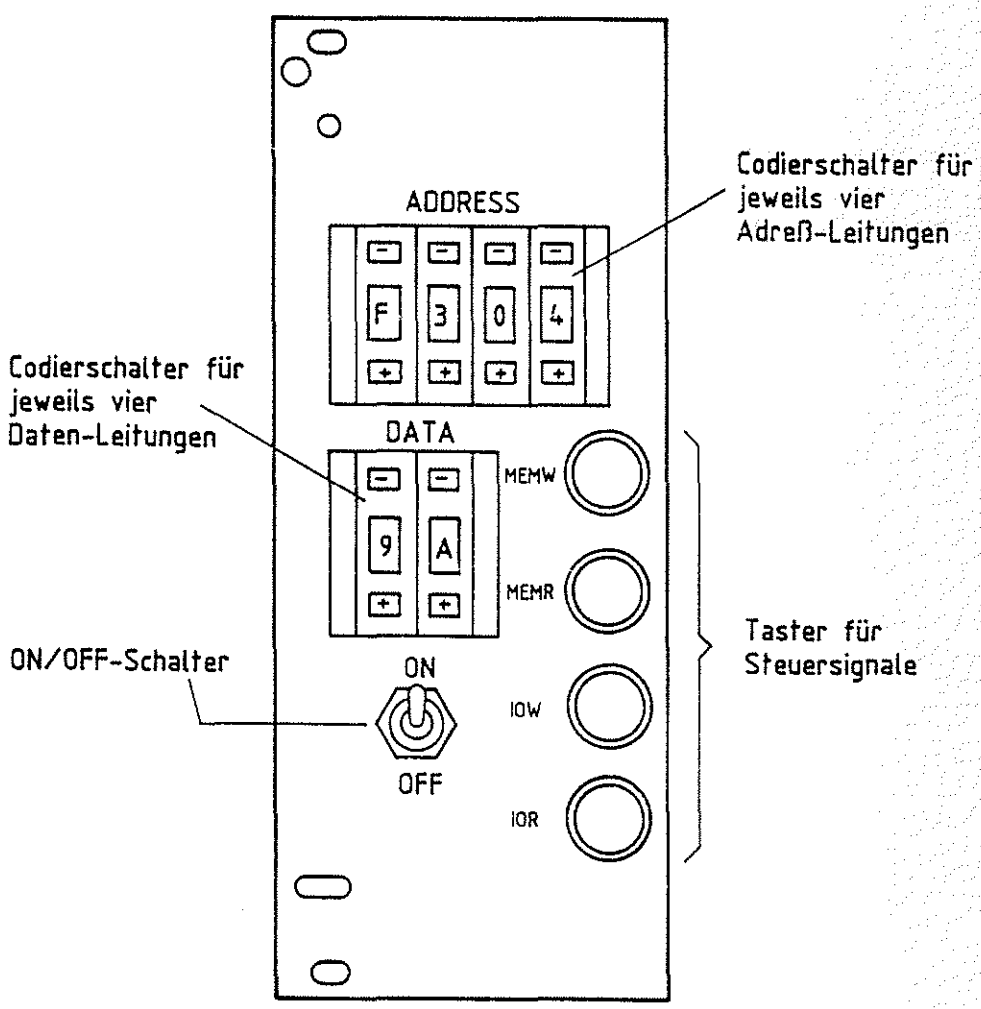
  log. "0" ≙ 0...0,8 V (LOW)
  log. "1" ≙ 2,4...5 V (HIGH)

- Alle zur Messung an den Baugruppen vorgegebenen Arbeitsblätter enthalten:
  = Angaben über den Sinn der jeweiligen Messung
  = Angaben über einzustellende Bedingungen (z.B. Schalterstellungen)
  = Aufgabenstellungen, ggf. mit Hinweisen zu möglichen Fehlern.

Bedienungshinweise:

Bus-Signalgeber

In der Frontplatte des Bus-Signalgebers befinden sich sogenannte Codierschalter, mit denen die Adreß- und Daten-Signale im Hexadezimalcode eingestellt werden können. Jeder einzelne Codierschalter liefert die 16 möglichen Signalkombinationen für 4 Busleitungen. Für die 16 Leitungen des Adreß-Busses sind daher vier Codierschalter vorhanden, für die 8 Leitungen des Daten-Busses zwei.
Für die Steuersignale zum Aktivieren der Speicher und Ein-/Ausgabe-Einheiten sind vier Taster vorgesehen.
Mit dem ON/OFF-Schalter kann die Signalausgabe des Bus-Signalgebers gesperrt (OFF) werden, so daß die eingestellten Adreß-, Daten- und Steuersignale nicht auf den System-Bus gelangen können. Der Schalter muß sich bei dieser Übung in Stellung "ON" befinden.

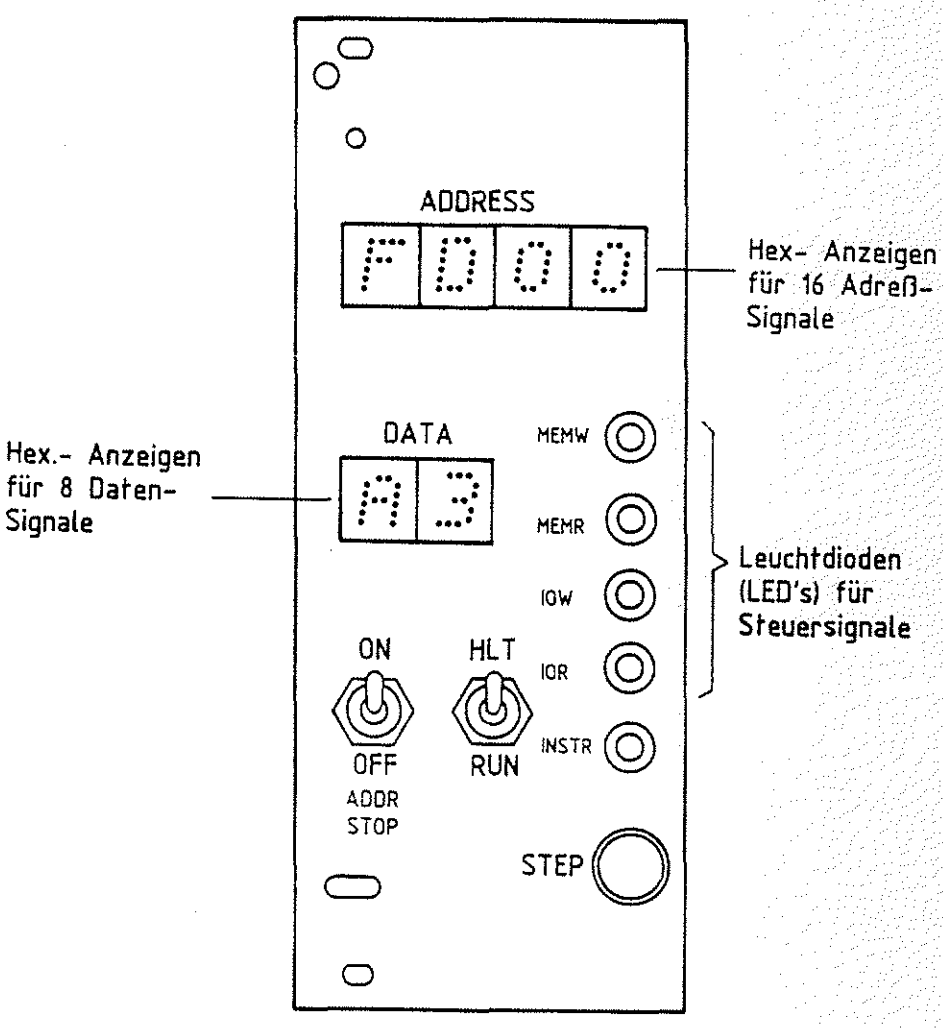


Bus-Signalgeber

Beachten Sie bitte, daß die Kurzzeichen der vier Steuersignale in den Schaltungsunterlagen mit einem Negationsstrich versehen sind. Bei Aktivierung eines Steuersignals führt die zugehörige Signalleitung logisch "0"-Signal ("aktiv-Low-Signal").

Bus-Signalanzeige

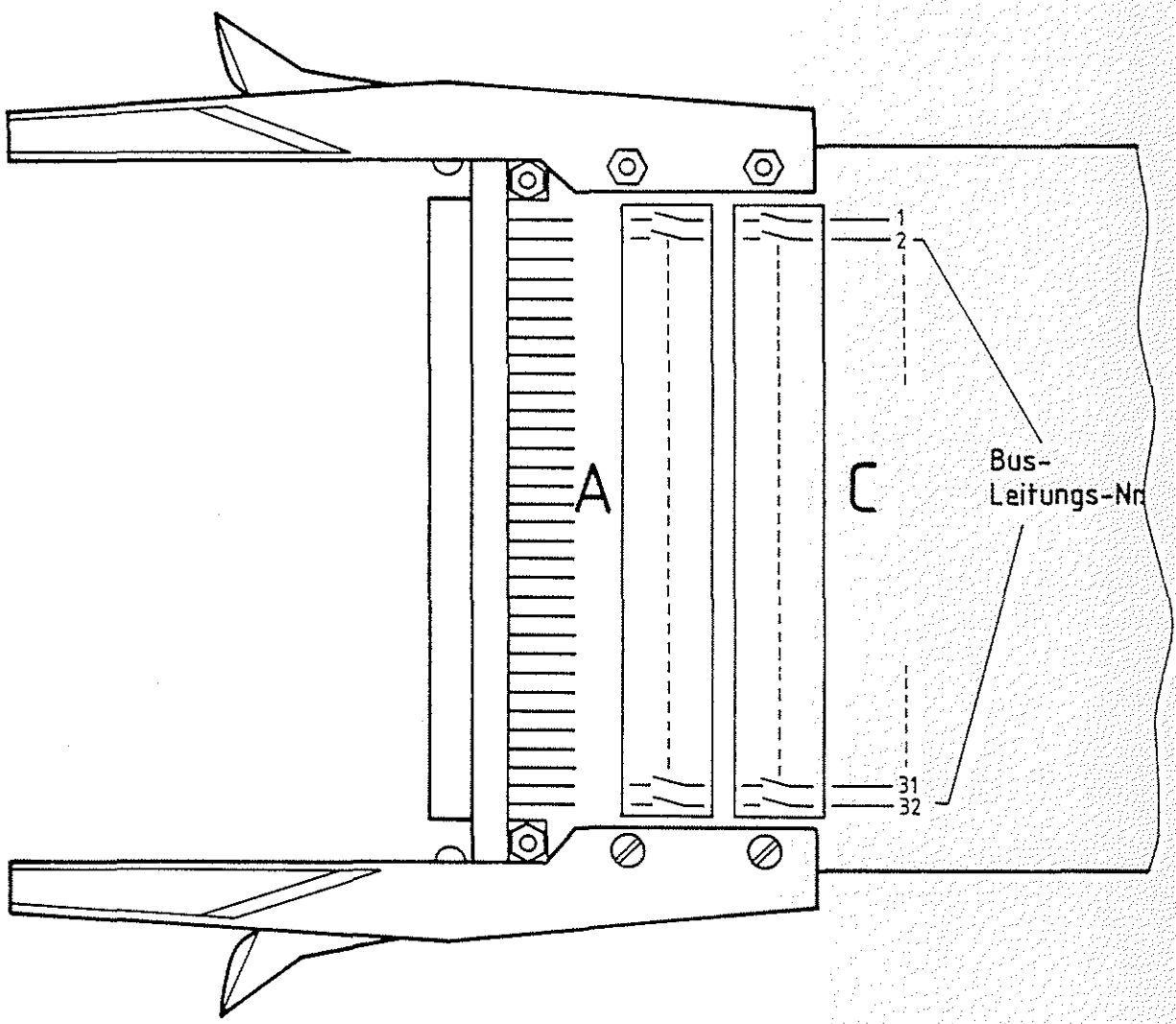
Der Signalzustand des Bus-Systems kann mit der Bus-Signalanzeige sichtbar gemacht werden. Die Anzeige der Adreß- und Daten-Signale erfolgt über spezielle Leuchtdioden-Anzeigen im Hexadezimal-Code. Für den Signalzustand der Steuerleitungen dagegen sind einzelne Leuchtdioden vorgesehen. Die Schalter und der Taster in der Frontplatte der Baugruppe werden in dieser Übung nicht benötigt, ihre Stellung ist beliebig.



Bus-Signalanzeige

Adapterkarte 64polig

Die untenstehende Abbildung zeigt die Belegung der Schalter/Brücken an der Frontseite der Baugruppe. Hier können die Signale des Bus-Systems gemessen werden.



Adapterkarte 64polig

Die Tabelle zeigt die Signalbelegung des Bus-Systems.

Stift-Nr.	Reihe		Bemerkung
	a	c	
1	+5V	+5V	
2			
3			
4	D0	D1	
5	D2	D3	
6	D4	D5	
7	D6	D7	
8			
9	$\overline{\text{IOW}}$	$\overline{\text{MEMW}}$	
10	$\overline{\text{IOR}}$	$\overline{\text{MEMR}}$	
11			
12			
13			
14			
15			
16		A0	
17	A1	A2	
18	A3	A4	
19	A5	A6	
20	A7	A8	
21	A9	A10	
22	A11	A12	
23	A13	A14	
24	A15		
25			
26			
27			
28			
29			
30			
31			
32	0V	0V	

Aufbau von DV-Anlagen und Bus-Systemen

Name:

Übungsteil 1

Datum:

A1

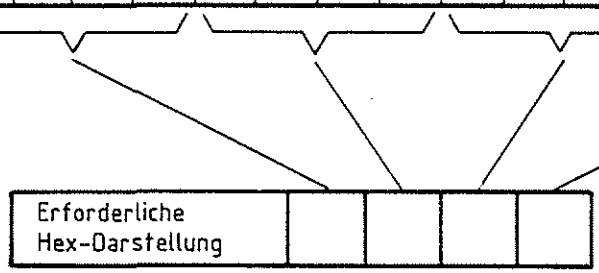
Messen der Adreß-Signale auf dem Adreß-Bus

Die notwendigen Adreß-Signale werden vom Bus-Signalgeber auf den Adreß-Bus geschaltet.

Stecken Sie den Bus-Signalgeber in den Baugruppenträger und schalten Sie die Spannungsversorgung ein. Schalter ON/OFF→ON*.

Stellen Sie den folgenden Signalzustand am Adreß-Bus ein. Ermitteln Sie dazu zuerst die hexadezimale Darstellung der Adreß-Signale. Überprüfen Sie die Spannungspegel auf den 16 Adreß-Leitungen mit Hilfe eines Logik-Testers oder Vielfachmeßinstruments und tragen Sie die Meßergebnisse in die Tabelle ein.

Adreß-Leitung	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
gewünschter Signalzustand	1	0	1	0	0	1	1	0	1	1	0	0	1	0	1	1



Adreß-Leitung	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Stift-Nr.																
gemessener Pegel																

* ON/OFF→ON bedeutet: Schalter ON/OFF auf ON stellen.

Aufbau von DV-Anlagen und Bus-Systemen

Name:

Übungsteil 1

Datum:

A2

Messen der Daten-Signale auf dem Daten-Bus

Die notwendigen Daten-Signale werden vom Bus-Signalgeber auf den Daten-Bus geschaltet.

Stellen Sie an den Daten-Bus-Schaltern den Zustand "A5" ein. Überlegen Sie sich vor der meßtechnischen Überprüfung, welche logischen Signalzustände auf den Leitungen vorhanden sein müssen. Tragen Sie diese in die Tabelle ein.

Codierschalter - Einstellung	A	5

Daten-Leitung	7	6	5	4	3	2	1	0
erforderlicher Pegel								

Überprüfen Sie Ihre Überlegungen durch Messung der Pegel.

Daten-Leitung	7	6	5	4	3	2	1	0
Stift - Nr.								
gemessener Pegel								

Welcher Dezimalzahl entspricht die Bit-Kombination auf den Datenleitungen?

Dezimal-Darstellung			

Anleitung:

Zur Umwandlung einer Dual- in eine Dezimal-Zahl kann z.B. das Stellenwertprinzip angewendet werden, bei dem jede Stelle der Dualzahl mit der zugehörigen Zweierpotenz (2^0, 2^1, 2^2,...) multipliziert und anschließend aufaddiert wird.

Aufbau von DV-Anlagen und Bus-Systemen

Name:

Übungsteil 1

Datum:

A3

Überprüfung der Steuersignale

Bei Betätigung der Steuersignal-Taster auf dem Bus-Signalgeber werden die entsprechenden Steuer-Bus-Leitungen aktiviert.

Betätigen Sie nacheinander die vier Steuer-Signale und messen Sie die zugehörigen Spannungspegel auf den Leitungen des Steuer-Busses.

Steuersignal-Taste	Stift-Nr.	Taster	Gemessener Pegel
MEMW		AUS	
		EIN	
MEMR		AUS	
		EIN	
IOW		AUS	
		EIN	
IOR		AUS	
		EIN	

Aufbau von DV-Anlagen und Bus-Systemen

Name:

Übungsteil 1

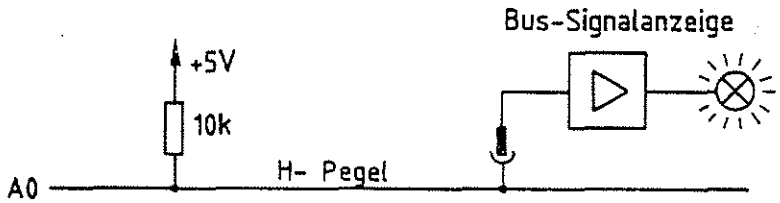
Datum:

A4.1

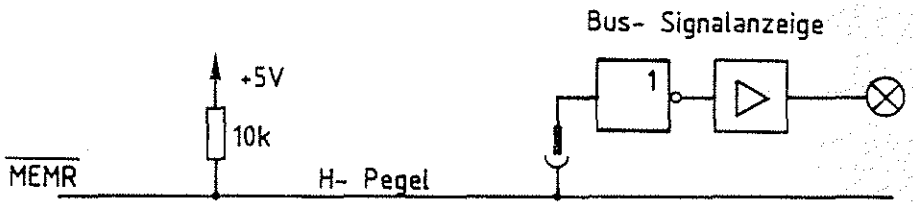
Überprüfen der Wirkung der Steuersignale auf der Bus-Signalanzeige

Stecken Sie anstelle des Bus-Signalgebers zunächst nur die Bus-Signalanzeige in den Baugruppenträger (Betriebsspannung vorher ausschalten).

Nach dem Einschalten der Betriebsspannung müssen alle Hexadezimalanzeigen "F" anzeigen. Das bedeutet, daß alle Adreß- und Daten-Leitungen H-Pegel führen. Der Grund wird aus der folgenden Skizze ersichtlich, in der die Adreß-Leitung A0 als Beispiel dargestellt ist. Alle Leitungen des Bus-Systems sind, wie in der Skizze dargestellt, über sogenannte Pull-up-Widerstände mit der Betriebsspannung verbunden. Wenn keine Baugruppe am Bus-System Signale auf den Bus schaltet, führen alle Leitungen H-Pegel.



Anders ist es bei den vier Steuersignalen. Die vier Leuchtdioden bleiben dunkel, weil der aktive Zustand eines Steuersignals durch L-Pegel auf der zugehörigen Steuerleitung signalisiert wird. Weil die Leuchtdioden nur den aktiven Zustand anzeigen sollen, werden sie auf der Bus-Signalanzeige mit den invertierten Signalen der Steuerleitungen angesteuert.



A4.2

Stecken Sie nun zusätzlich noch den Bus-Signalgeber in den Baugruppenträger.

Stellen Sie beliebige Adreß- und Daten-Signale (allerdings nicht alles "F") ein und betätigen Sie nacheinander die vier Steuersignale. Beobachten Sie dabei, welcher Signalzustand sich am Daten-Bus einstellt und versuchen Sie das Verhalten zu begründen.

betätigte Steuersignal- Taste	Daten	
	eingestellt	angezeigt
MEMW		
MEMR		
IOW		
IOR		

Die Ursache für das Verhalten:
Immer dann, wenn der Speicher oder die Eingabe-Baugruppen durch die Steuersignale $\overline{\text{MEMR}}$ bzw. $\overline{\text{IOR}}$ aufgefordert werden, ihrerseits Daten auf den Daten-Bus zu schalten, müssen die Datensignale des Bus-Signalgebers vom Daten-Bus getrennt werden. Da sich keine weitere Baugruppe im Baugruppenträger befindet, führen beim Auftreten dieser Steuersignale alle Datenleitungen H-Pegel und die Bus-Signalanzeige zeigt "FF" an.

Ende Übungsteil 1

FACHTHEORETISCHE ÜBUNG
MIKROCOMPUTER – TECHNIK

AUFBAU VON DV-ANLAGEN UND BUS-SYSTEMEN

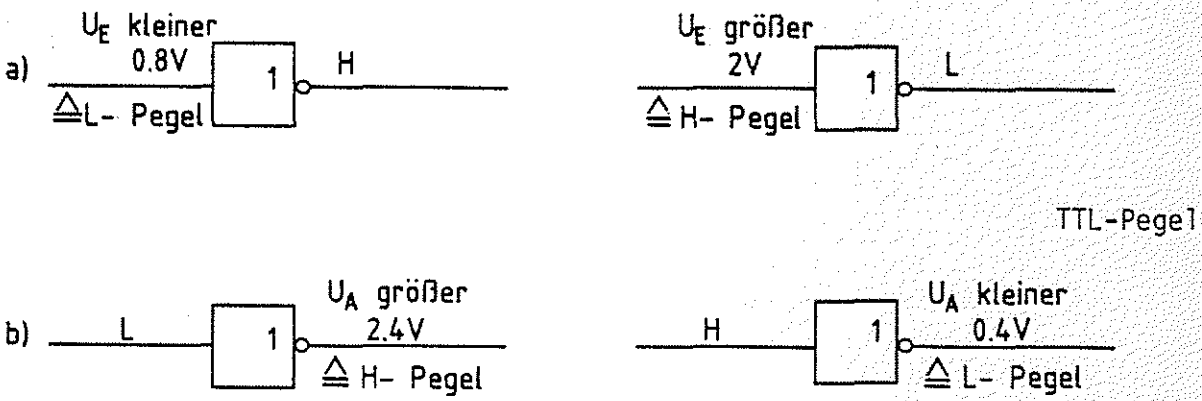
BFZ/MFA 10.1.

THEORIETEIL 2

2.1. Tristate-Technik

Im folgenden soll näher beleuchtet werden, welche Eigenschaften die digitalen Bausteine (Gatter, usw.) aufweisen müssen, die an die Bus-Leitungen angeschlossen werden. Die Anschlüsse vieler Mikroprozessor-Bausteine sind TTL-kompatibel, d.h., daß ihre Schalteigenschaften der Transistor-Transistor-Logikfamilie angepaßt sind. Die Bezeichnung TTL resultiert aus der Art der Schaltungstechnik, die im wesentlichen nur Transistoren für den Aufbau der Verknüpfungsschaltungen verwendet. Unter anderem bedeutet dies, daß die Spannungswerte an den Ein- und Ausgängen der IC's ganz bestimmte Grenzwerte nicht überschreiten dürfen. Betrachten wir dazu z.B. einen einfachen TTL-Inverter. In Bild 13a sind die Grenzwerte für die Eingangsspannungen eingetragen.

TTL-Schaltkreis-Familie



TTL-Pegel

Bild 13 a und b: Eingangs- und Ausgangspegel für die TTL-Technik

Die Eingangsschaltung des TTL-Gatters erkennt L-Signal, wenn die Eingangsspannung kleiner als 0,8 V ist und H-Signal, wenn sie größer als 2 V ist. Bild 13b zeigt die Grenzwerte für die Ausgangsspannungen.
Ein TTL-Gatter liefert bei H-Signal am Ausgang eine Spannung, die größer als 2,4 V und bei L-Signal kleiner als 0,4 V ist.

Bild 14 zeigt die Innenschaltung eines TTL-Inverters, weil für die folgenden Überlegungen die Kenntnis der Ausgangsstufe der Gatter wichtig ist.

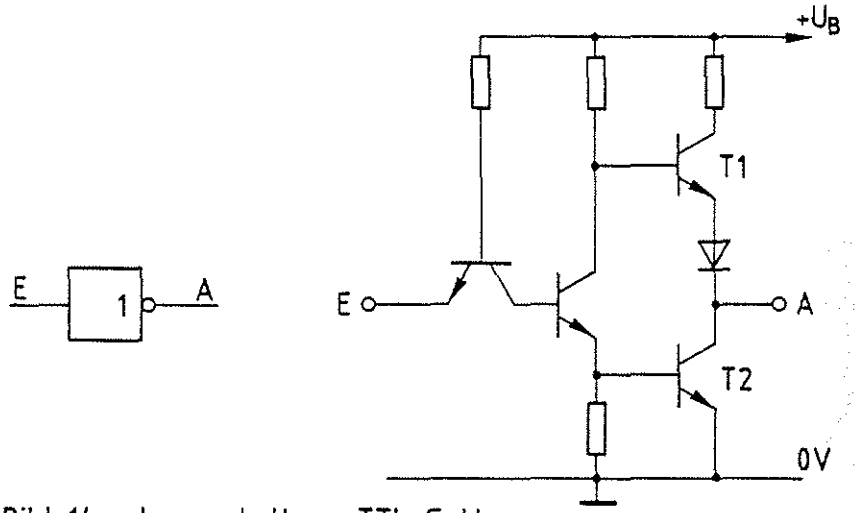


Bild 14: Innenschaltung TTL-Gatter

Die Ausgangsstufe der TTL-Gatter besteht üblicherweise aus einer sogenannten Gegentaktschaltung (totem pole). Je nachdem, ob der Ausgang H- oder L-Pegel führt, wird einer der beiden Transistoren T1 und T2, die wie Schalter wirken, gesperrt (Bild 15).

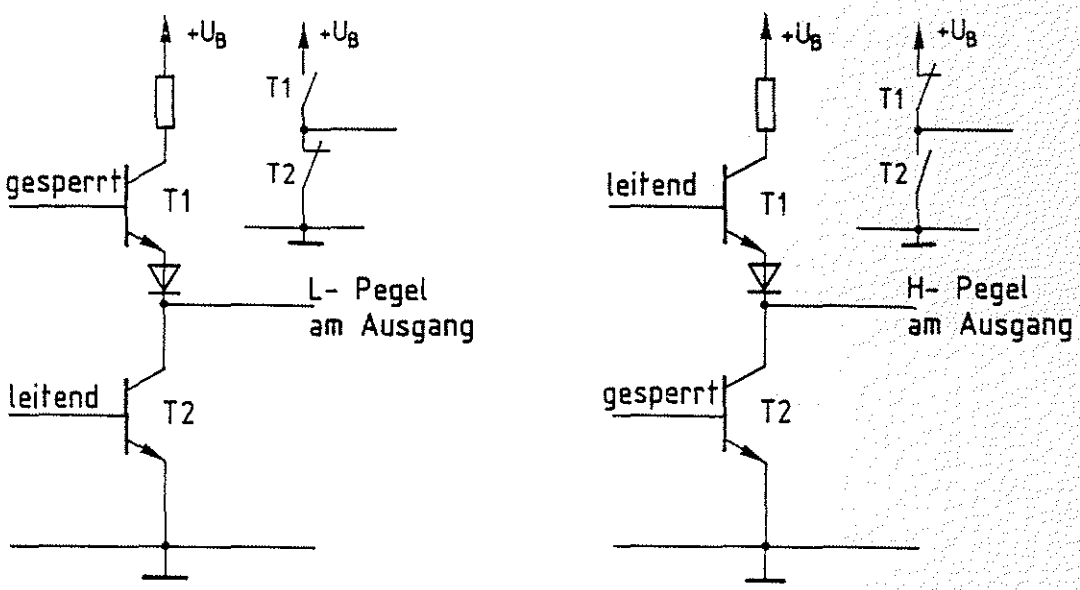


Bild 15: Zustand der Ausgangsstufe bei L- und H-Pegel

Diese TTL-Gatter sind für Baugruppen, die als Sender auf den Daten-Bus wirken, nicht geeignet.

Betrachten Sie dazu das Bild 16, in dem ein Daten-Bus dargestellt ist. An diesem Daten-Bus sind zwei Geräte angeschlossen, die jeweils bei Aufforderung durch den Prozessor Daten auf den Bus schalten können.

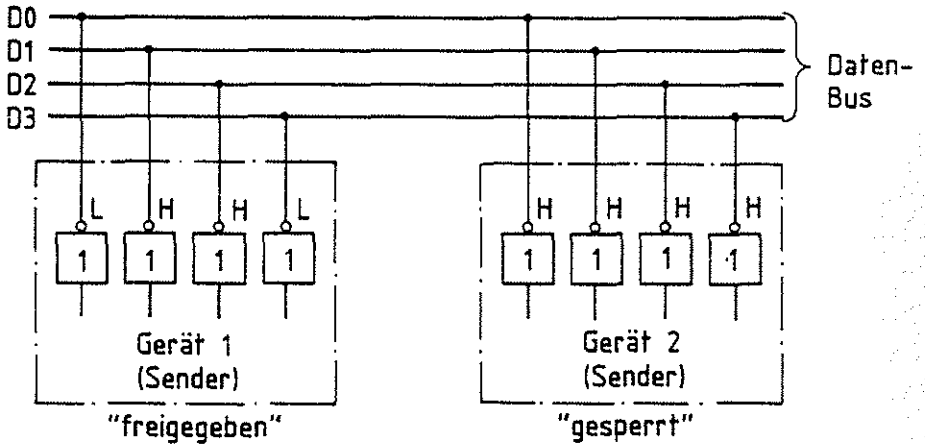


Bild 16: TTL-Gatter am Bus

Als Beispiel sei angenommen, daß das Gerät 1 gerade zum Senden seiner Daten aufgefordert wird und den Signalzustand 0110 auf den Daten-Bus geschaltet hat. Da das Gerät 2 gesperrt ist, führen die zugehörigen Ausgänge H-Signal. Wenn wir nun die Datenleitung D0 getrennt betrachten und etwas verändert herauszeichnen (Bild 17), so erkennt man, daß es an den Ausgängen der angeschlossenen Gatter zu einem Kurzschluß kommt, der die Transistoren zerstören kann.

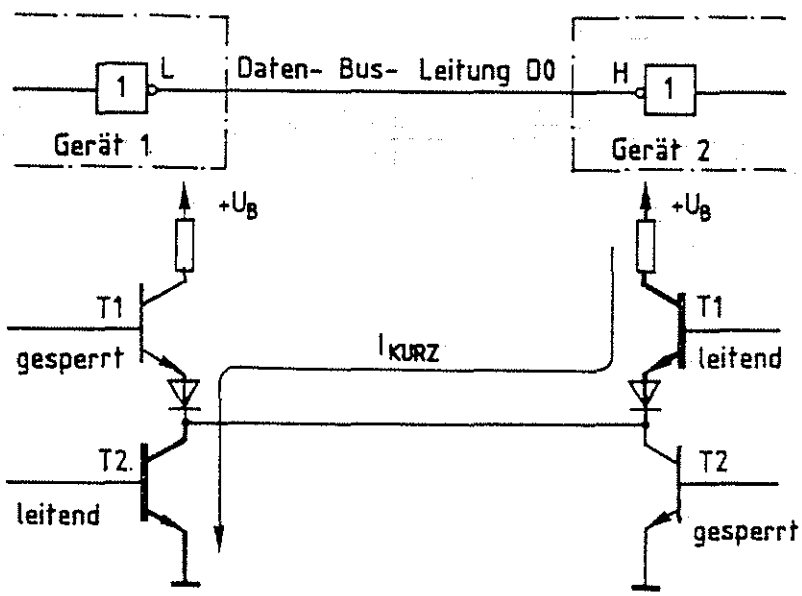


Bild 17: Kurzschlußstrom zwischen TTL-Gattern

Tristate-Technik

Um dies zu vermeiden, hat man die sogenannte Tristate-Technik (Dreistufen-Technik) entwickelt. Tristate-Gatter besitzen einen besonderen Eingang, über den der Gatter-Ausgang vollkommen gesperrt werden kann, so daß der Ausgang weder H- noch L-Pegel führt (offener Ausgang). Im Prinzip kann man das mit Hilfe eines Schalters erreichen, über den der Gatter-Ausgang intern vom IC-Anschluß getrennt wird (Bild 18). Führt der Freigabe-Eingang H-Signal, so ist der Schalter geschlossen; bei L-Signal ist er geöffnet.

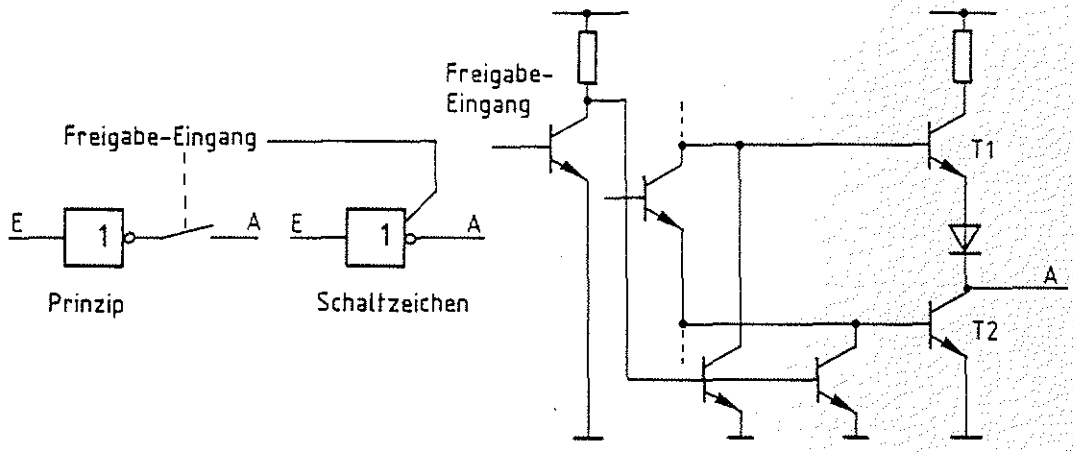
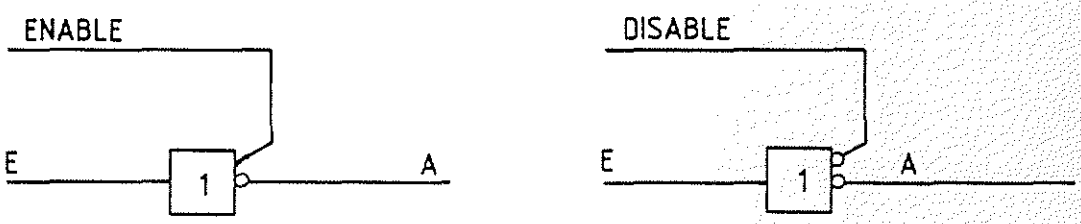


Bild 18: Tristate - Gatter, Prinzip, Schaltzeichen u. Innenschaltung

ENABLE
DISABLE

Praktisch wird dieser dritte Zustand (neben H- und L-Pegel) am Ausgang dadurch erreicht, daß die beiden Transistoren der Gegentaktstufe gesperrt werden. Der Freigabe-Eingang wird meist mit "ENABLE" (ermöglichen, d.h. hier "verbunden") bzw. mit "DISABLE" (unmöglich machen, d.h. hier "nicht verbunden") bezeichnet, je nachdem, ob das H-Signal am Freigabe-Eingang das Gatter freigibt oder sperrt (Bild 19).



ENABLE	AUSGANG
L	gesperrt
H	L oder H

DISABLE	AUSGANG
L	L oder H
H	gesperrt

Bild 19: Tristate-Gatter , Wahrheitstabelle

Alle Daten-Bus-Leitungen derjenigen Baugruppen, die Daten auf den Bus schalten können (Prozessor, Speicher, Eingabe), werden über Tristate-Gatter an den Bus angekoppelt. Da der Mikroprozessor immer nur eine Baugruppe freigeben wird, ist auch immer nur ein Sender am Daten-Bus wirksam. Baugruppen, die nur Daten empfangen, brauchen dagegen nicht vom Daten-Bus getrennt zu werden.

2.2. Kurzschlüsse auf Bus-Leitungen

In ähnlicher Weise wie oben beschrieben, können die Gatterausgänge zerstört werden, wenn Kurzschlüsse zwischen benachbarten Bus-Leitungen vorliegen und die kurzgeschlossenen Leitungen unterschiedliche Signalpegel führen (Bild 20).

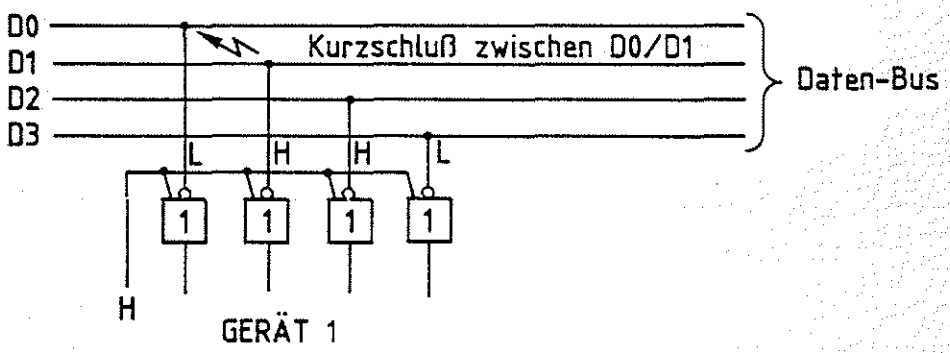


Bild 20: Kurzschluß zwischen Bus-Leitungen

Mit solchen Kurzschlußfehlern muß insbesondere bei der Inbetriebnahme von Bus-Systemen gerechnet werden. Um auch in diesen Fällen eine Zerstörung der Gatter zu verhindern, setzt man in Prüfbaugruppen Schaltkreise mit "Open-Collector-Ausgängen" (offener Kollektor) ein. Bei diesen Gattern fehlt der obere Transistor der Gegentaktstufe am Ausgang, so daß sie für den Betrieb noch einen externen Arbeitswiderstand benötigen (Bild 21).

Open-Collector, OC-Technik

Diesen Widerstand nennt man auch "Pull-Up-Widerstand" (pull up = hochziehen, hier nach U_B).

Pull-Up-Widerstand

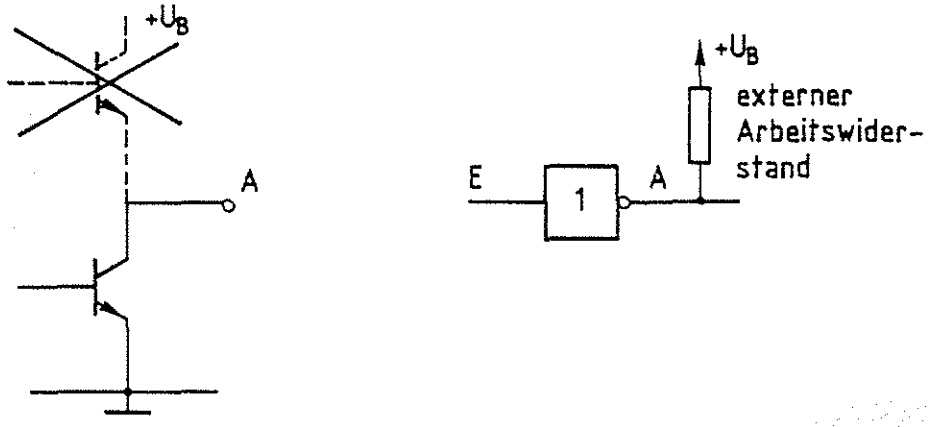


Bild 21: Gatterausgang mit "Pull-Up-Widerstand"

Bei Verwendung dieser Gatter fließt zwar der doppelte Strom durch den leitenden Transistor, sofern der oben geschilderte Kurzschlußfall auftritt, der aber bei geeigneter Dimensionierung der Pull-up-Widerstände nicht zur Zerstörung der Transistoren führt (Bild 22).

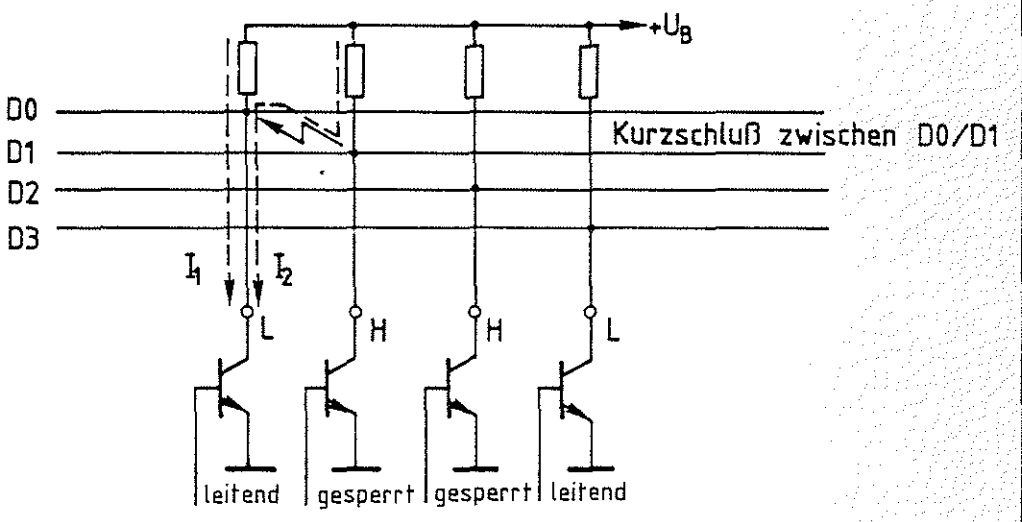


Bild 22: Kurzschluß zwischen Bus-Leitungen bei Gattern mit O.C.-Technik

Alle Adreß-, Daten- und Steuer-Signale des Bus-Signalgebers werden über Open-Collector-Treiber an das System geschaltet, da der Bus-Signalgeber in Verbindung mit der Bus-Signalanzeige auch für die Inbetriebnahme der Baugruppen und des Gesamtsystems verwendet wird.

2.3. Unterbrechungen auf Bus-Leitungen

Leitungsunterbrechungen in digitalen Schaltungen, die aus TTL-Schaltkreisen aufgebaut sind, können nicht zur Zerstörung der Bausteine führen. Allerdings kommt es zu einer fehlerhaften Signalübertragung bzw. -Verarbeitung. Nicht beschaltete Eingänge von TTL-Schaltkreisen wirken so, als läge H-Pegel am Eingang an. Dagegen kann es bei CMOS-Schaltkreisen durch unbeschaltete Eingänge aufgrund einer dann auftretenden "Schwingneigung" zu einer Überhitzung und damit zur Zerstörung der Bausteine kommen. Dieses Verhalten hängt mit den hochohmigen Gate-Eingängen der MOS-Transistoren zusammen.

CMOS

2.4. Signal- Zeit-Diagramme für Bus-Systeme

In ähnlicher Weise, wie man die Bus-Verbindungen in Blockschaltbildern als breiten Verbindungsbalken darstellt, vereinfacht man die graphische Darstellung des Signalflusses auf den Bus-Leitungen durch schematisierte Liniendiagramme. Beispielsweise möge sich der Signalzustand auf den einzelnen Daten-Bus-Leitungen eines Mikrocomputersystems wie in Bild 23 dargestellt, ändern. In der Darstellung ist für jede einzelne Bus-Leitung ein Liniendiagramm gezeichnet.

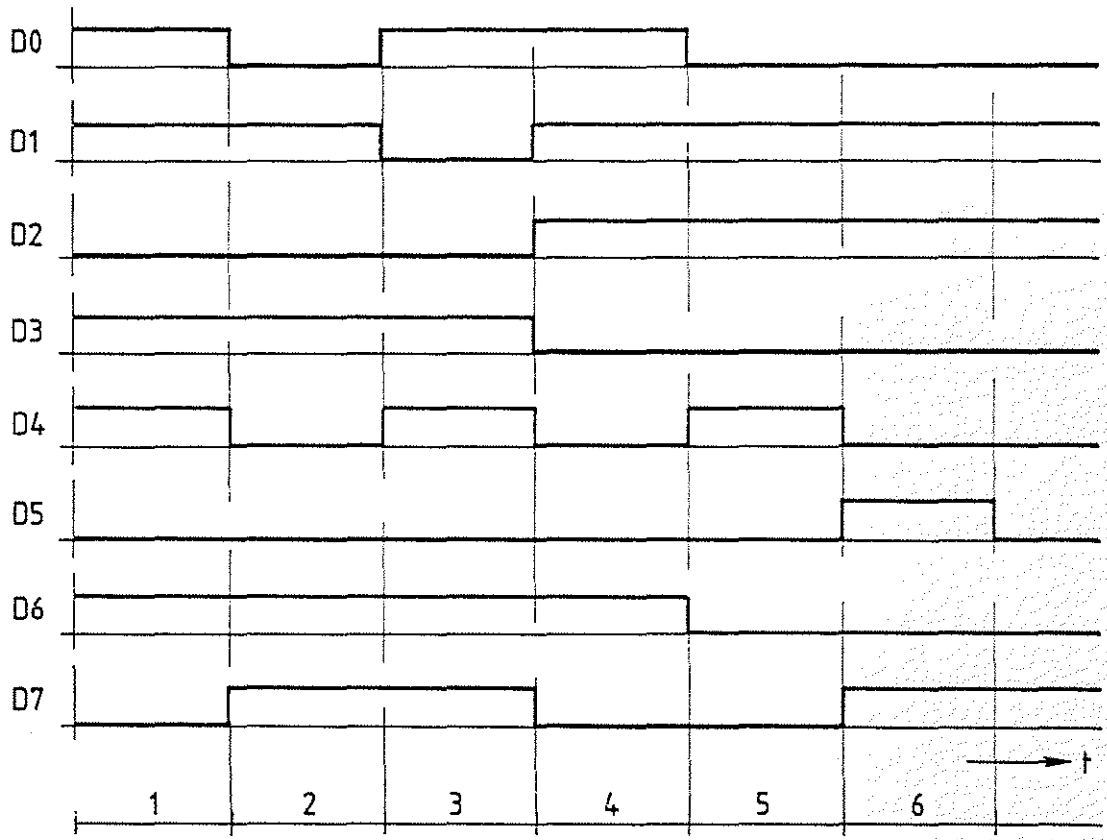


Bild 23: Zeitlicher Verlauf der Signale auf dem Daten-Bus

Diese Art der Darstellung ist sehr unübersichtlich und wird noch schwieriger zu überschauen, wenn beispielsweise noch Adreß- und Steuer-Signale berücksichtigt werden müssen. Statt dessen beschreibt man den Bus-Signalzustand mit einem Liniendiagramm, welches die gleiche Aussagekraft besitzt, wie die Liniendiagramme der einzelnen Bus-Leitungen zusammen.

Ein solches Liniendiagramm ist für das Beispiel aus Bild 23 in Bild 24 dargestellt.

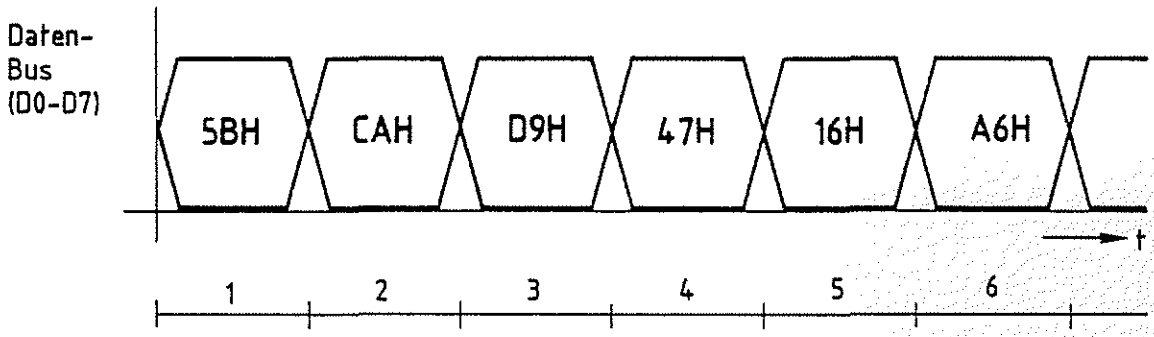


Bild 24: Vereinfachte Darstellung von Bus-Signalen im Liniendiagramm

Dieses Liniendiagramm kennzeichnet den Signalzustand aller acht Bus-Leitungen zusammen, weil der Signalzustand der Einzelleitungen hexadezimal verschlüsselt in das Diagramm eingezeichnet ist. Die Zeitpunkte, zu denen sich neue Signalzustände einstellen, werden schematisch durch H-L- und L-H-Übergänge gekennzeichnet. Damit soll angedeutet werden, daß auf einigen Leitungen der Pegel von High nach Low und auf anderen von Low nach High wechselt. Auf welchen Leitungen ein Wechsel stattgefunden hat, muß aus dem hexadezimal verschlüsselten Zustand abgeleitet werden. Betrachten Sie z.B. den Zeitabschnitt 3. Auf dem Daten-Bus soll der Zustand D9H (H für hexadezimal) vorhanden sein. Mit Hilfe der Hex-Tabelle Bild 11 erhält man für 9H den Zustand 1001 und für DH 1101, so daß die Leitungen die folgenden Pegel führen: D7 = H, D6 = H, D5 = L, D4 = H, D3 = H, D2 = L, D1 = L und D0 = H. Vergleichen Sie diese Pegel mit den Liniendiagrammen in Bild 23.

Am Daten-Bus kann der Zustand auftreten, daß alle Sender gesperrt sind. Aufgrund der Tristate-Technik sind dann alle Bus-Leitungen offen und führen weder H- noch L-Pegel. Dieser Zustand wird häufig wie in Bild 25 dargestellt gekennzeichnet:

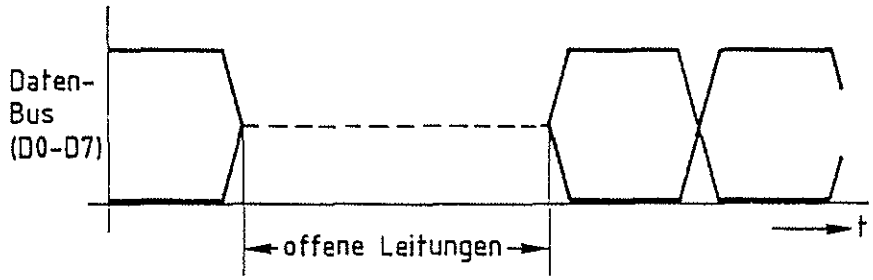


Bild 25: Darstellung des hochohmigen Zustandes von Bus-Leitungen

Soll zusätzlich im Diagramm noch zum Ausdruck gebracht werden, welche Signale andere Signalwechsel zur Folge haben, so trägt man Wirkungspfeile in die Diagramme ein, die Ursache und Wirkung erkennen lassen. Das gewählte Beispiel in Bild 26 verdeutlicht, daß das Steuersignal vom Prozessor für "Speicher lesen" ($\overline{\text{MEMR}}$) zur Folge hat, daß der Speicher seine Daten auf den Bus schaltet.

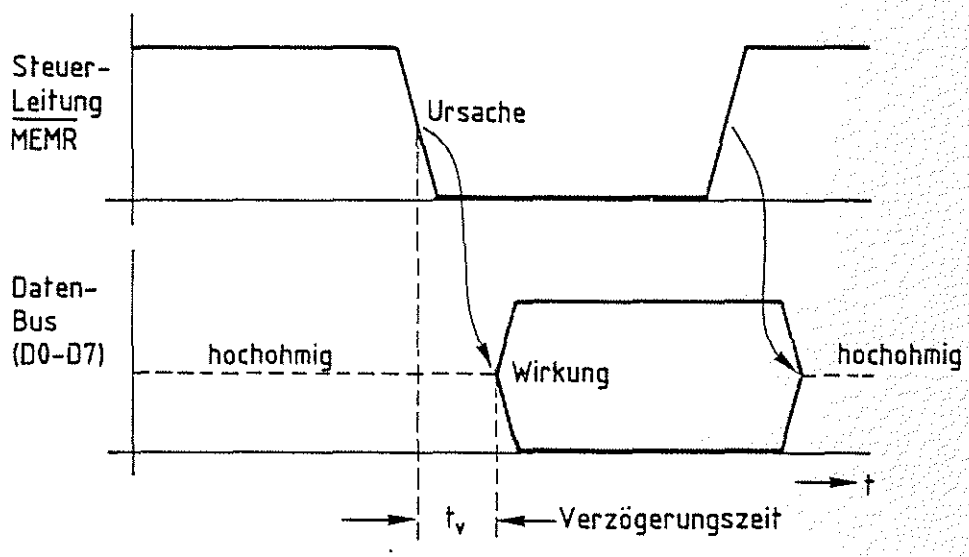


Bild 26: Darstellung von Ursache und Wirkung in Signal-Zeit-Diagrammen

Da die Signale von den Baugruppen nicht unendlich schnell verarbeitet werden, ergeben sich meist kurze Verzögerungszeiten, wie im Bild 26 dargestellt. Der Speicher benötigt nach der Aufforderung, seine Daten auf den Bus zu schalten, eine kurze Verzögerungszeit, um den Schaltvorgang durchzuführen (bei üblichen Speichern zwischen 50...500nsec). Dies wird durch den Versatz zwischen Ursache und Wirkung gekennzeichnet.

Mit Hilfe dieser Darstellungsart ist in Bild 27 der zeitliche Signalablauf auf dem System-Bus beim Lesen der Speicherstelle 1F03H durch den Mikroprozessor dargestellt. Der Speicher soll dabei die Daten 11001010 bereitstellen.

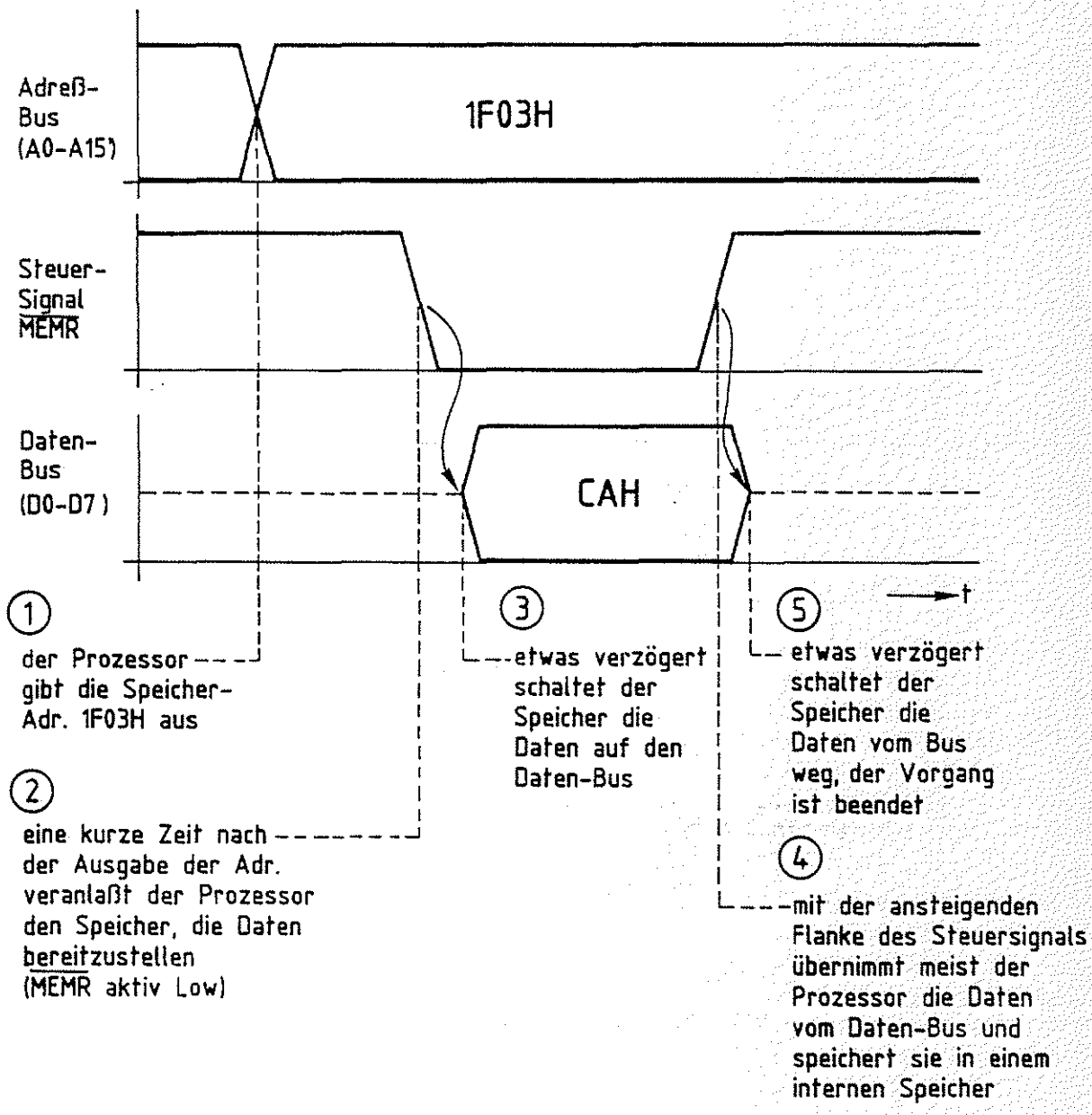


Bild 27: Signal-Zeit-Diagramm für das Lesen einer Speicherstelle durch den Prozessor

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

AUFBAU VON DV-ANLAGEN UND BUS-SYSTEMEN

BFZ/MFA 10.1.

ÜBUNGSTEIL 2

Übungsteil 2

In den folgenden Arbeitsschritten werden Sie Messungen am Bus-System eines Mikrocomputers durchführen.

Dazu benötigen Sie:

- 1 Baugruppenträger mit Busverdrahtung (BFZ/MFA 0.1.)
- 1 Bus-Abschluß (BFZ/MFA 0.2.)
- 1 Trafo-Einschub (BFZ/MFA 1.1.)
- 1 Spannungsregelung (BFZ/MFA 1.2.)

} zusammengebaut und geprüft nach FPO BFZ/MFA 1.2. Arbeitsblatt A7

- 1 Bus-Signalgeber (BFZ/MFA 5.1.)
- 1 Bus-Signalanzeige (BFZ/MFA 5.2.)
- 1 Adapterkarte 64polig (BFZ/MFA 5.3.)
- 1 Logik-Tester oder Vielfach-Meßinstrument
- 4 Meßleitungen und Meßklips

Allgemeine Hinweise zur Durchführung der Übungen:

- Die Einschübe dürfen nur bei abgeschalteter Betriebsspannung gesteckt oder gezogen werden
- Aufgrund der Busverdrahtung können die Baugruppen in beliebige Steckplätze gesteckt werden
- Messungen an den Bus-Leitungen sollten mit Hilfe der Adapterkarte durchgeführt werden
- Den logischen Signalen "0" und "1" sind die folgenden Pegel zugeordnet:

  log. "0" ≙ 0...0,8 V (LOW)

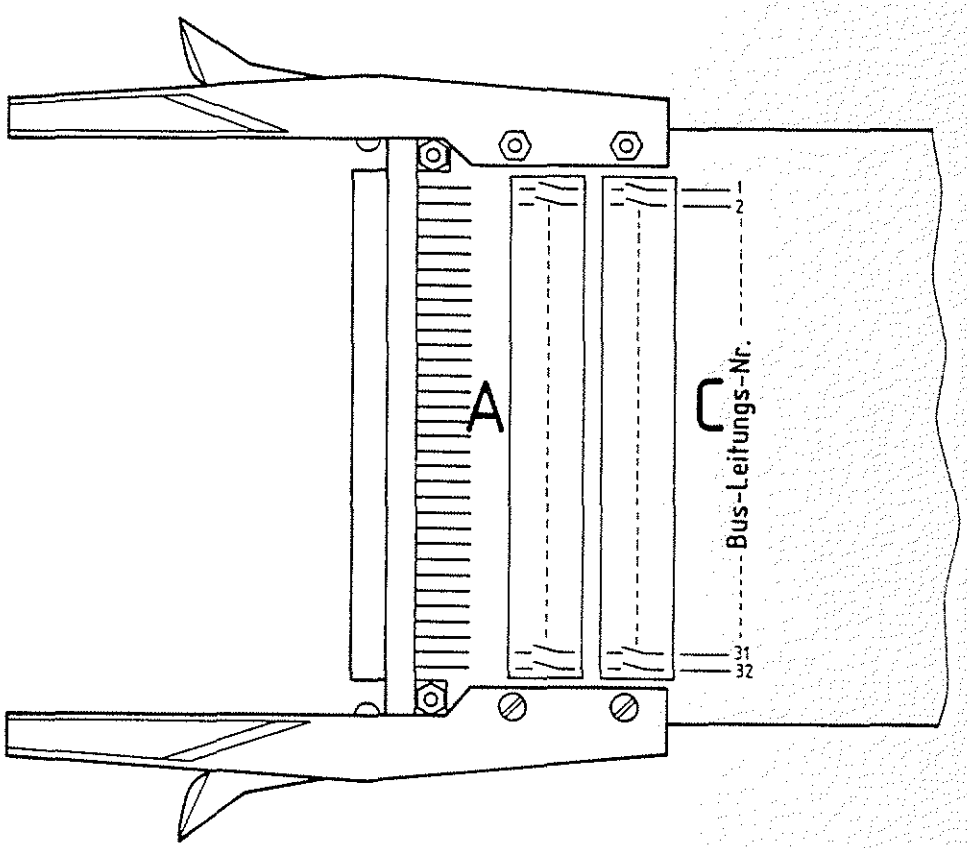
  log. "1" ≙ 2,4...5 V (HIGH)

- Alle zur Messung an den Baugruppen vorgegebenen Arbeitsblätter enthalten:
  - = Angaben über den Sinn der jeweiligen Messung
  - = Angaben über einzustellende Bedingungen (z.B. Schalterstellungen)
  - = Aufgabenstellungen, ggf. mit Hinweisen zu möglichen Fehlern.

Bedienungshinweise:

Adapterkarte 64polig

Kurzschlüsse auf Bus-Leitungen können Sie mit Hilfe von Meßleitungen/Meßklips an den Drahtbrücken/Schaltern herstellen, die sich an der Vorderseite der Adapterkarte befinden.



Adapterkarte 64polig

Zum Herstellen von Unterbrechungen wird diejenige Baugruppe, für die die Unterbrechungen wirksam werden sollen, über die Adapterkarte mit dem System-Bus verbunden. Dazu müssen vorher die entsprechenden Drahtbrücken oder Schalter geöffnet werden.

Aufbau von DV-Anlagen und Bus-Systemen | Name:

Übungsteil 2 | Datum:

A1

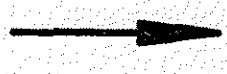
Messungen am Daten-Bus bei einer unterbrochenen Datenleitung

Stecken Sie die Bus-Signalanzeige direkt und den Bus-Signalgeber über die Adapterkarte in den Baugruppenträger. Unterbrechen Sie die Datenleitung D2.
Schalten Sie die Betriebsspannung ein.

Ermitteln Sie die von der Bus-Signalanzeige angezeigten Daten, wenn Sie nacheinander alle möglichen sechzehn Signalzustände an den unteren vier Daten-Leitungen einstellen. Überprüfen Sie auch mit Hilfe eines Logiktesters/Meßgeräts den Zustand der offenen Datenleitung an der Bus-Signalanzeige.

eingestellte Daten	angezeigte Daten
00	
01	
02	
03	
04	
05	
06	
07	
08	
09	
0A	
0B	
0C	
0D	
0E	
0F	

Begründen Sie, warum nicht alle Daten falsch angezeigt werden.



Aufbau von DV-Anlagen und Bus-Systemen Name:

Übungsteil 2 Datum:

A2

Messungen am Daten-Bus bei einem Kurzschluß zwischen zwei Datenleitungen

Stellen Sie die Verbindung der Datenleitung D2 wieder her und schließen Sie die Leitungen D2 und D3 kurz.

Ermitteln Sie zu den eingestellten Daten die vom Bus-Signalgeber angezeigten.

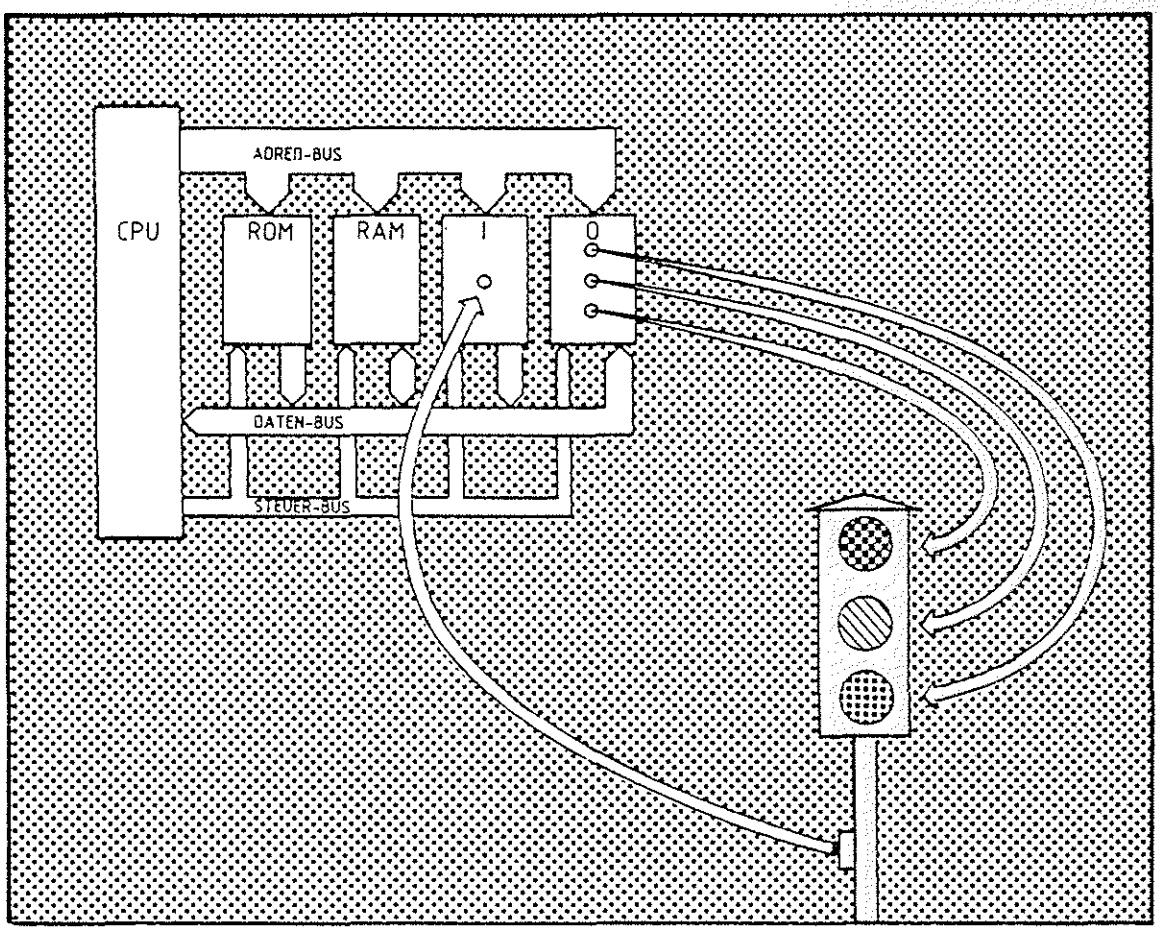
eingestellte Daten	angezeigte Daten
00	
01	
02	
03	
04	
05	
06	
07	
08	
09	
0A	
0B	
0C	
0D	
0E	
0F	

Begründen Sie, warum nicht alle Daten falsch angezeigt werden.

Ende der Übung!

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

EIN-UND AUSGABE-EINHEITEN BFZ/MFA 10.2.



BFZ MFA

Diese Übung ist Bestandteil eines Mediensystems, das im Rahmen eines vom Bundesminister für Bildung und Wissenschaft, vom Bundesminister für Forschung und Technologie sowie der Bundesanstalt für Arbeit geförderten Modellversuches zum Einsatz der "Mikrocomputer-Technik in der Facharbeiterausbildung" vom BFZ-Essen e.V. entwickelt wurde.

Inhaltsverzeichnis

Theorieteil 1

1.1. Einleitung

1.2. Ein-Ausgabe-Techniken
1.2.1. Die parallele Datenübertragung
1.2.2. Ein- und Ausgabe von analogen Signalen
1.2.3. Serielle Ein- und Ausgabe-Einheiten

1.3. Der Aufbau einer parallelen Ausgabe-Baugruppe

1.4. Der Aufbau einer parallelen Eingabe-Baugruppe

Übungsteil 1

A1 Übergabe von Daten an die Ausgabe-Baugruppe

A2 Prüfen der RESET-Funktion

A3 Lesen von Daten von der Eingabe-Baugruppe

A4 Lesen der Daten der Eingabe-Baugruppe und Ausgabe dieser Daten an die Ausgabe-Baugruppe

Theorieteil 2

2.1. Vereinfachte Schaltung der Ausgabe-Baugruppe

2.2. Vereinfachte Schaltung der Eingabe-Baugruppe

Übungsteil 2

A1 Überprüfen des Adreßvergleichers der Ausgabe-Baugruppe

A2 Überprüfen des Adreßvergleichers der Eingabe-Baugruppe

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

EIN-UND AUSGABE-EINHEITEN BFZ/MFA 10.2.

THEORIETEIL 1

1.1. Einleitung

Jeder (Mikro-)Computer steht über Ein- und Ausgabe-Leitungen mit anderen Baugruppen, Geräten oder Anlagen für den Austausch von Daten und Informationen in Verbindung. Betrachten wir ein typisches Problem für den Einsatz eines Mikrocomputers Bild 1. An verkehrsreichen Straßen sind für die Fußgänger Ampelanlagen eingerichtet, damit sie die Straßen gefahrlos überqueren können. Den Fußgängern wird

Ampelanlage

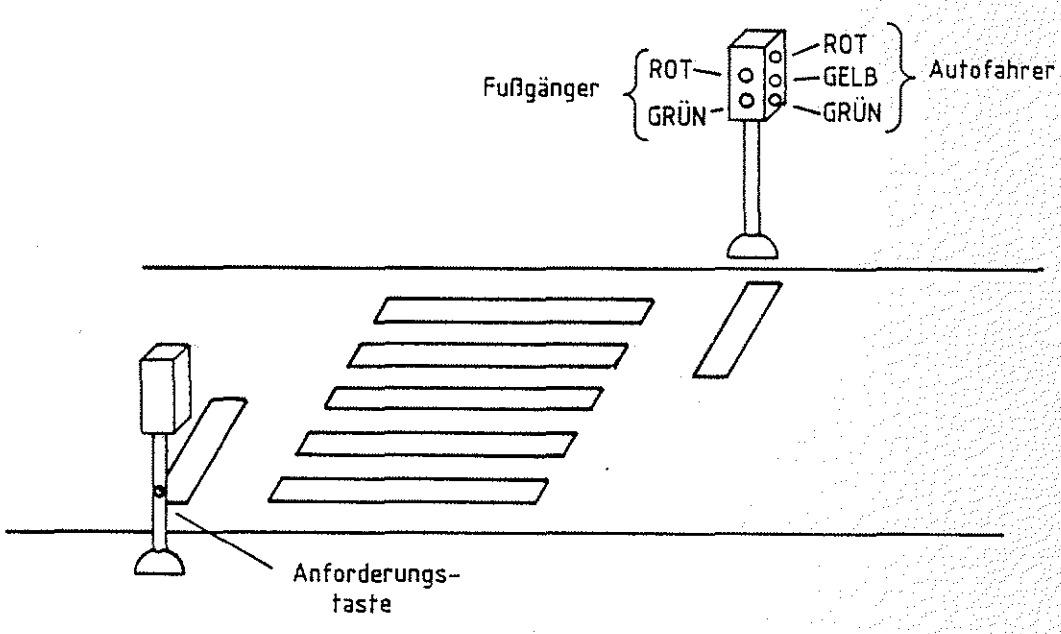


Bild 1: Einfache Ampelanlage

durch ROT- und GRÜN-Licht signalisiert, daß sie entweder WARTEN müssen, oder die Straße ÜBERQUEREN dürfen. Für die Autofahrer sind die üblichen drei Lichtsignale ROT, GELB und GRÜN vorhanden. Die Autofahrer haben solange freie Fahrt (Grünphase), bis ein Fußgänger durch Betätigen der Anforderungstaste den Wunsch zum Überqueren der Straße signalisiert. Die Folge der Lichtsignale, die daraufhin dem Fußgänger und den Autofahrern angezeigt wird, ist in Bild 2 zusammengefaßt.

Phasen Nr.	Fußgänger		Autofahrer			Bemerkung
	Grün	Rot	Grün	Gelb	Rot	
0	AUS	EIN	EIN	AUS	AUS	Grünphase für Autofahrer
1	AUS	EIN	AUS	EIN	AUS	Gelbphase für Autofahrer
2	EIN	AUS	AUS	AUS	EIN	Grünphase für Fußgänger
3	AUS	EIN	AUS	EIN	EIN	Rot-Gelb-Phase für Autofahrer
4	AUS	EIN	EIN	AUS	AUS	Grünphase für Autofahrer

Bild 2: Signalkombinationen bei verschiedenen Ampelphasen

Ampelphasen

Die Phase 0 ist der Ausgangszustand, bei dem ein Fußgänger die Anforderungstaste betätigt. Nach einer kurzen Verzögerungszeit zeigt die Ampel die Gelbphase für die Autofahrer (Phase 1) an. Nach einer erneuten Verzögerungszeit erfolgt die Weiterschaltung zur Rotphase (Phase 2) für die Autofahrer und zur Grünphase für die Fußgänger und so fort. Jeder der Zustände bleibt für eine bestimmte Zeit bestehen. Mit dem Übergang zur Phase 4 ist der Ausgangszustand wieder erreicht und ein vollständiger Zyklus abgelaufen. Der Vorgang kann durch Betätigen der Anforderungstaste erneut ausgelöst werden. Soll dieser Prozeß von einem Mikrocomputer gesteuert werden, so müssen Ausgangsleitungen für die Signallampen und Eingangsleitungen für Anforderungstaster vorhanden sein. Ein- und Ausgangsleitungen stellen die Verbindung eines Computers mit seinem Umfeld (Peripherie, hier die Fußgängerampel) her. Zum Anschluß dieser Leitungen besitzt der Computer Ein- und Ausgabe-Einheiten.

Peripherie

E/A-Einheiten

1.2. Ein-Ausgabe-Techniken

1.2.1. Die parallele Datenübertragung

Bild 3 zeigt schematisch den Anschluß einer Ein- und einer Ausgabe-Einheit an den System-Bus des Mikrocomputers.

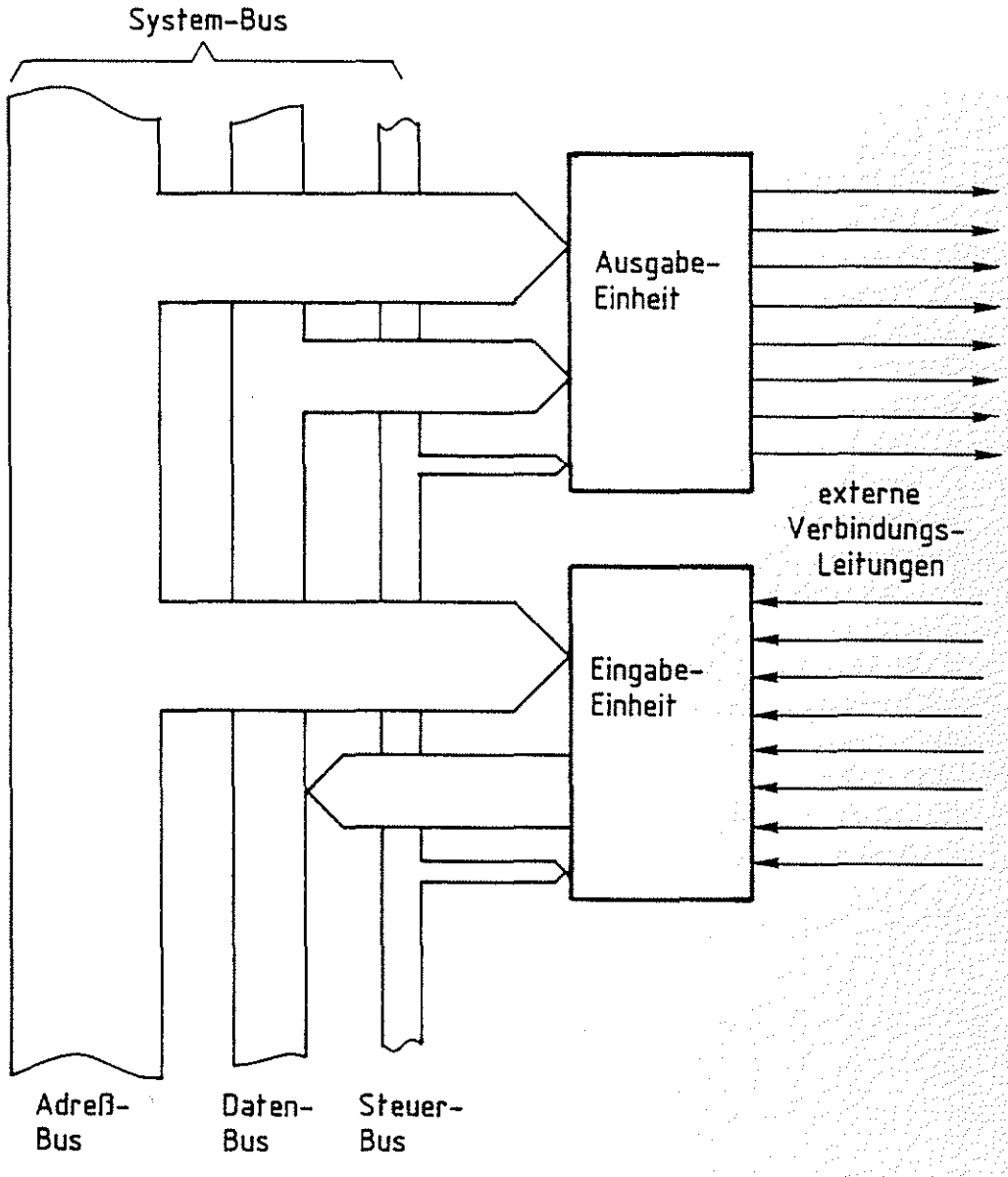


Bild 3: Anschluß von Ein- und Ausgabe-Einheit an den System-Bus

Die Anzahl der für die Ein- und Ausgabe von Informationen verfügbaren Leitungen ist an die Daten-Bus-Struktur des Mikrocomputers angepaßt. Besitzt der Daten-Bus acht Leitungen, so stellt eine Ausgabe- oder eine Eingabe-Einheit ebenfalls acht Verbindungsleitungen zur Außenwelt des Computers bereit.

Auf diesen acht Leitungen werden 8-Bit-Datensignale übertragen. In der Computer-Technik faßt man 8 Bit zu einem Byte zusammen. Man spricht deshalb bei 8-Bit-Daten auch von Daten-Bytes. Bild 4 veranschaulicht diese Begriffe noch einmal.

Byte

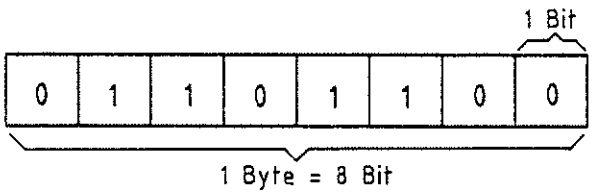


Bild 4: Der Zusammenhang von Bits und Byte

Die gleichzeitige Bereitstellung bzw. Übertragung mehrerer Signale nennt man "Parallele Datenübertragung". Die in Bild 3 dargestellten Ein- und Ausgabe-Einheiten nennt man daher auch parallele E/A-Einheiten. Im Gegensatz dazu gibt es auch serielle Ein- und Ausgabe-Einheiten. Ihr Funktionsprinzip wird später erklärt.

Parallele Datenübertragung

Kehren wir nun zu unserem Problem "Fußgängerampel" zurück. Der Mikrocomputer muß dafür entsprechende Aus- und Eingangsleitungen bereitstellen (Bild 5). Die Signallampen der Ampel werden über Leistungsverstärker an die Ausgangsleitungen der parallelen Ausgabe-Einheit angeschlossen und die Anforderungstaster verbindet man eventuell über Pegelanpassungen oder Entprellschaltungen mit den Eingangsleitungen der Eingabe-Einheit. Als weitere Eingangssignale können z.B. in Betracht kommen: Störungsmeldungen, die den Ausfall von Signallampen anzeigen oder auch Tag- Nachtumschaltung, die in den Nachtstunden das gelbe Blinklicht einschaltet.

Leistungsverstärker

Pegelanpassung

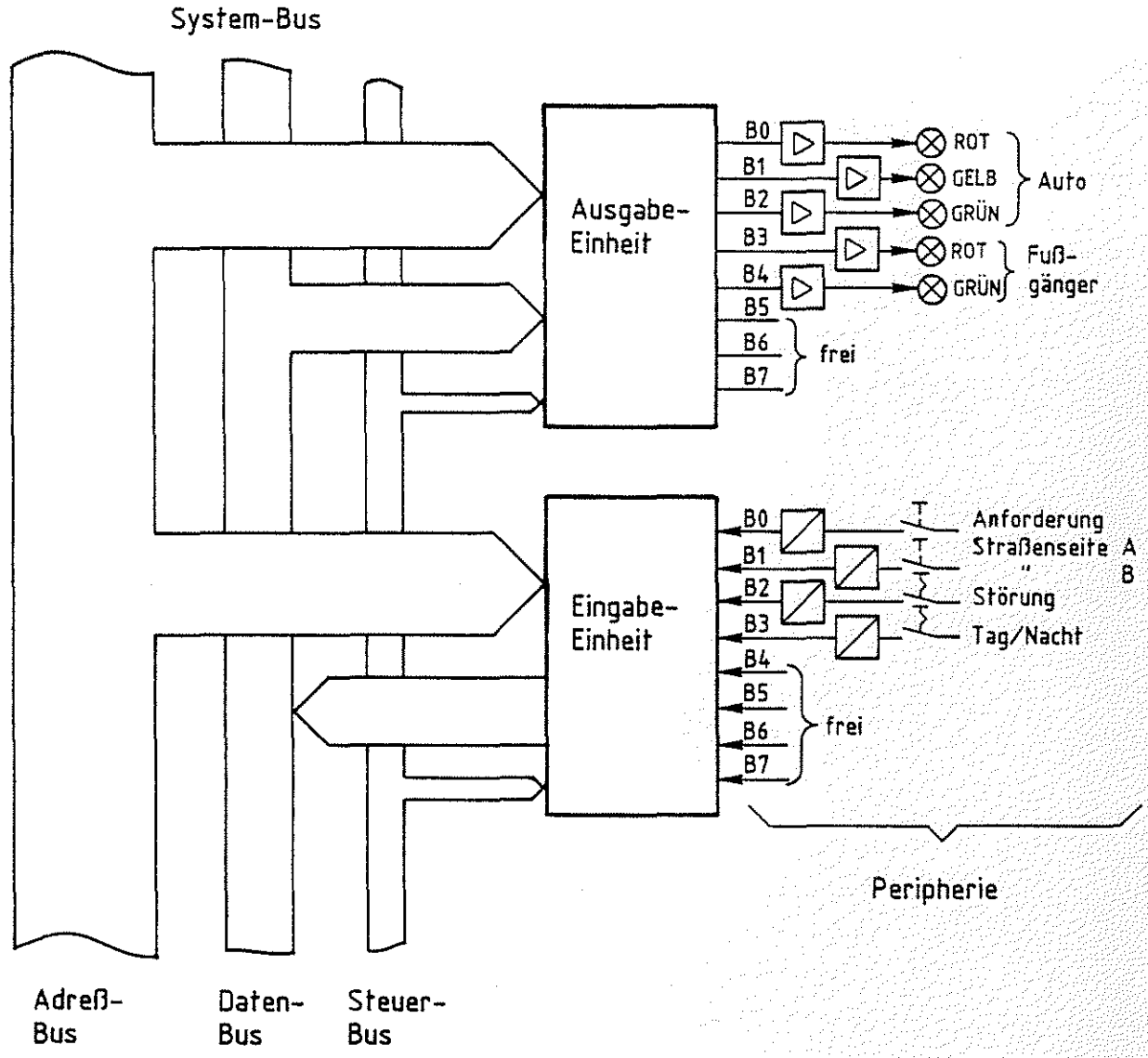


Bild 5: Anschluß der Peripherie am Mikrocomputer

Die Tabelle in Bild 6 zeigt die Daten-Bytes, die für die verschiedenen Ampelphasen an die Ausgabe-Einheit übergeben werden müssen.

Phasen Nr.	Daten-Bytes								
	binär								hex.
	B7	B6	B5	B4	B3	B2	B1	B0	
0	X	X	X	0	1	1	0	0	0C
1	X	X	X	0	1	0	1	0	0A
2	X	X	X	1	0	0	0	1	11
3	X	X	X	0	1	0	1	1	0B
4	X	X	X	0	1	1	0	0	0C
					Fußgänger		Auto		

X bedeutet, daß der Signalzustand beliebig sein kann. Diese Leitungen sind nicht belegt. Zur Bestimmung der Hex-Werte wurde X=0 gesetzt.

Bild 6: An die Ampeln auszugebende Daten-Bytes

Die Probleme, die sich mit dem Anschluß von Baugruppen, Geräten und Anlagen an den Mikrocomputer ergeben, gehören zum Gebiet der "Interface-Technik". Interface kommt aus dem Englischen und bedeutet Grenz- oder Berührungsfläche. Interface-Baugruppen nennt man alle Baugruppen, die die Verbindung zwischen dem eigentlichen Computer und seiner Umwelt herstellen.

Interface-Technik

1.2.2. Ein- und Ausgabe von analogen Signalen

Neben den oben beschriebenen Interface-Baugruppen, die Daten und Signale in digitaler Form bereitstellen oder empfangen, gibt es auch solche, die z.B. analoge Spannungen abgeben oder empfangen können. Dafür verwendet man sogenannte Digital- Analog-Wandler (D/A-Wandler) bzw. Analog- Digital-Wandler (A/D-Wandler). Soll beispielsweise ein Mikrocomputer für ein Meßwertprotokoll einen Spannungswert erfassen, so muß dieser in digitaler Form einer Eingabe-Einheit zugeführt werden. Dazu wird vor die Eingabe-Einheit ein Analog- Digital-Wandler geschaltet (Bild 7).

analoge E/A-Einheiten

A/D- und D/A-Wandler

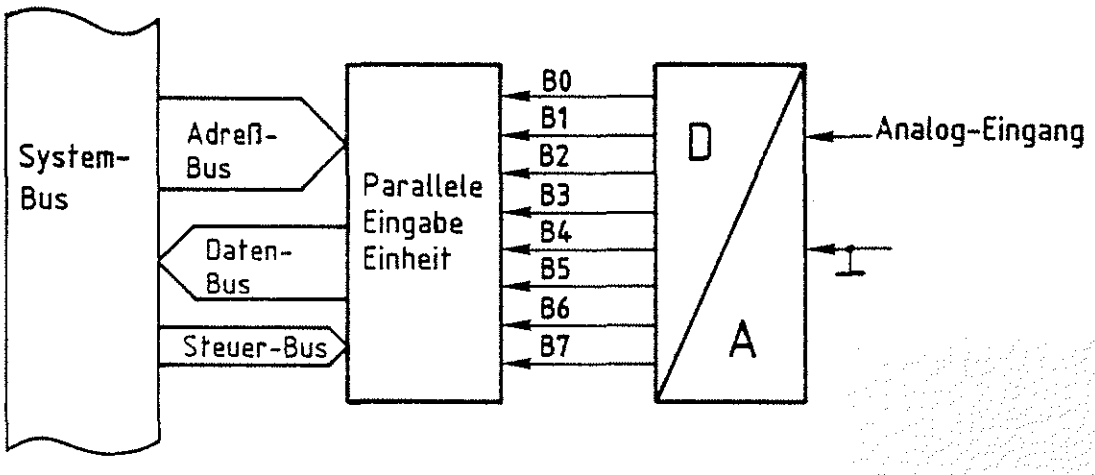


Bild 7: Eingabe von analogen Werten in den Mikrocomputer

Typische A/D-Wandler erzeugen z.B. zu einem Spannungsbereich von 0 V bis 2,55 V 8-Bit-Daten von 00000000 bis 11111111 (hex. 00 bis FF).
Für den Fall der analogen Spannungsausgabe durch einen Mikrocomputer muß einer parallelen Ausgabe-Einheit ein Digital-/Analog-Wandler nachgeschaltet werden.
Die Interface-Technik ist so vielfältig, wie es verschiedene Geräte und Anlagen gibt, die durch Mikrocomputer gesteuert werden; sie stellt ein umfangreiches Feld dar, auf das hier nicht weiter eingegangen werden soll.

1.2.3. Serielle Ein- und Ausgabe-Einheiten

Eine serielle Ausgabe-Einheit ist eine Baugruppe, die alle vom Daten-Bus empfangenen parallelen Daten in einen seriellen Datenstrom umwandelt, der dann auf einer Leitung übertragen wird. Dabei wird an die Ausgangsleitung der Baugruppe, jeweils eine bestimmte Zeit lang (Taktzeit), zunächst der Signalzustand des ersten Daten-Bits, dann der des zweiten, des dritten usw. angelegt, bis alle acht Daten-Bits jeweils während einer Taktzeit "T" am Ausgang zur Verfügung standen. Bild 8 zeigt das beschriebene Prinzip.

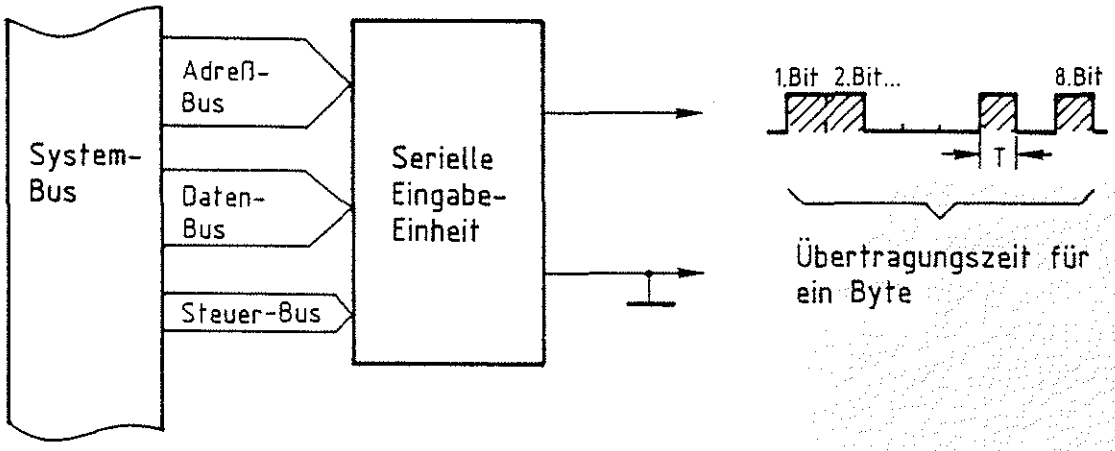


Bild 8: Prinzip der seriellen Datenausgabe

Ein externes Gerät, das den seriellen Datenstrom empfangen soll, muß zur selben Zeit und im gleichen Rhythmus den Signalzustand der Übertragungsleitung abfragen, bis alle Daten-Bits empfangen wurden. Ebenso empfängt eine serielle Eingabe-Einheit einen seriellen Datenstrom und stellt ihn als parallele Daten am Daten-Bus bereit (Bild 9).

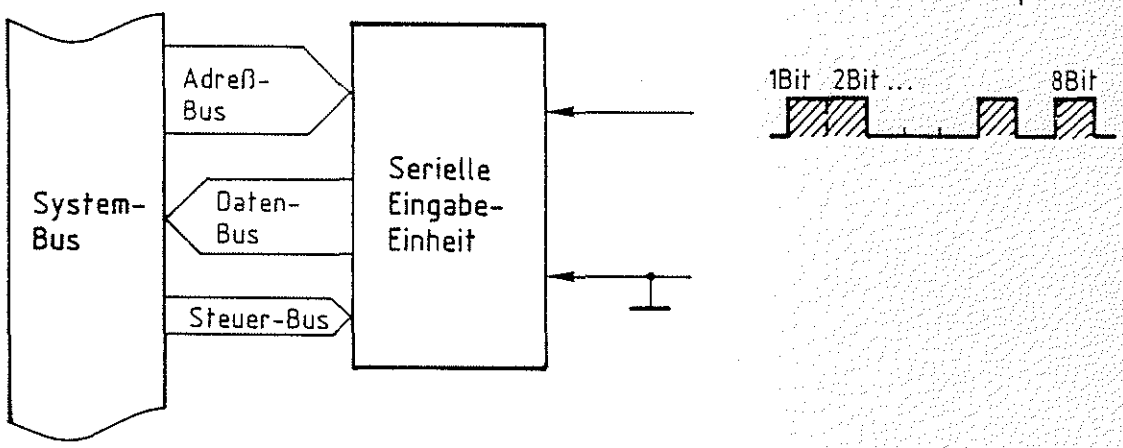


Bild 9: Prinzip des seriellen Datenempfangs

Die Anwendung der parallelen bzw. der seriellen Datenübertragung für den Informationsaustausch zwischen zwei Geräten hängt von den verschiedensten Faktoren ab. Entscheidend für die Anwendung der parallelen Übertragung kann in einem Fall die höhere Übertragungsgeschwindigkeit sein, im anderen die Art der angeschlossenen Geräte. Die serielle Datenübertragung wird man in der Regel immer dann anwenden, wenn größere Entfernungen überbrückt werden müssen.

1.3. Der Aufbau einer parallelen Ausgabe-Baugruppe

Im Bild 10 ist eine parallele Ausgabe-Baugruppe mit ihren Anschlüssen als Block dargestellt. Das wesentliche Merkmal dieser Baugruppe ist es, daß sie die vom Prozessor übernommenen Daten speichert und solange am Ausgang bereitstellt, bis der Prozessor ihr neue Daten übergibt.

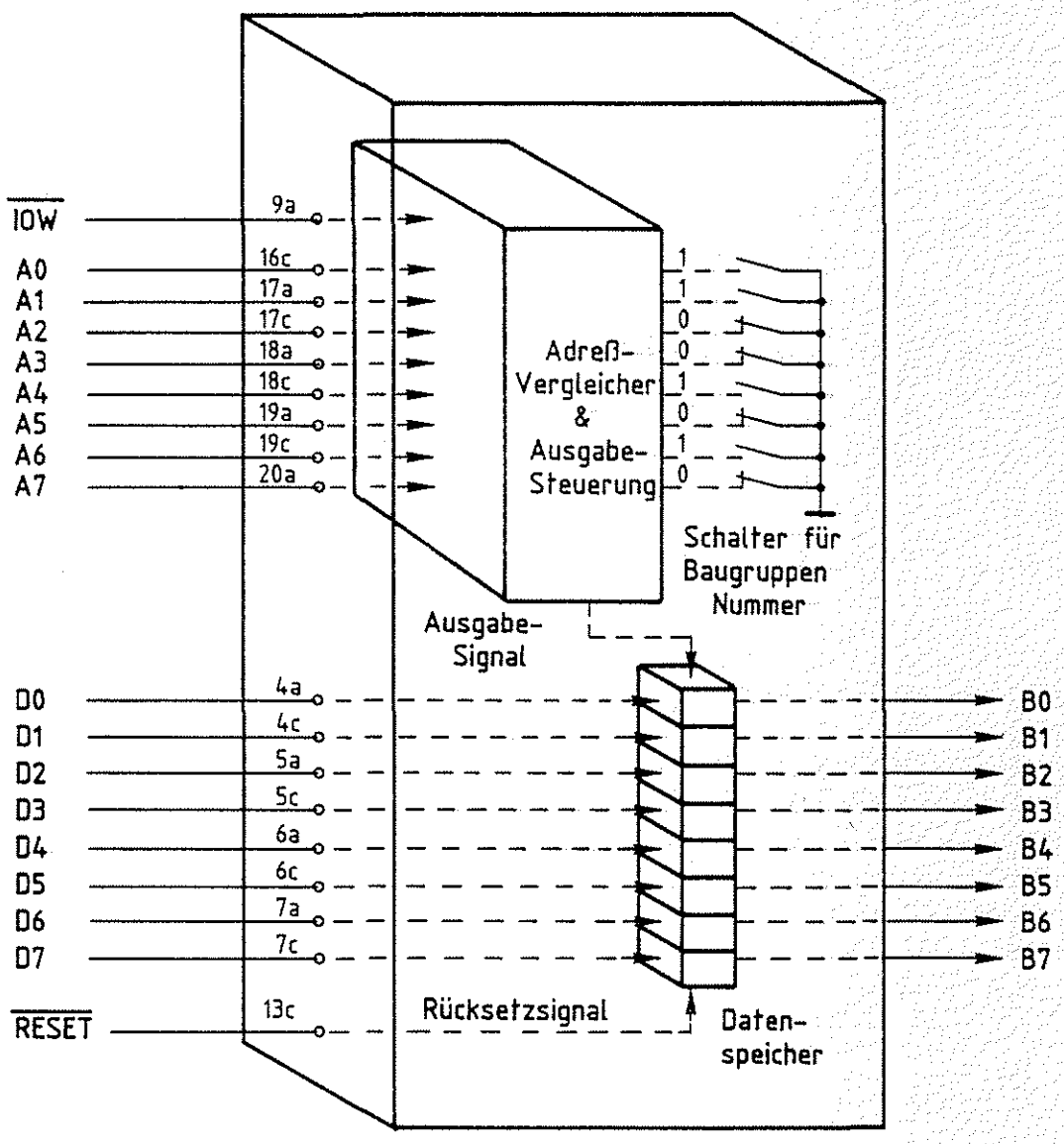


Bild 10: Aufbau einer parallelen Ausgabe-Baugruppe

Die Speicherung der Daten ist notwendig, weil der Prozessor nicht ständig mit der Ausgabe-Baugruppe verbunden ist, während er z.B. neue Anweisungen im Programmspeicher liest oder den Signalzustand an den Eingangsleitungen der Eingabe-Baugruppe abfragt. Die Eingänge der acht Datenspeicher sind direkt mit den Daten-Bus-Leitungen verbunden und die Speicherausgänge werden meist über hier nicht gezeichnete Verstärker nach außen geführt. Jede Ausgabe-Baugruppe besitzt eine bestimmte Adresse oder Baugruppen-Nummer, die oft über Schalter oder Lötbrücken eingestellt werden kann. Zu beachten ist, daß der Prozessor zur Auswahl einer Ein-Ausgabe-Baugruppe nicht alle Adreßleitungen benutzt. Die Nummer derjenigen Ausgabe-Baugruppe, die Daten vom Daten-Bus übernehmen soll, wird vom Prozessor über die unteren acht Adreßleitungen A0 bis A7 ausgesendet. Insgesamt kann der Prozessor 2^8=256 verschiedene Ausgabe-Baugruppen unterscheiden. Die eingestellte Baugruppen-Nummer im Bild 10 ist z.B. 01010011 oder hexadezimal 53. Tritt dieser Signalzustand auf den Adreßleitungen auf, so wird das vom Adreßvergleicher dieser Baugruppe angezeigt. Sendet nun der Prozessor noch das Steuersignal INPUT/OUTPUT WRITE aus, indem er die Leitung $\overline{\text{IOW}}$ auf L-Signal schaltet, so wird die Ausgabesteuerung aktiv und erzeugt ein Signal zur Übernahme der Daten vom Daten-Bus in die Speicher und damit zur Ausgabe an die Ausgangsbuchsen. Üblicherweise erfolgt die Datenübernahme flankengesteuert, d.h. mit einem Signalwechsel von L- nach H-Pegel auf der Steuerleitung $\overline{\text{IOW}}$. Dieser Flankenwechsel löst unmittelbar das Ausgabesignal aus.

Beim Anlegen der Betriebsspannung nehmen die Datenspeicher in zufälliger Weise H- und L-Pegel an. Dieser unkontrollierte Signalzustand kann an den angeschlossenen Maschinen und Anlagen unter Umständen falsche Aktionen auslösen. Um dies zu verhindern, wird mit dem Einschalten der Betriebsspannung ein Rücksetzsignal ($\overline{\text{RESET}}$, aktiv low) erzeugt, welches die Datenspeicher löscht, so daß alle Ausgänge L-Signal führen. Dieses Rücksetzsignal liefert der Prozessor.

Baugruppen-Nummer

Adreßvergleicher

INPUT/OUTPUT

Flankensteuerung

Unkontrollierter Signalzustand

RESET

Zusammengefaßt sind folgende Schritte zur Datenübertragung an eine Ausgabe-Baugruppe erforderlich:

Der Prozessor...

- wählt durch Aussenden einer Nummer (Adresse) auf den unteren acht Adreß-leitungen die gewünschte Ausgabe-Bau-gruppe aus,
- schaltet danach die Ausgabe-Daten auf den Daten-Bus,
- sendet kurzzeitig auf der Steuerleitung $\overline{\text{IOW}}$ L-Signal aus und löst damit das "Ausgabesignal" (L-H-Wechsel) aus, wo-durch die Daten in den Speicher der Ausgabe-Baugruppe übernommen und an den Ausgangs-buchsen ausgegeben werden.

1.4. Der Aufbau einer parallelen Eingabe-Baugruppe

Die parallele Eingabe-Baugruppe unterscheidet sich von der Ausgabe-Baugruppe nur dadurch, daß sich die Datenflußrichtung ändert und an die Stelle der Speicher elektronische Schalter treten, über die die Eingangssignale auf den Daten-Bus geschaltet werden können (Bild 11). Das Rücksetzsignal

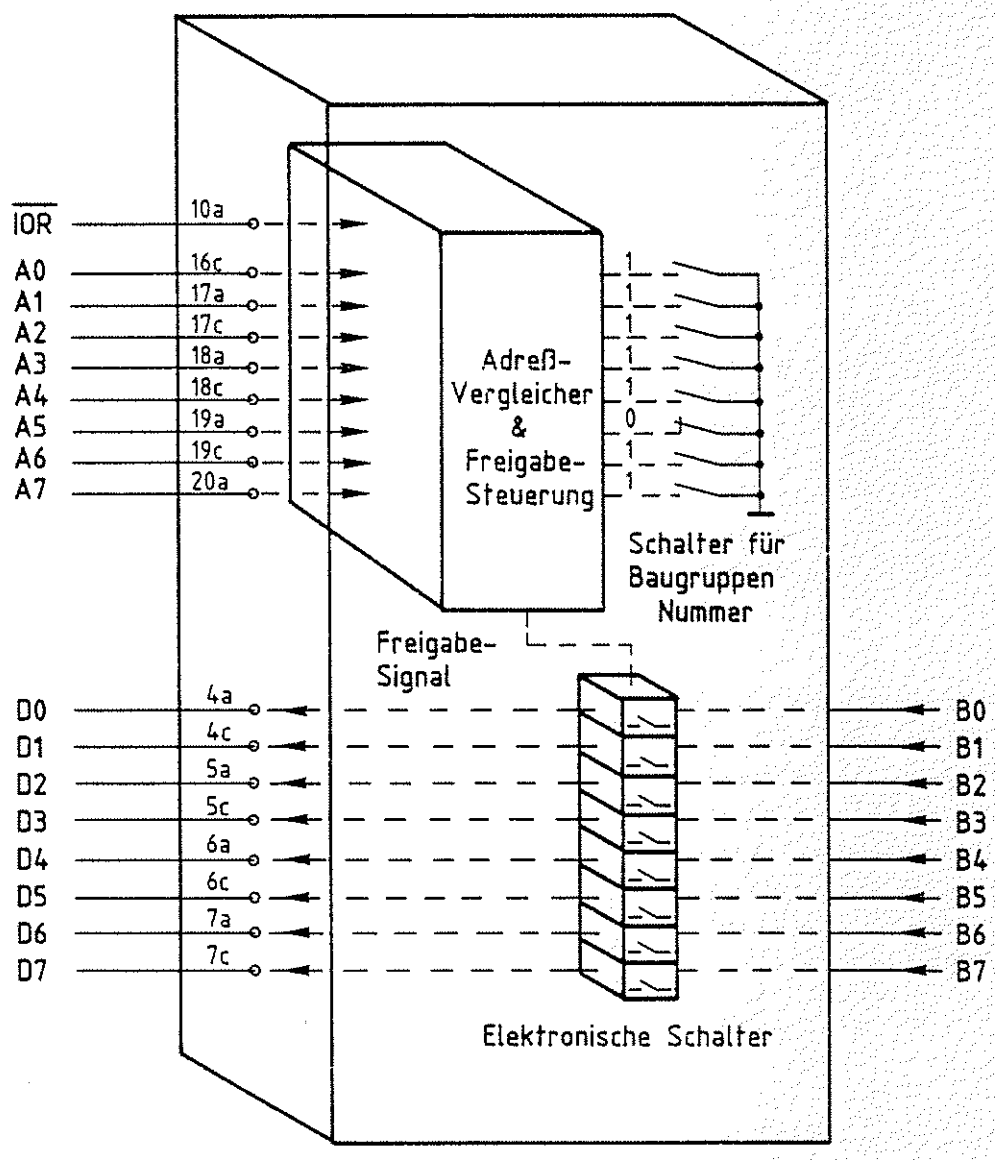


Bild 11: Aufbau einer parallelen Eingabe-Baugruppe

ist hier nicht erforderlich. Als auslösendes Steuersignal wird für die Eingabe-Baugruppen das Signal INPUT/OUTPUT READ verwendet. Dieses Steuersignal ($\overline{IOR}$, aktiv low) wird hier einer Freigabesteuerung zugeführt. Meldet der Adreßvergleicher, daß die eingestellte Baugruppennummer und der Signalzustand auf den Adreßleitungen A0 bis A7 übereinstimmen, so wird mit L-Signal auf der Leitung $\overline{IOR}$ der Signalzustand der Eingangsleitungen auf den Daten-Bus geschaltet. Neben den 256 Ausgabe-Baugruppen können somit noch 256 Eingabe-Baugruppen an das Bus-System des Mikrocomputers angeschlossen werden.

INPUT/OUTPUT READ (IOR)

Das Lesen des Signalzustandes an den Eingangsleitungen der Eingabe-Baugruppe vollzieht sich in folgenden Schritten:

Der Prozessor...

- wählt durch Aussenden einer Nummer (Adresse) auf den unteren acht Adreßleitungen die gewünschte Eingabe-Baugruppe aus,
- sendet kurzzeitig auf der Steuerleitung $\overline{IOR}$ L-Signal aus, wodurch die Eingabe-Baugruppe die Daten auf dem Daten-Bus bereitstellt,
- übernimmt diesen Signalzustand vom Daten-Bus flankengesteuert mit dem L-H-Pegelwechsel auf der Steuerleitung $\overline{IOR}$ und speichert die Daten intern in einem besonderen Speicher, dem Akkumulator.

Akkumulator

Häufig sind Eingabe-Baugruppen noch mit Speichern versehen, die beim Auslösen des Steuersignals $\overline{IOR}$ den letzten Signalzustand vor dem Auftreten des Steuersignals $\overline{IOR}$ festhalten und auf den Daten-Bus schalten. Dadurch haben Signalwechsel an den Eingängen, die während des Lesevorgangs zu Störungen führen können, keinen Einfluß. Dies ist möglich, da der Lesevorgang sehr schnell abläuft; er dauert nur einige hundert Nanosekunden.

Eingabespeicher

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

EIN-UND AUSGABE-EINHEITEN BFZ/MFA 10.2.

ÜBUNGSTEIL 1

Nicht alle Mikroprozessoren senden für Ein- Ausgabe-Einheiten besondere Steuersignale ($\overline{IOR}$ und $\overline{IOW}$) aus. Viele Mikroprozessoren behandeln die Ein- und Ausgabe-Baugruppen wie Speicherplätze (siehe FTÜ BFZ/MFA 10.3.). Besondere Steuersignale werden z.B. bei den Prozessoren 8080/8085, Z80, NBC800 bereitgestellt. Dieses Verfahren wird I/O-mapping genannt. Statt von Ein- Ausgabe-Baugruppen spricht man hier häufig von Ein- Ausgabe-Ports. "Port" bedeutet im Englischen Pforte und bezeichnet das Tor zur Umwelt des Computers.

I/O-mapping

Ein- Ausgabe-Port

Übungsteil 1

In den folgenden Arbeitsschritten werden Sie mit Hilfe des Bus-Signalgebers und der Bus-Signalanzeige Daten an eine Ausgabe-Baugruppe übergeben bzw. von einer Eingabe-Baugruppe lesen. Sie werden damit die einzelnen Funktionsschritte auslösen, die später der Prozessor ausführt.

Dazu benötigen Sie:

- 1 Baugruppenträger mit Busverdrahtung (BFZ/MFA 0.1.)
- 1 Bus-Abschluß (BFZ/MFA 0.2.)
- 1 Trafo-Einschub (BFZ/MFA 1.1.)
- 1 Spannungsregelung (BFZ/MFA 1.2.)

} zusammengebaut und geprüft nach FPÜ BFZ/MFA 1.2. Arbeitsblatt 7

- 1 Bus-Signalgeber (BFZ/MFA 5.1.)
- 1 Bus-Signalanzeige (BFZ/MFA 5.2.)
- 1 8-Bit-Parallel-Ausgabe (BFZ/MFA 4.1.)
- 1 8-Bit-Parallel-Eingabe (BFZ/MFA 4.2.)
- 1 Adapter 64polig (BFZ/MFA 5.3.)
- 2 Meßleitungen

Allgemeine Hinweise zur Durchführung der Übungen:

- Die Einschübe dürfen nur bei abgeschalteter Betriebsspannung gesteckt oder gezogen werden
- Aufgrund der Busverdrahtung können die Baugruppen in beliebige Steckplätze gesteckt werden
- Den logischen Signalen "0" und "1" sind die folgenden Pegel zugeordnet:

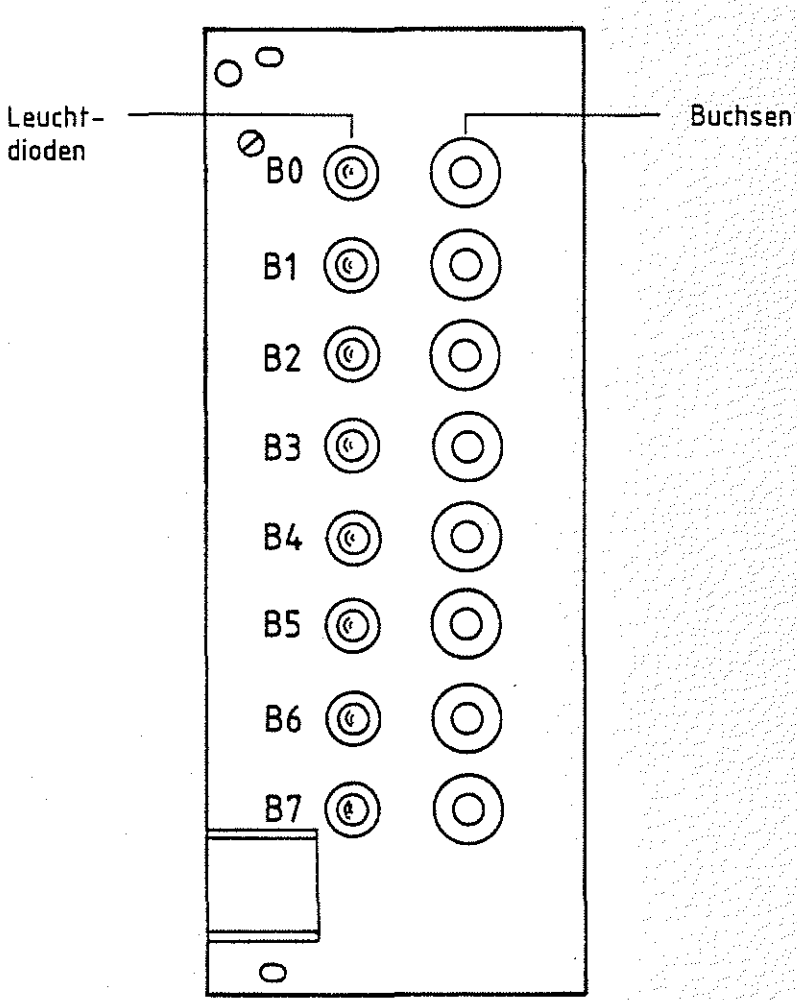
  log. "0" ≙ 0...0,8 V (LOW)
  log. "1" ≙ 2,4...5 V (HIGH)

- Alle zur Messung an den Baugruppen vorgegebenen Arbeitsblätter enthalten:
  = Angaben über den Sinn der jeweiligen Messung
  = Angaben über einzustellende Bedingungen (z.B. Schalterstellungen)
  = Aufgabenstellungen, ggf. mit Hinweisen zu möglichen Fehlern.

Bedienungshinweise:

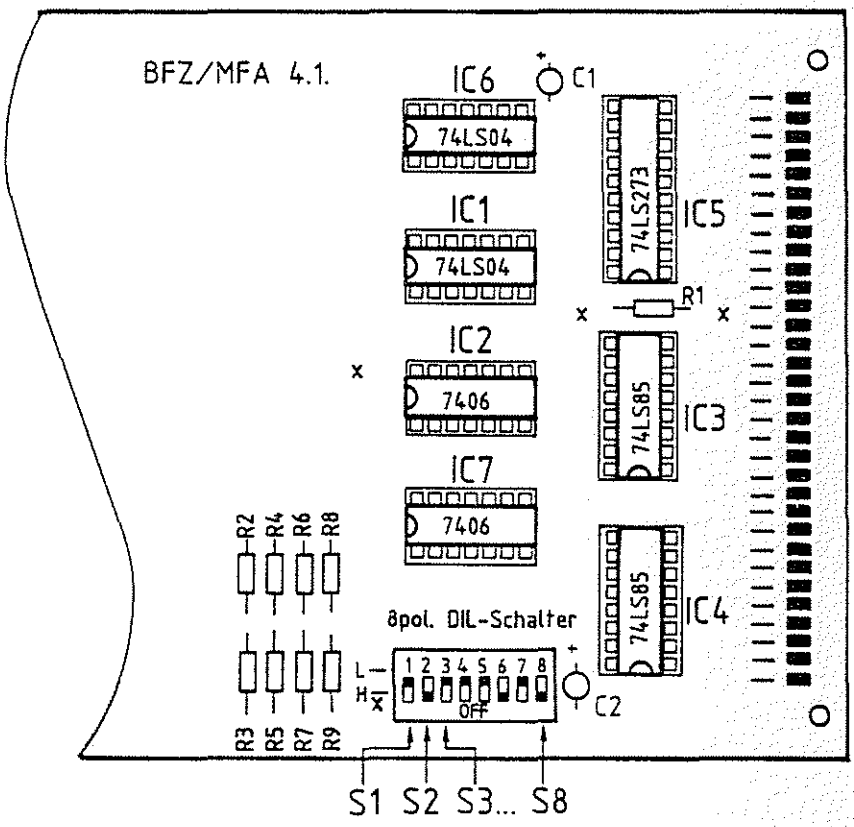
Ausgabe-Baugruppe (BFZ/MFA 4.1.):

In der Frontplatte der Ausgabe-Baugruppe sind Buchsen eingebaut, an denen die Ausgangssignale abgegriffen werden können. Der jeweilige Signalzustand wird durch Leuchtdioden angezeigt. Die Anschlußdaten für den Anschluß externer Geräte können Sie in der FPÜ BFZ/MFA 4.1. nachlesen. Da der Bus-Signalgeber im Gegensatz zum Prozessor kein Rücksetzsignal liefert, können nach dem Einschalten der Betriebsspannung einige Ausgänge H-Signal führen.



Ausgabe-Baugruppe

Die Nummer (Adresse) der Baugruppe kann an dem 8-poligen DIL-Schalter S1 bis S8 eingestellt werden. S1 ist bestimmend für die Adreßleitung A0, S2 für A1, usw.. Ein geöffneter Schalter (Stellung OFF) entspricht einem H-Signal, ein geschlossener (Stellung ON) dem L-Signal. In der untenstehenden Tabelle sind einige Schalterstellungen mit den zugehörigen Port-Nummern aufgeführt.



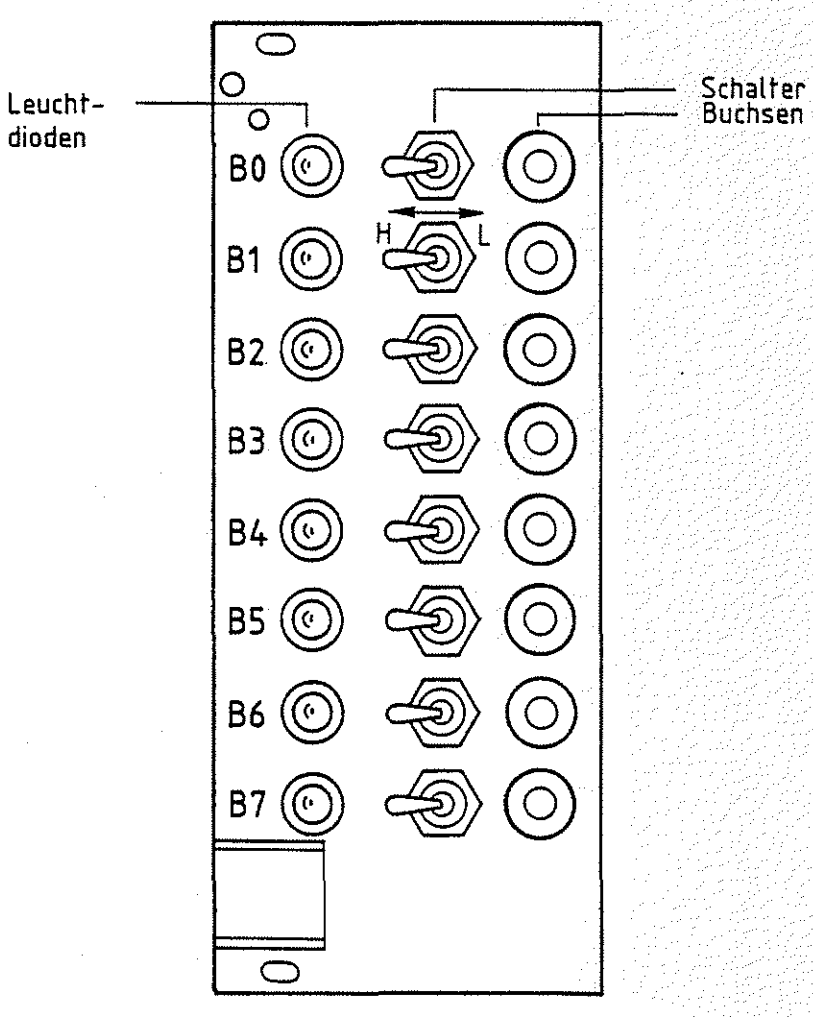
Ausgabe-Baugruppe

Schalter								Port-Nr.
S8	S7	S6	S5	S4	S3	S2	S1	
L	L	L	L	L	L	L	L	00
L	L	L	L	L	L	L	H	01
L	L	L	L	L	L	H	L	02
H	H	H	H	H	H	H	L	FE
H	H	H	H	H	H	H	H	FF

ON ≙ L-Signal
OFF ≙ H-Signal

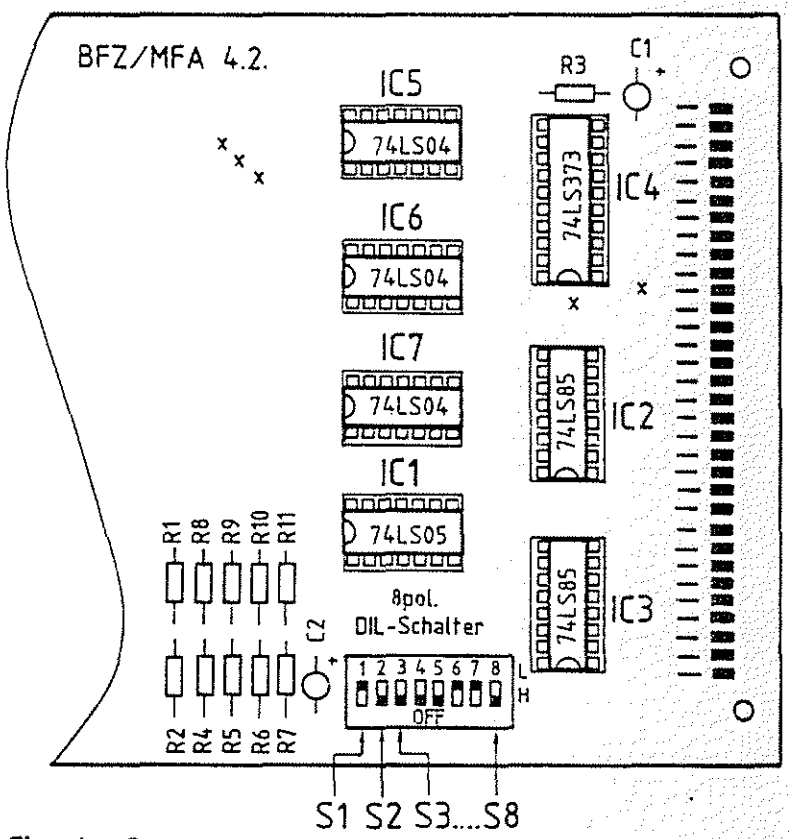
Eingabe-Baugruppe (BFZ/MFA 4.2.):

Die Eingangssignale der Eingabe-Baugruppe können entweder über Schalter eingestellt, oder über Buchsen der Baugruppe zugeführt werden. Der jeweilige Signalzustand an den Eingängen wird ebenso wie an der Ausgabe-Baugruppe über Leuchtdioden angezeigt. Werden die Buchsen für die externe Signalzuführung verwendet, so ist zu beachten, daß alle zugehörigen Schalter zuvor in die Stellung gebracht werden, die dem H-Signal (LED's leuchten) entspricht. Die Anschlußdaten können Sie der FPÜ BFZ/MFA 4.2. entnehmen.



Eingabe-Baugruppe

Die Einstellung der Baugruppen-Nummer (Port-Adresse) erfolgt ebenso wie auf der Ausgabe-Baugruppe.



Eingabe-Baugruppe

Ein- und Ausgabe-Einheiten Name:

Übungsteil 1 Datum:

A1

Übergabe von Daten an die Ausgabe-Baugruppe

Stellen Sie die Port-Adresse der Ausgabe-Baugruppe auf A7H ein. Stecken Sie Bus-Signalgeber, Bus-Signalanzeige und Ausgabe-Baugruppe in den Baugruppenträger ein und schalten Sie die Betriebsspannung ein.

Übergeben Sie nacheinander der Ausgabe-Baugruppe mit der Port-Nummer A7H die Daten-Bytes, die den Ampelphasen der Fußgängerampel entsprechen (Tabelle).

Phasen-Nr.	Daten-Bytes								
	binär								hex.
	B7	B6	B5	B4	B3	B2	B1	B0	
0	0	0	0	0	1	1	0	0	0C
1	0	0	0	0	1	0	1	0	0A
2	0	0	0	1	0	0	0	1	11
3	0	0	0	0	1	0	1	1	0B
4	0	0	0	0	1	1	0	0	0C

nicht benötigt | Fußgänger | Auto

Ein- und Ausgabe-Einheiten

Name:

Übungsteil 1

Datum:

A2

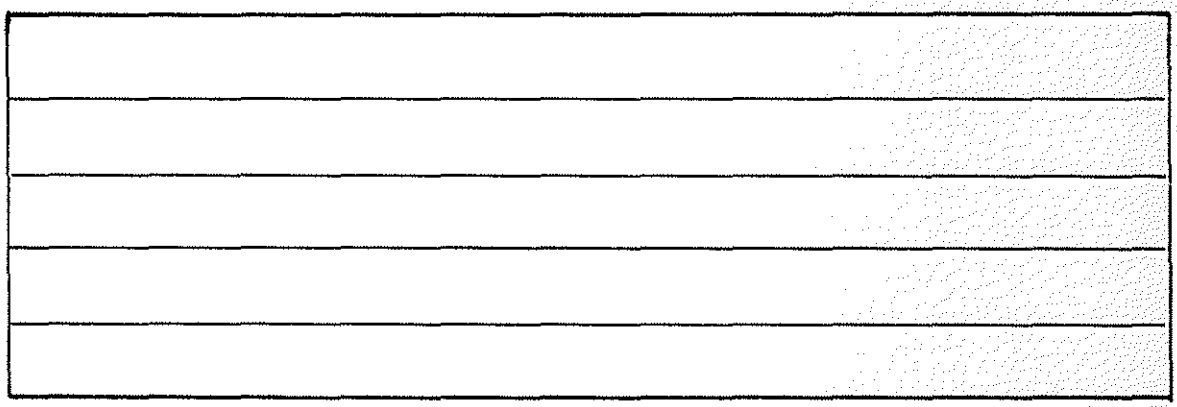
Prüfen der Reset-Funktion

Stecken Sie die Adapter-Karte zusätzlich in den Baugruppenträger.

Stellen Sie durch entpsrechende Datenübergabe an die Ausgabe-Baugruppe einen beliebigen Signalzustand ein, jedoch nicht an allen Ausgängen L-Signal. Überprüfen Sie die Rücksetzfunktion durch das RESET-Signal, indem Sie kurzzeitig L-Signal über eine Meßleitung an Stift 13c anlegen. (Über Adapter-Karte)

Gehen Sie bitte beim Anlegen des L-Signals vorsichtig vor, damit Sie auf anderen Leitungen keinen Kurzschluß verursachen.

Beschreiben Sie die Wirkung der Reset-Funktion:



Ein- und Ausgabe-Einheiten Name:

Übungsteil 1 Datum:

A3

Lesen von Daten der Eingabe-Baugruppe

Stellen Sie die Port-Adresse der Eingabe-Baugruppe auf 5CH ein. Stecken Sie anstelle der Ausgabe- nun die Eingabe-Baugruppe in den Baugruppenträger.

Stellen Sie verschiedene Daten an den Schaltern der Eingabe-Baugruppe ein und lesen Sie den Signalzustand über das Bus-System ein, so wie es der Prozessor durch Auslösen der entsprechenden Funktionen macht.
Protokollieren Sie Ihre Arbeitsschritte:

Überprüfen Sie die Speicherwirkung während des Lesevorgangs der Eingabe-Daten, indem Sie während der Betätigung des Steuersignals $\overline{\text{IOR}}$ die Eingabe-Daten verändern. Die Daten auf dem Daten-Bus dürfen sich dabei nicht ändern.

Ein- und Ausgabe-Einheiten Name:

Übungsteil 1 Datum:

A4.1

Lesen der Daten der Eingabe-Baugruppe und Ausgabe dieser Daten an die Ausgabe-Baugruppe

Stellen Sie an der Eingabe-Baugruppe die Port-Adresse 12 und an der Ausgabe-Baugruppe die Port-Adresse 13 ein. Stecken Sie dann beide Baugruppen in den Baugruppenträger.

Sie sollen nun die Funktion eines Prozessors übernehmen, indem Sie ein vorgegebenes Programm abarbeiten. Das Programm liegt in Form einer schriftlichen Anweisung vor. Gehen Sie ebenso wie der Prozessor vor und lesen Sie zunächst die erste Anweisung. Danach führen Sie die Anweisung sofort aus. Lesen Sie dann die nächste Anweisung usw.. Das zu bearbeitende Programm, das für einen Prozessor natürlich in verschlüsselter Form im Speicher abgelegt sein muß, lautet wie folgt:

1. Anweisung/Befehl
   Lesen Sie den Signalzustand der Eingabe-Baugruppe mit der Port-Adresse 12.
2. Anweisung/Befehl
   Schreiben Sie den zuvor gelesenen Signalzustand zur Ausgabe-Baugruppe mit der Port-Adresse 13.
3. Anweisung/Befehl
   Fahren Sie mit der Programmabarbeitung bei der 1. Anweisung fort.

Ein solches Programm, das ständig abgearbeitet wird, nennt man ein zyklisches Programm. Verändern Sie während der Abarbeitung auch die Eingangssignale an der Eingabe-Baugruppe.



Arbeitsblatt

BFZ / MFA 10.2. - 24

Ein- und Ausgabe-Einheiten

Name:

Übungsteil 1

Datum:

A4.2

Protokollieren Sie Ihr Vorgehen zur Bearbeitung der einzelnen Programm-Befehle:

Ende Übungsteil 1

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

EIN-UND AUSGABE-EINHEITEN BFZ/MFA 10.2.

THEORIETEIL 2

2.1. Vereinfachte Schaltung der Ausgabe-Baugruppe

Bild 12 zeigt die Schaltung der Ausgabe-Baugruppe, bei der einige Schaltungsdetails gegenüber der ausgeführten Schaltung der besseren Übersicht wegen weggelassen worden sind.

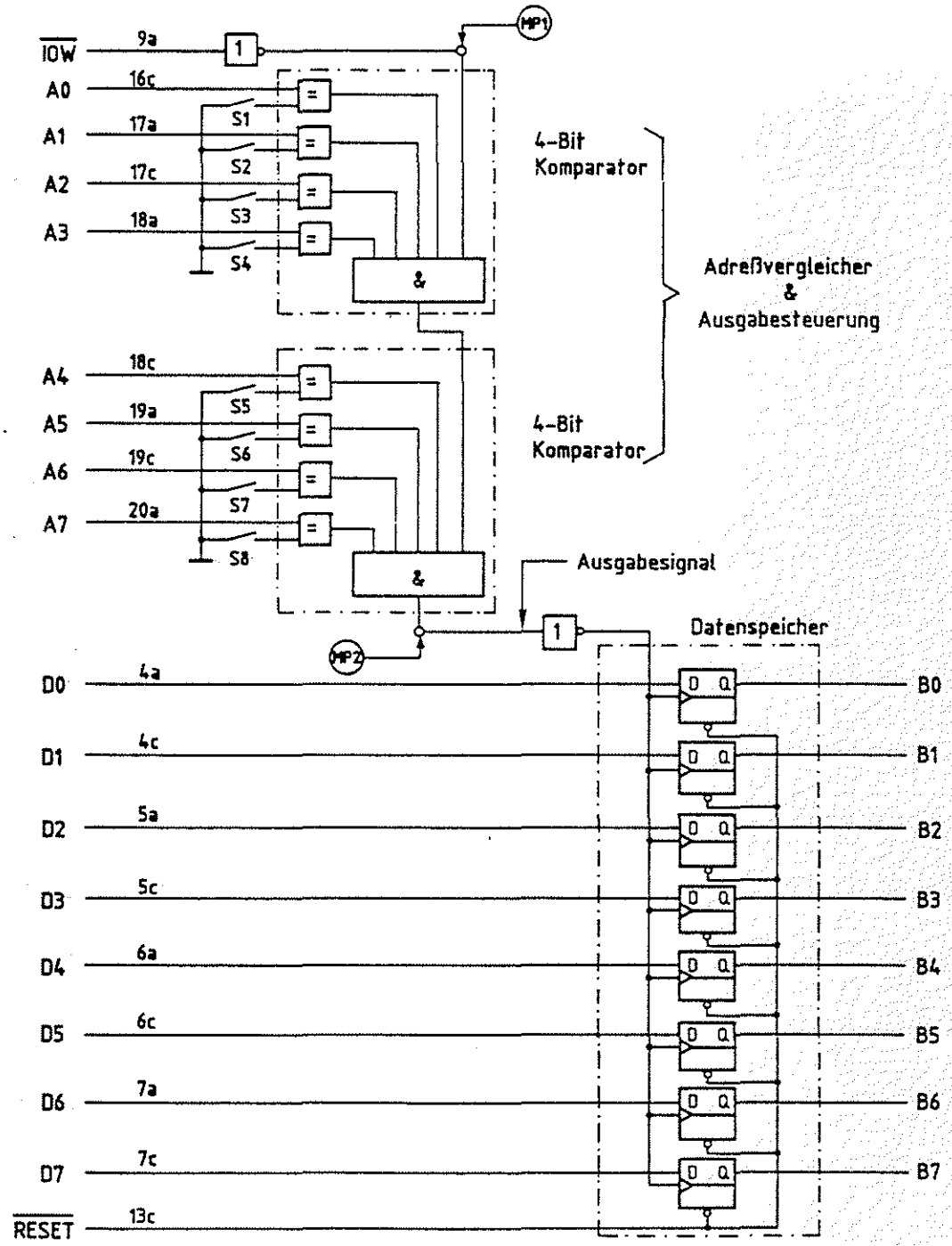


Bild 12: Vereinfachte Schaltung der Ausgabe-Baugruppe

Außer dem Adreßvergleicher und dem Datenspeicher werden nur wenige andere Bausteine verwendet. Der Datenspeicher besteht aus acht D-Flipflops, die es speziell für diese Anwendungsfälle als fertiges IC gibt. Diese Integrierten Schaltkreise nennt man auch Daten-Latch. Intern sind alle Takt- und Rücksetzeingänge der D-Flipflops miteinander verbunden. Die Flipflops sind flankengesteuert und übernehmen bei einem positiven Signalwechsel (L-H-Signalwechsel) am Takteingang den Signalzustand des D-Eingangs. Dieser Signalzustand wird gespeichert und zum Q-Ausgang durchgeschaltet.

Daten-Latch (Latch = Falle, Klinke)

D-Flipflop

positiv-flankengesteuert

Der Adreßvergleicher wird mit Hilfe von Komparatoren (engl. compare, vergleichen) aufgebaut. Intern werden Komparatoren mit Äquivalenzgattern realisiert. Äquivalenzgatter liefern am Ausgang nur dann H-Signal, wenn der Signalzustand beider Eingänge übereinstimmt (äquivalent ist) (Bild 13).

Komparator

Äquivalenz-Gatter



E2	E1	A
L	L	H
L	H	L
H	L	L
H	H	H

Bild 13: Schaltsymbol und Funktionstabelle eines Äquivalenzgatters

Die Ausgangssignale der Äquivalenzgatter werden im Komparator UND-verknüpft, so daß nur bei paarweiser Gleichheit aller Adreß- und Schaltersignale am Ausgang H-Signal auftritt. Für die Hintereinanderschaltung (Kaskadierung) besitzen die UND-Gatter einen weiteren Eingang, über den in der Schaltung Bild 12 gleichzeitig eine Verknüpfung mit dem Steuersignal IOW erfolgt. Ein Ausgabesignal am Ausgang des Adreßvergleichers (H-Pegel) tritt nur dann auf, wenn ...

Kaskadierung

- die Schaltersignale S1 bis S8 mit den Signalzuständen auf den Adreßleitungen A0 bis A7 übereinstimmen und

— das Steuersignal $\overline{IOW}$ aktiv ist, d.h. die Leitung $\overline{IOW}$ auf L-Signal liegt.

Da die D-Flipflops positiv flankengesteuert sind, muß dieses Ausgabesignal noch einmal invertiert werden, bevor es an die Takteingänge der Flipflops gelangt. Das Signal- Zeit-Diagramm für die Datenübergabe an die Ausgabe-Baugruppe durch den Prozessor ist in Bild 14 dargestellt.

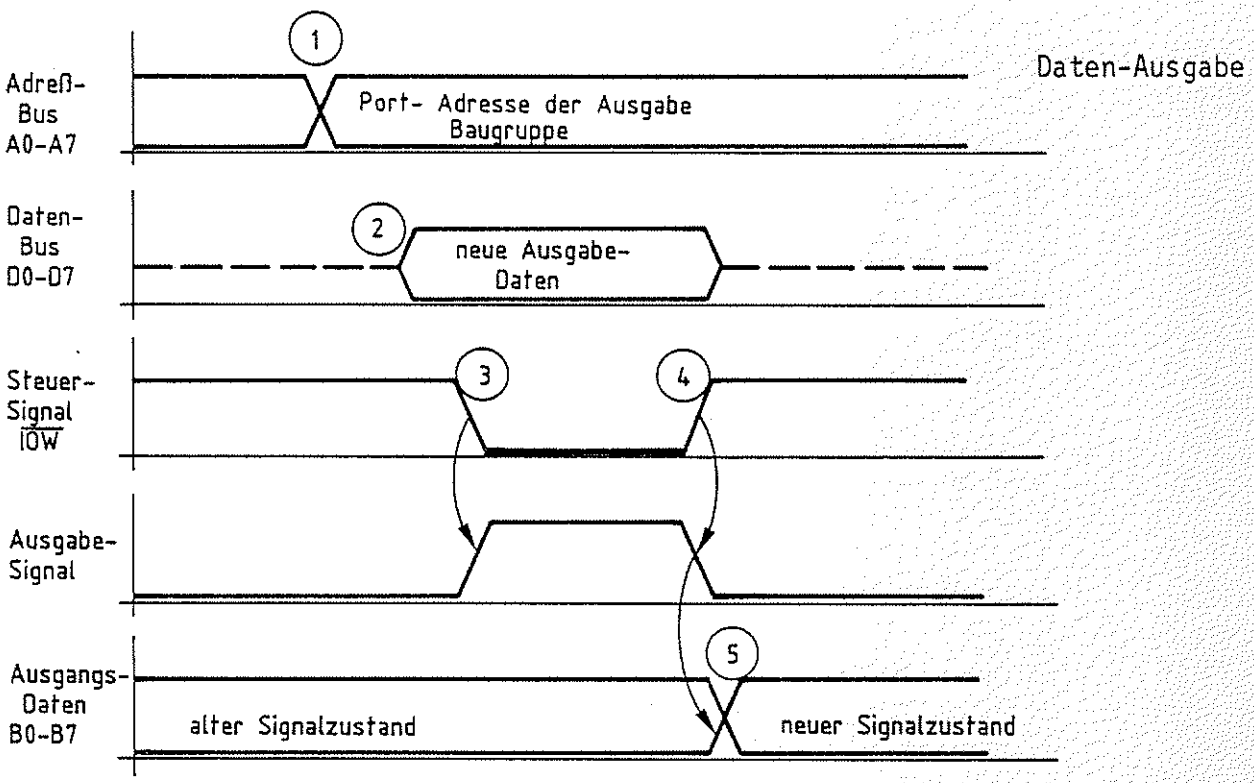


Bild 14: Datenausgabe an der Ausgabe-Baugruppe

Zum Zeitpunkt 1 sendet der Prozessor die Port-Adresse auf den Adreßleitungen A0 bis A7 aus.
Zum Zeitpunkt 2 stellt er die Ausgabedaten auf dem Daten-Bus bereit.
Zum Zeitpunkt 3 aktiviert er das Steuersignal $\overline{IOW}$.

Die Übernahme der Daten in die Speicher bzw. ihre Ausgabe an die Ausgangsbuchsen zum Zeitpunkt 5 erfolgt dann mit dem Wegschalten des Steuersignals $\overline{IOW}$ zum Zeitpunkt 4.

2.2. Vereinfachte Schaltung der Eingabe-Baugruppe

Wie schon im Theorieteil 1 erwähnt, unterscheiden sich Aus- und Eingabe-Baugruppen hauptsächlich in der Datenflußrichtung. An die Stelle der acht D-Flipflops treten im einfachsten Fall Tristate-Gatter, über die die Eingangssignale nach Anforderung durch den Prozessor auf den Daten-Bus geschaltet werden (Bild 15).

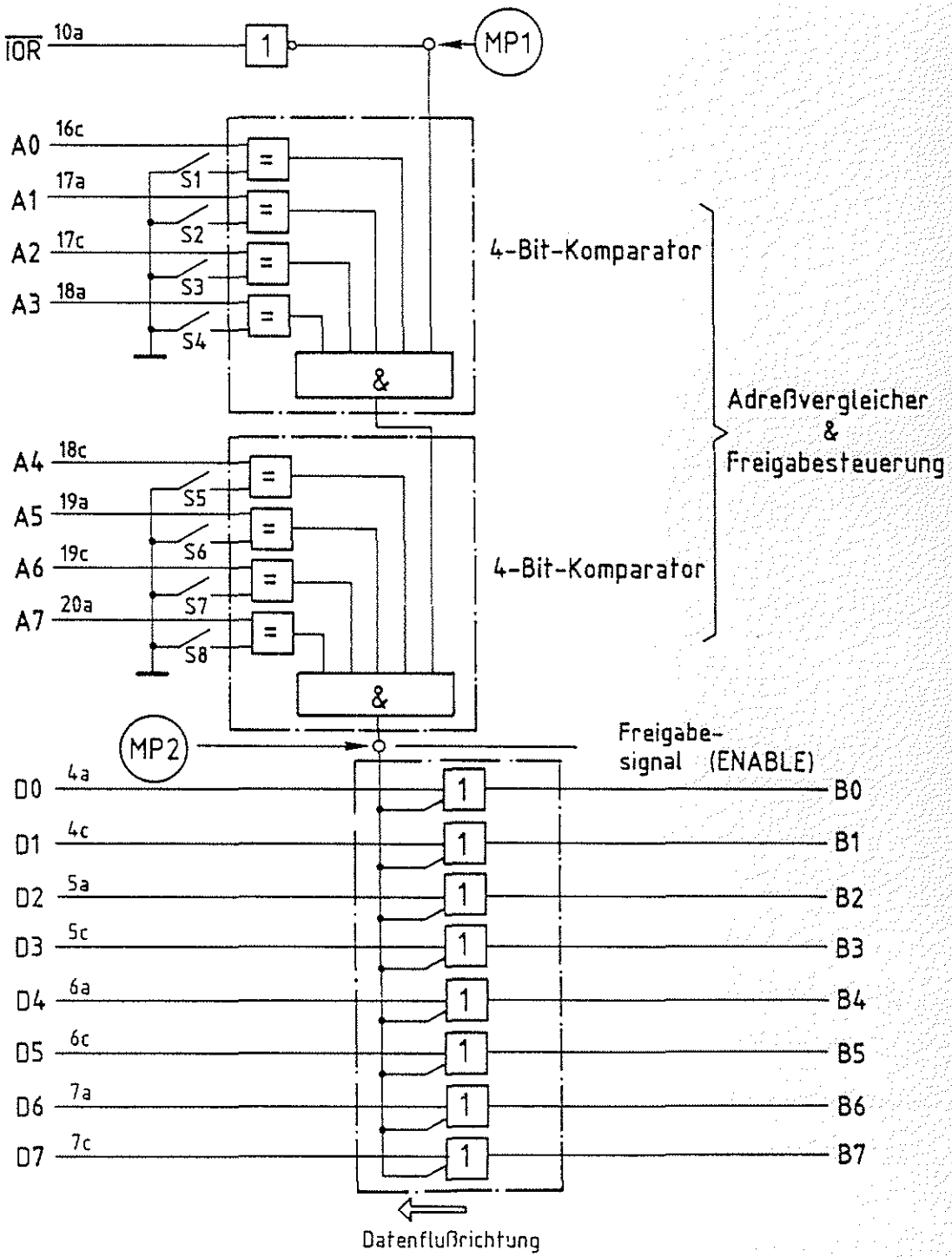


Bild 15: Vereinfachte Schaltung der Eingabe-Baugruppe

Das Freigabesignal für die Tristate-Gatter tritt nur dann auf, wenn die Adreßsignale A0 bis A7 paarweise mit den Schaltersignalen S1 bis S8 übereinstimmen und hier das Steuersignal $\overline{IOR}$ aktiv ist. Eine Speicherung der Eingangssignale während des Lesevorgangs erreicht man, indem man z.B. zwischen die Dateneingänge B0...B7 und die Tristate-Gatter D-Flipflops schaltet. Diese wurden im Schaltbild nicht berücksichtigt, da sie nicht unbedingt erforderlich sind. Das Signal- Zeit-Diagramm für den Lesevorgang durch den Prozessor ist in Bild 16 dargestellt.

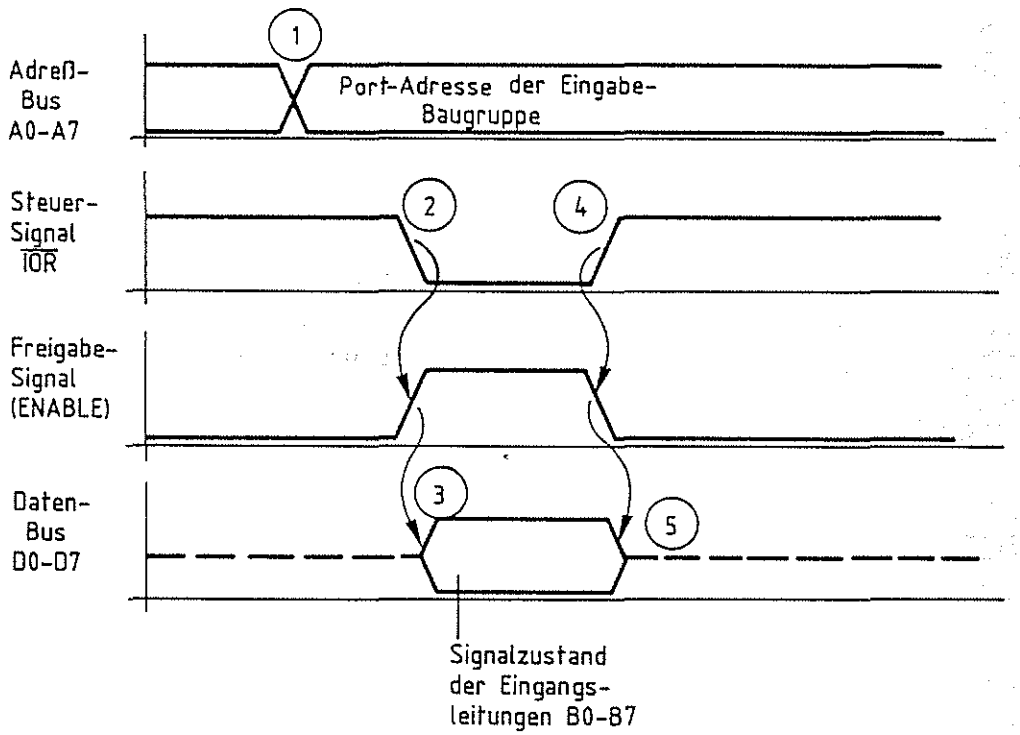


Bild 16: Dateneingabe in die Eingabe-Baugruppe

Zum Zeitpunkt 1 sendet der Prozessor die Port-Adresse auf den Adreßleitungen A0 bis A7 aus.
Zum Zeitpunkt 2 veranlaßt er durch L-Signal auf der Leitung $\overline{IOR}$, daß die Eingabe-Baugruppe die Eingangssignale auf den Bus schaltet. Dies geschieht etwas verzögert zum Zeitpunkt 3. Mit dem L-H-Signalwechsel auf der Steuerleitung $\overline{IOR}$ übernimmt der Prozessor zum Zeitpunkt 4 die Daten vom Daten-Bus und beendet damit gleichzeitig den Lesevorgang, so daß die Eingabe-Baugruppe zum Zeitpunkt 5 die Daten vom Bus wegschaltet.

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

EIN-UND AUSGABE-EINHEITEN BFZ/MFA 10.2.

ÜBUNGSTEIL 2

Übungsteil 2

In den folgenden Arbeitsschritten werden Sie einige Signale auf der Ein- und der Ausgabe-Baugruppe meßtechnisch überprüfen.

Dazu benötigen Sie:

- 1 Baugruppenträger mit Busverdrahtung (BFZ/MFA 0.1.)
- 1 Bus-Abschluß (BFZ/MFA 0.2.)
- 1 Trafo-Einschub (BFZ/MFA 1.1.)
- 1 Spannungsregelung (BFZ/MFA 1.2.)

} zusammengebaut und geprüft nach FPO BFZ/MFA 1.2. Arbeitsblatt 7

- 1 Bus-Signalgeber (BFZ/MFA 5.1.)
- 1 Bus-Signalanzeige (BFZ/MFA 5.2.)
- 1 8-Bit-Parallel-Ausgabe (BFZ/MFA 4.1.)
- 1 8-Bit-Parallel-Eingabe (BFZ/MFA 4.2.)
- 1 Adapter 64polig (BFZ/MFA 5.3.)
- 1 Logiktester oder Vielfachmeßinstrument
- 2 Meßleitungen

Allgemeine Hinweise zur Durchführung der Übungen:

- Die Einschübe dürfen nur bei abgeschalteter Betriebsspannung gesteckt oder gezogen werden
- Aufgrund der Busverdrahtung können die Baugruppen in beliebige Steckplätze gesteckt werden
- Den logischen Signalen "0" und "1" sind die folgenden Pegel zugeordnet:

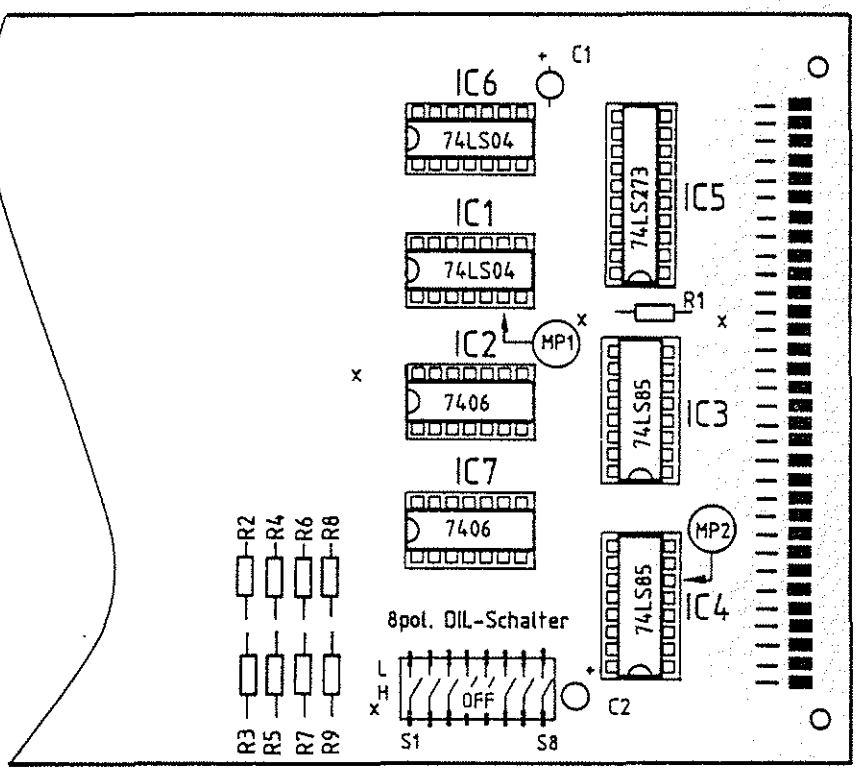
  log. "0" ≙ 0...0,8 V (LOW)
  log. "1" ≙ 2,4...5 V (HIGH)

- Alle zur Messung an den Baugruppen vorgegebenen Arbeitsblätter enthalten:
  = Angaben über den Sinn der jeweiligen Messung
  = Angaben über einzustellende Bedingungen (z.B. Schalterstellungen)
  = Aufgabenstellungen, ggf. mit Hinweisen zu möglichen Fehlern.

Bedienungshinweise:

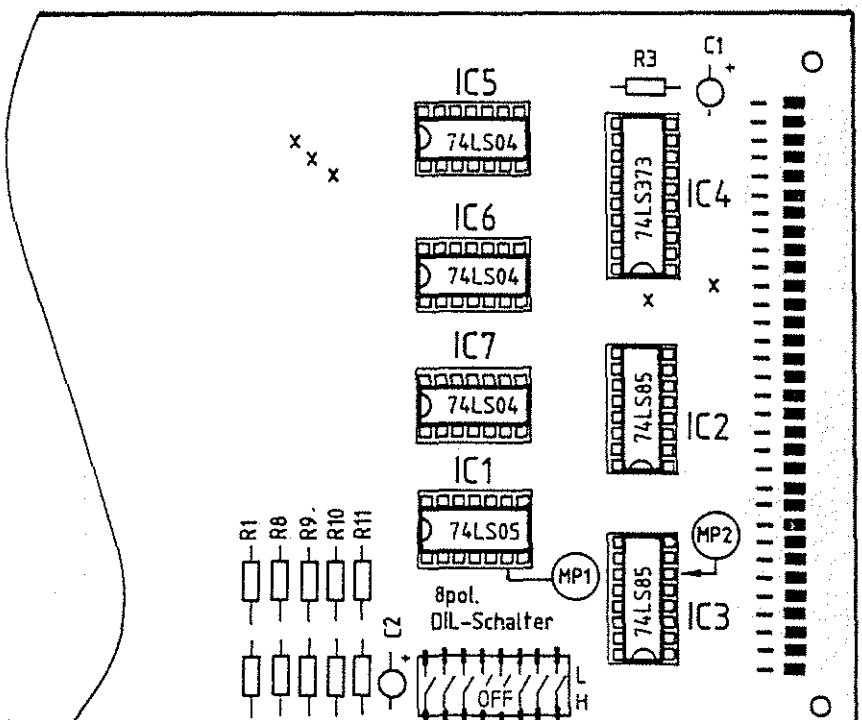
Die Bilder 17 und 18 zeigen jeweils einen Ausschnitt der Bestückungsseite der Aus- und der Eingabe-Baugruppe. In den Bildern sind diejenigen Meßpunkte eingetragen, die Sie in den vereinfachten Schaltungen der Baugruppen (Bild 12 und Bild 15) wiederfinden.

An diesen Meßpunkten werden Sie in den folgenden Arbeitsschritten Messungen durchführen. Seien Sie dabei besonders vorsichtig, damit keine Kurzschlüsse entstehen, die eventuell Bauteile zerstören.



MP1 = Steuersignal IOW (aktiv High)
MP2 = Ausgabesignal (aktiv High)

Bild 17: Ausgabe-Baugruppe BFZ/MFA 4.1.



MP1 = Steuersignal IOR (aktiv High)
MP2 = Freigabesignal (ENABLE),(aktiv High)

Bild 18: Eingabe-Baugruppe BFZ/MFA 4.2.

Ein- und Ausgabe-Einheiten Name:

Übungsteil 2 Datum:

A1

Überprüfen des Adreßvergleichers der Ausgabe-Baugruppe

Stellen Sie die Port-Adresse 13 auf der Ausgabe-Baugruppe ein. Stecken Sie dann die Baugruppe über die Adapter-Karte neben Bus-Signalgeber und Bus-Signalanzeige in den Baugruppenträger und schalten Sie die Betriebsspannung ein.

a) Überprüfen Sie zunächst mit einem Logiktester oder Meßgerät das Steuersignal IOW am Eingang des Adreßvergleichers (MP1), indem Sie während der Messung das Steuersignal am Bus-Signalgeber auslösen.
Testen Sie dann in gleicher Weise das Ausgabesignal (MP2), wenn Sie mit dem Bus-Signalgeber zusätzlich noch die Port-Adresse (13) ausgeben.

b) Wiederholen Sie die Messung a) für den Fall, daß die vom Bus-Signalgeber ausgegebene Adresse nicht mit der Port-Adresse der Baugruppe (13) übereinstimmt.

c) Wiederholen Sie die Messung a) für den Fall, daß zwar der Bus-Signalgeber die Adresse 13 aussendet, Sie aber zuvor die Port-Adresse mit den Schaltern S1 bis S8 verändert haben.

	Ausgegebene Adresse = Port-Adresse		Ausgegebene Adresse (z.B.15) ≠ Port-Adresse (13)		Ausgegebene Adresse (13) ≠ Port-Adresse (z.B.23)	
	IOW-Taste ...		IOW-Taste ...		IOW-Taste ...	
	nicht betätigt	betätigt	nicht betätigt	betätigt	nicht betätigt	betätigt
IOW (MP1)						
Ausgabesignal (MP2)						

Ein- und Ausgabe-Einheiten Name:

Übungsteil 2 Datum:

A2

Überprüfen des Adreßvergleichers der Eingabe-Baugruppe

Stellen Sie auf der Eingabe-Baugruppe die Port-Adresse 12 ein. Ersetzen Sie die Ausgabe-Baugruppe durch die Eingabe-Baugruppe (über Adapter-Karte). Schalten Sie die Betriebsspannung ein.

a) Überprüfen Sie mit einem Logiktester oder Meßgerät das Steuersignal IOR am Eingang des Adreßvergleichers (MP1) bei gleichzeitiger Betätigung des Steuersignals am Bus-Signalgeber.
Testen Sie in gleicher Weise das Freigabesignal (MP2) für die Tristate-Gatter, wenn Sie mit dem Bus-Signalgeber zusätzlich noch die Port-Adresse 12 ausgeben.

b) Wiederholen Sie die Messung a) für den Fall, daß die vom Bus-Signalgeber ausgegebene Adresse nicht mit der Port-Adresse der Baugruppe (12) übereinstimmt.

c) Wiederholen Sie die Messung a) für den Fall, daß zwar der Bus-Signalgeber die Adresse 12 ausgibt, Sie aber zuvor die Port-Adresse mit den Schaltern S1 bis S8 verändert haben.

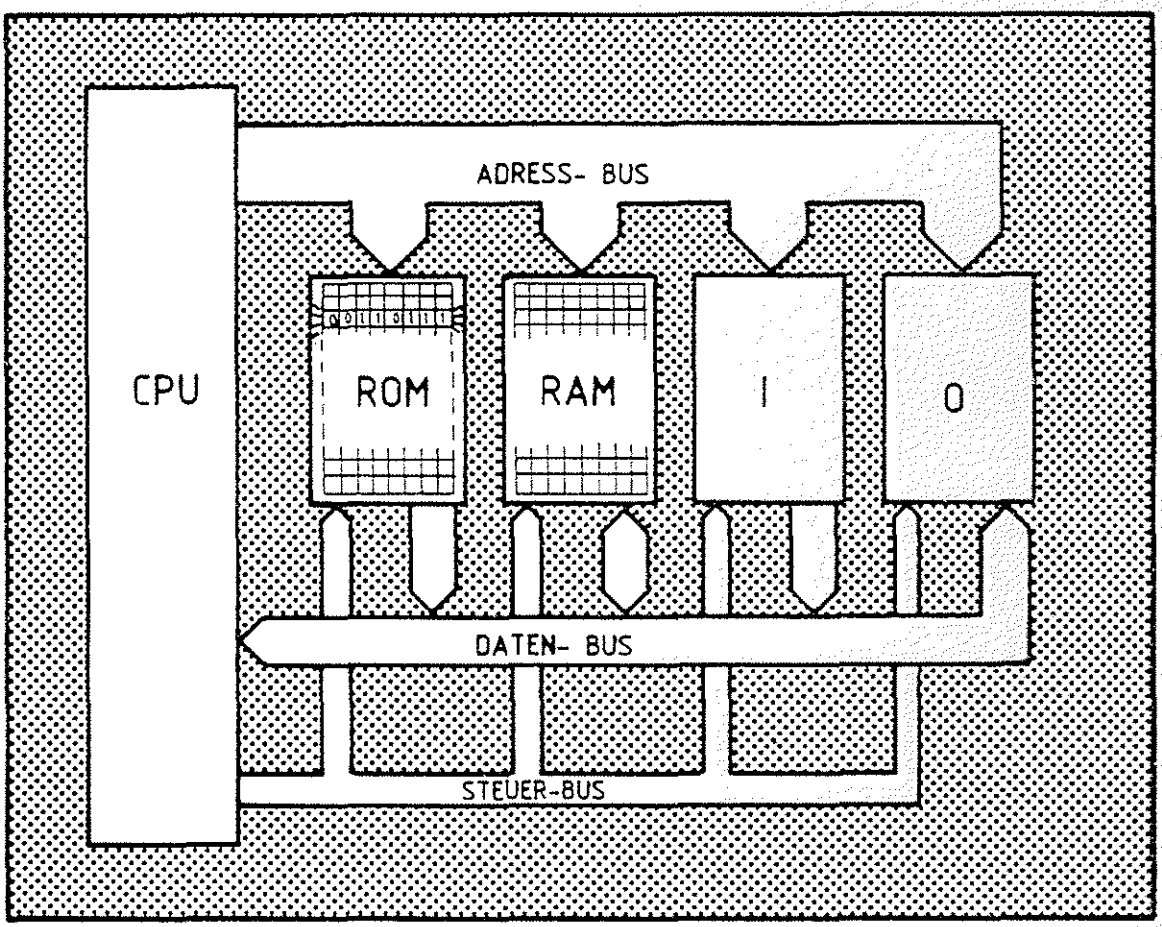
	Ausgegebene Adresse = Port-Adresse		Ausgegebene Adresse (z.B.25) ≠ Port-Adresse (12)		Ausgegebene Adresse (12) ≠ Port-Adresse (z.B.31)	
	IOR-Taste...		IOR-Taste...		IOR-Taste...	
	nicht betätigt	betätigt	nicht betätigt	betätigt	nicht betätigt	betätigt
IOR (MP1)						
Freigabesignal (MP2)						

Ende der Übung

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

SPEICHER-EINHEITEN

BFZ/MFA 10.3.



Diese Übung ist Bestandteil eines Mediensystems, das im Rahmen eines vom Bundesminister für Bildung und Wissenschaft, vom Bundesminister für Forschung und Technologie sowie der Bundesanstalt für Arbeit geförderten Modellversuches zum Einsatz der "Mikrocomputer-Technik in der Facharbeiterausbildung" vom BFZ-Essen e.V. entwickelt wurde.

Inhaltsverzeichnis

Theorieteil 1

1.1 Einleitung
1.2 Aufbau eines Speichers (Grundprinzip)
1.3 Speicher-Baugruppen
1.4 Aufbau der Speicherzellen
1.5 Bausteine für Festwertspeicher

Übungsteil 1

A1 Überprüfen des Inhalts von RAM-Speicherzeilen
A2 Eingabe eines kleinen Programms in den RAM-Speicher und Überprüfen des Speicherinhalts nach kurzzeitigem Abschalten der Betriebsspannung
A3 Lesen des Inhalts von ROM-Speicherzeilen

Theorieteil 2

2.1 Adreßdecodierung
2.2 Vereinfachte Schaltung einer Speicher-Baugruppe

Übungsteil 2

A1 Prüfen der Wirkung der Steuersignale am Eingang des Adreß-Vergleichers
A2 Prüfen des Baugruppen-Freigabe-Signals
A3 Prüfen der Baustein-Freigabe-Signale

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

SPEICHER-EINHEITEN

BFZ/MFA 10.3.

THEORIETEIL 1

1.1 Einleitung

Im Speicher des Mikrocomputers werden neben der Arbeitsanweisung (Programm), die der Prozessor bearbeiten soll, auch die zu verarbeitenden Daten und Zwischenergebnisse aufbewahrt. Bei diesen Speichern handelt es sich fast ausschließlich um elektronische Speicher (Halbleiterspeicher), in denen die Informationen in binärer Form (0- und 1-Informationen) abgelegt werden. Aus Ihrer Umgebung kennen Sie andere Speicher als Träger für Informationen, z.B. werden Sprache und Musik, also akustische Signale und Informationen, magnetisch auf einem Tonband gespeichert. Das Telefonbuch ist ein Speicher, der die Informationen in Form von Schriftzeichen auf Papier aufbewahrt. Das Telefonbuch entspricht einem Nur-Lese-Speicher (ROM), da die Informationen im Speicher nur gelesen werden können. Dagegen kann die Information auf einem Tonband überschrieben werden. Dies ist das Merkmal eines Schreib-Lese-Speichers (RAM). Gleichgültig um welchen Informationsspeicher es sich handelt, sie haben alle ein gemeinsames Merkmal: "Die Informationen liegen in geordneter Form vor." Eine Ordnungsstruktur ist notwendig, um gezielt auf die gespeicherten Daten zugreifen zu können. Beim Tonband findet man mit Hilfe des Zählwerks die Position der aufgenommenen akustischen Signale. Die Ordnungsstruktur im Telefonbuch ergibt sich aus der alphabetischen Reihenfolge der Telefonbesitzer. Der Speicher im Computer wird dadurch geordnet, daß jeder Speicherplatz eine Adresse hat. Jeder Adresse entspricht genau ein Signalzustand auf dem Adreß-Bus. Muß beim Tonband der Zählerstand und beim Telefonbuch der Teilnehmername bekannt sein, um die gewünschten Informationen (Daten) wiederzufinden, so muß im Computer die Adresse bekannt sein, unter der die gesuchten (benötigten) Daten gespeichert sind.

Halbleiterspeicher

Nur-Lese-Speicher
(Read only memory)

Schreib-Lese-
Speicher
(Random access
memory)

Ordnungsstruktur

1.2 Aufbau eines Speichers (Prinzip)

Im Bild 1 ist der Aufbau eines Speichers dargestellt. Jedem Signalzustand auf den Adreßleitungen A0 bis A15 ist eine Speicherzeile zugeordnet. Eine Speicherzeile umfaßt jeweils acht Bit und ist somit an die Daten-Bus-Struktur des Prozessors angepaßt.

Speicherzeile

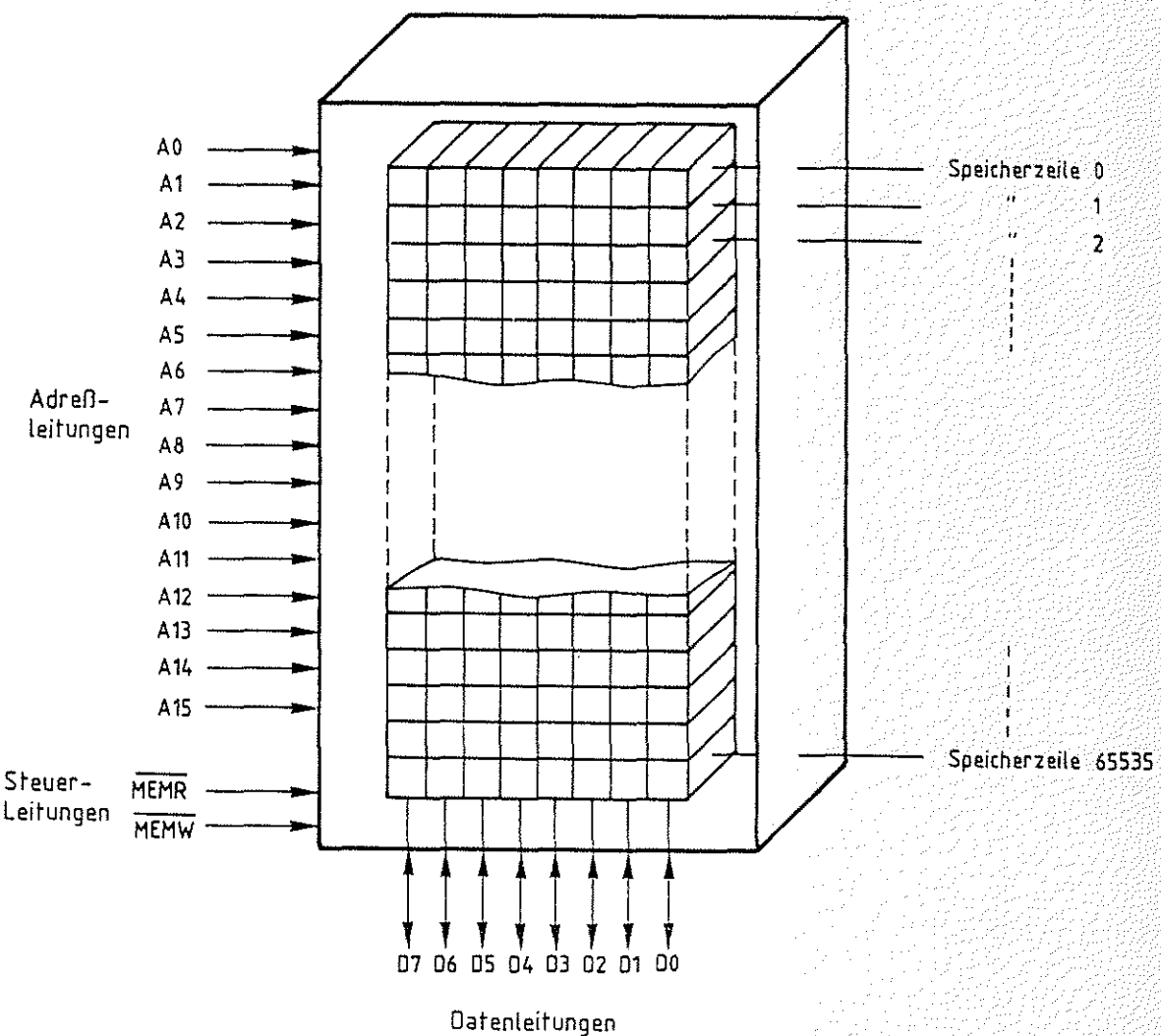


Bild 1: Aufbau eines Speichers (Prinzip)

Der Prozessor unterscheidet nach dem Aussenden einer Adresse auf dem Adreß-Bus zwischen dem Lesen des Inhalts einer Speicherzeile und dem Speichern von Daten in eine Speicherzeile. Das Lesen einer Speicherzeile erfolgt mit dem Steuersignal MEMORY READ (MEMR) und das Ablegen eines Daten-Bytes im Speicher mit MEMORY WRITE (MEMW). Die Überstreichung der Kurzbezeichnungen MEMR und MEMW macht deutlich, daß es sich um Low-aktive Signale handelt. Ein Teil der möglichen Speicheradressen auf dem Adreß-Bus ist in Bild 2 zusammengefaßt.

MEMORY READ

MEMORY WRITE

Signalzustände auf den Adreßleitungen																Hex. Adr.	Speicher-zeile
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	0000	0
L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	H	0001	1
L	L	L	L	L	L	L	L	L	L	L	L	L	L	H	L	0002	2
L	L	L	L	L	L	L	L	L	L	L	L	L	L	H	H	0003	3
L	L	L	L	L	L	L	L	L	L	L	L	L	H	L	L	0004	4
L	H	L	H	H	H	L	L	H	H	H	H	L	H	H	H	5CF7	23799
H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	FFFE	65534
H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	FFFF	65535

Bild 2: Mögliche Speicheradressen auf dem Adreß-Bus

Mit den 16 Adreßleitungen können 65536 (2^{16}) Speicherzeilen unterschieden werden. Zentrales Element hierzu in einem Speicher ist ein Adreß-Decoder (Bild 3). Dieser wählt aufgrund des Signalzustandes auf den Adreßleitungen die zugehörige Speicherzeile aus. Liegt beispielsweise am Adreß-Bus die Adresse 0000H an, so wird die Zeile 0 ausgewählt, bei 0001H die Zeile 1,... und bei FFFFH die Zeile 65535.

Adreß-Decoder

Ein Lesevorgang der Speicherzeile 0002H ist im Bild 3 dargestellt. Mit dem Steuersignal $\overline{MEMR}$ wird der Speicherinhalt auf den Daten-Bus geschaltet und kann vom Prozessor gelesen werden.

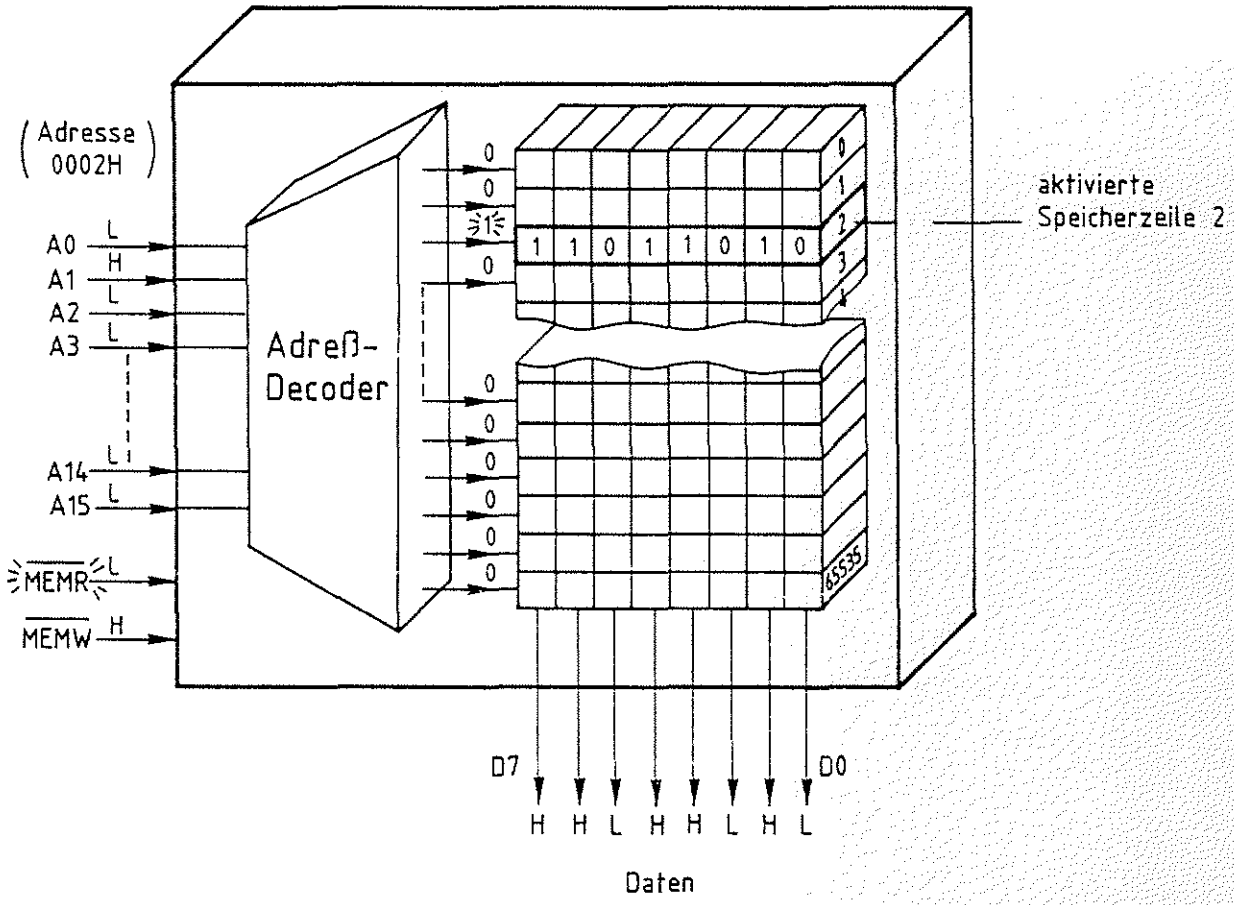


Bild 3: Funktionsweise des Adreß-Decoders

Ein Kennwert des Speichers ist neben der Speichergröße - auch Speicherkapazität genannt - insbesondere die Zugriffszeit. Das ist die Zeit, die der Speicher nach Aktivierung des entsprechenden Steuersignals benötigt, um den Speicherinhalt einer adressierten Speicherzeile bereitzustellen bzw. ein Daten-Byte in die Speicherzeile zu übernehmen. Diese Zeit liegt bei üblichen Speichern in der Größenordnung von einigen hundert Nanosekunden. Die Angabe der Speicherkapazität erfolgt in der "Einheit" KByte (K=Kilo).

Speicherkapazität

Zugriffszeit

KByte

Beachten Sie, daß das "K" im Gegensatz zu unserer alltäglichen Kilo-Angabe groß geschrieben wird. Bedeutet bei Maß- und Gewichtsangaben 1 kilo gleich 1000, so bezeichnet 1 KByte in der Computertechnik 1024 Byte. Mit 10 Adreßleitungen kann man $1024=2^{10}$ Speicherzeilen ansteuern. Ein Mikrocomputer, der über 16 Adreßleitungen verfügt, kann insgesamt 64 KByte adressieren. Dies kann auf folgende Weise veranschaulicht werden (Bild 4):

1KByte=1024 Byte

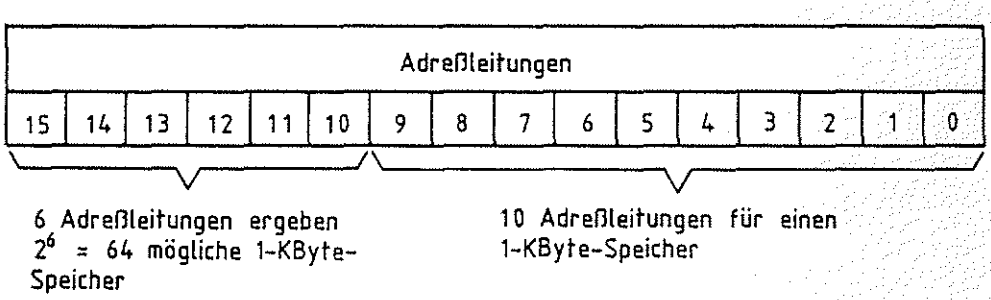


Bild 4: Verteilung von 16 Adreßleitungen auf 64 einzelne 1-KByte-Speicher

Beim Einsatz eines Mikrocomputers kommt es häufig vor, daß wesentlich kleinere Speicher als 64-KByte-Speicher für die Aufnahme der Programme und Daten benötigt werden. Außerdem sind fast immer Nur-Lese- und Schreib-Lese-Speicher erforderlich. Aus diesen und anderen Gründen werden kleinere RAM- und ROM-Module (-Einheiten) verwendet, die man in gewünschter Weise kombiniert.
Es gibt Anwendungsfälle, bei denen der Nur-Lese-Speicher (ROM) nicht größer als 1 KByte ist und 1/4-KByte-RAM ausreichend sind. Es kann auch vorkommen, daß der Speicherbereich von 64 KByte nicht ausreicht. Dann greift man auf andere Speichermedien (-Geräte/-Techniken) zurück, z.B. auf einen Magnetplattenspeicher. Das Speicherprinzip entspricht dem des Tonbandgerätes, bei dem durch Magnetisierung die Information auf einen magnetischen Träger aufgebracht wird. Beim Magnetplattenspeicher ist der Träger eine rotierende Magnetplatte. Je nach Plattensystem können Vielfache von 64 KByte

Magnetplatten-speicher

auf einer Platte gespeichert werden. Die Angabe der Speicherkapazität eines Plattensystems erfolgt bei entsprechender Größe in Mega-Byte, wobei 1 MByte gleich 1000 KByte sind. Bei diesen Systemen wird der Computer veranlaßt, immer nur den Programmteil von der Platte in den Systemspeicher zu laden, der gerade bearbeitet werden muß. Weitaus häufiger werden aber in der Mikrocomputertechnik kleine Speicher-Einheiten verwendet, die im folgenden Abschnitt behandelt werden.

MByte

1.3 Speicher-Baugruppen

Eine mögliche Unterteilung des gesamten Systemspeichers in acht gleichgroße Speicherblöcke ist im Bild 5 dargestellt. Jeder Speicherblock hat eine Kapazität von 8 KByte. Es sind auch andere Aufteilungen möglich, das angewandte Prinzip ist jedoch immer ähnlich.

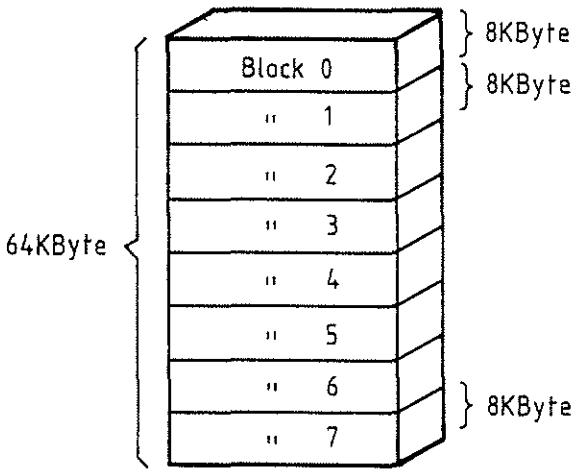


Bild 5: Aufteilung des 64-KByte-Systemspeichers in acht 8- KByte Speicherblöcke

Betrachten wir zunächst einen dieser 8-KByte-Blöcke (Bild 6). Für die Adressierung der Speicherzeilen werden die 13 Adreßleitungen A0 bis A12 benötigt. ($2^{13}=2^3 x 2^{10}=8x1024$)

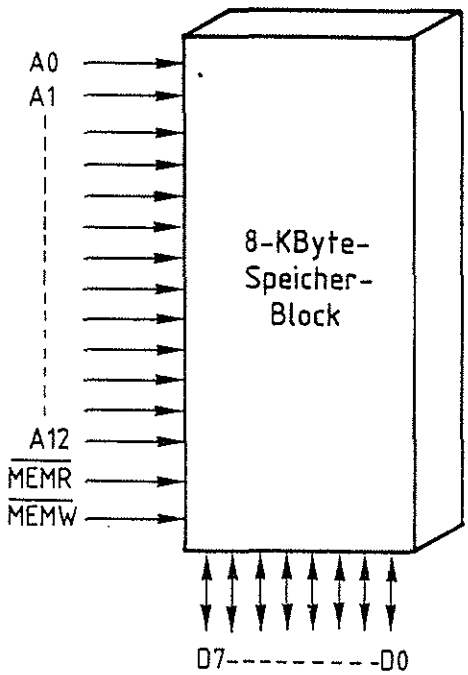


Bild 6: Die Adreßleitungen für einen 8-KByte-Speicherblock

Von den 16 Adreßleitungen des Mikroprozessors verbleiben die drei Leitungen A13, A14 und A15. Über diese drei Leitungen kann man die acht verschiedenen Blöcke unterscheiden, wenn jedem Block einer der acht möglichen Signalzustände auf diesen Leitungen zugeordnet wird. Dies ist in Bild 7 dargestellt.

Block-Nr.	Adreßleitungen			
	A15	A14	A13	A12 .... A0
0	L	L	L	für Block 0
1	L	L	H	für Block 1
2	L	H	L	für Block 2
3	L	H	H	für Block 3
4	H	L	L	für Block 4
5	H	L	H	für Block 5
6	H	H	L	für Block 6
7	H	H	H	für Block 7

Bild 7: Unterscheidung von 8 Blöcken durch die Adreßleitungen A13, A14 und A15

Block-Nummer
Baugruppen-Nummer

Die acht Signalzustände kann man als Block- oder Baugruppen-Nummer auffassen, ähnlich wie die Baugruppen-Nummer der Ein- und Ausgabe-Baugruppen. Für die Erkennung dieser Block- oder Baugruppen-Nummer muß jeder Speicherblock einen Adreßvergleicher besitzen, der den Signalzustand auf den Adreßleitungen A13, A14 und A15 mit einem vorgegebenen Signalzustand vergleicht, welcher der Block- oder Baugruppen-Nummer entspricht. Dieses Verfahren hat den Vorteil, daß alle Speicherblöcke vollkommen gleich aufgebaut und universell einsetzbar sind. Die entsprechend der Problemstellung geforderte Speichergröße eines Mikrocomputersystems kann man jetzt durch die Anzahl der Blöcke anpassen. Außerdem wird es so möglich, RAM- und ROM-Blöcke miteinander zu kombinieren.
Bild 8 zeigt, wie mehrere 8-KByte-Speicherblöcke an den Adreß-Bus des Systems angeschlossen sind.

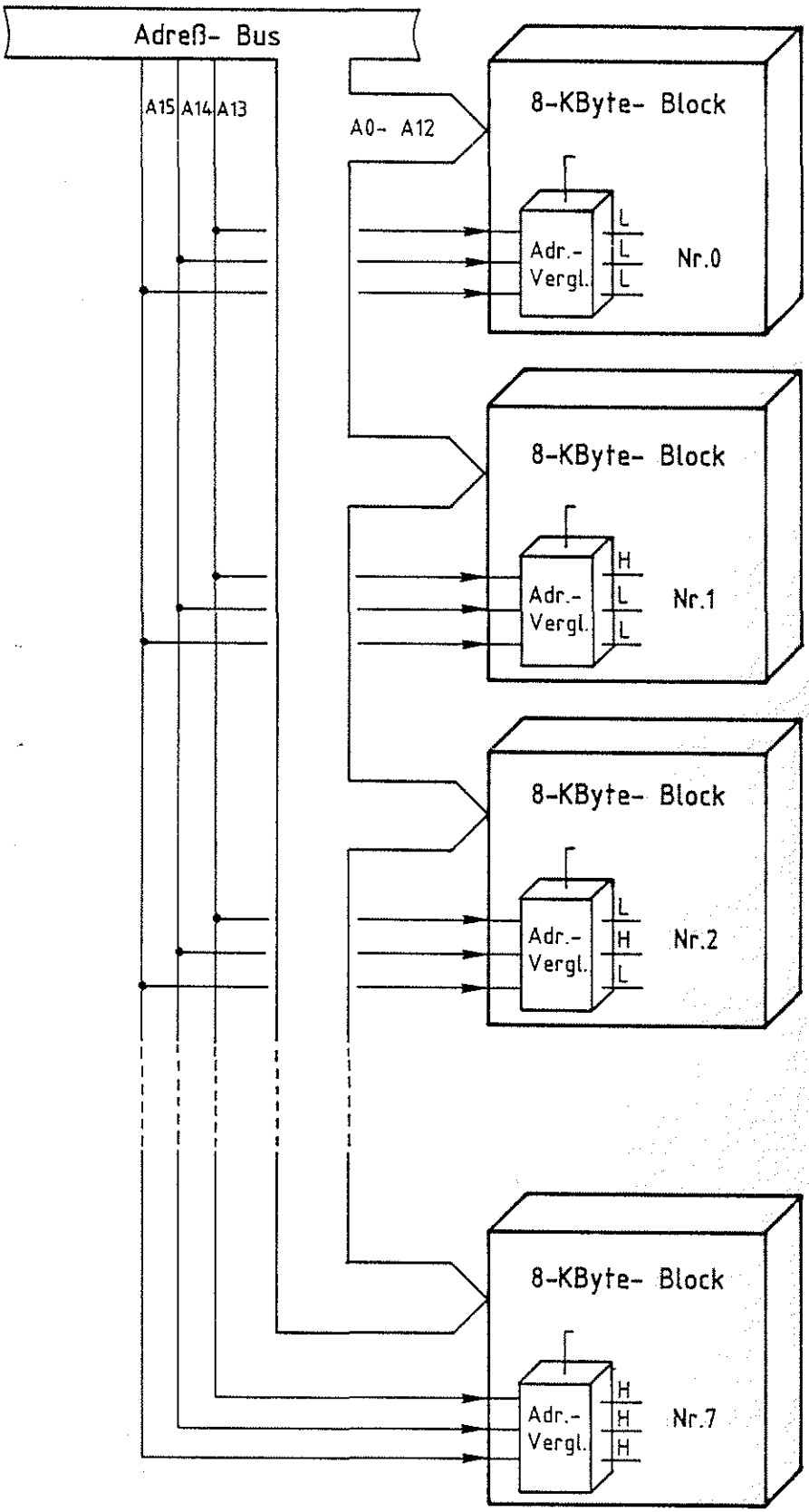


Bild 8: Aufbau eines Speichers aus mehreren 8-KByte-Speicherblöcken

Der Adreßbereich, den ein einzelner 8-KByte-Speicherblock im Systemspeicher überdeckt, hängt von der eingestellten Blocknummer ab. In Bild 9 ist für jeden der acht Blöcke die Adresse der ersten und der letzten Speicherstelle im Block aufgeführt. Die erste Adresse in einem Block nennt man auch Basis-Adresse.

Basis-Adresse

Signalzustände auf den Adreßleitungen																Adreßbereich von... bis	Block-Nr.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	0000	0
L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	1FFF	
L	L	H	L	L	L	L	L	L	L	L	L	L	L	L	L	2000	1
L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	3FFF	
L	H	L	L	L	L	L	L	L	L	L	L	L	L	L	L	4000	2
L	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	5FFF	
L	H	H	L	L	L	L	L	L	L	L	L	L	L	L	L	6000	3
L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	7FFF	
H	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	8000	4
H	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	9FFF	
H	L	H	L	L	L	L	L	L	L	L	L	L	L	L	L	A000	5
H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	BFFF	
H	H	L	L	L	L	L	L	L	L	L	L	L	L	L	L	C000	6
H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	DFFF	
H	H	H	L	L	L	L	L	L	L	L	L	L	L	L	L	E000	7
H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	FFFF	

Bild 9: Die erste (Basis-Adresse) und letzte Adresse für die acht möglichen 8-KByte-Speicherblöcke

Das folgende Bild 10 zeigt detaillierter den Aufbau einer 8-KByte-RAM-Baugruppe. Neben Adreßdecoder, Speicherblock und Adreßvergleicher sind ein Lese- und ein Schreibtor eingefügt. Das Lesetor schaltet den Inhalt einer Speicherzeile auf den Daten-Bus, sofern die Baugruppen-Nummer oder Block-Nummer an den Adreßleitungen A13 bis A15 ansteht und das Steuersignal MEMORY READ ($\overline{\text{MEMR}}$) L-Pegel führt.

Lesetor, Schreibtor

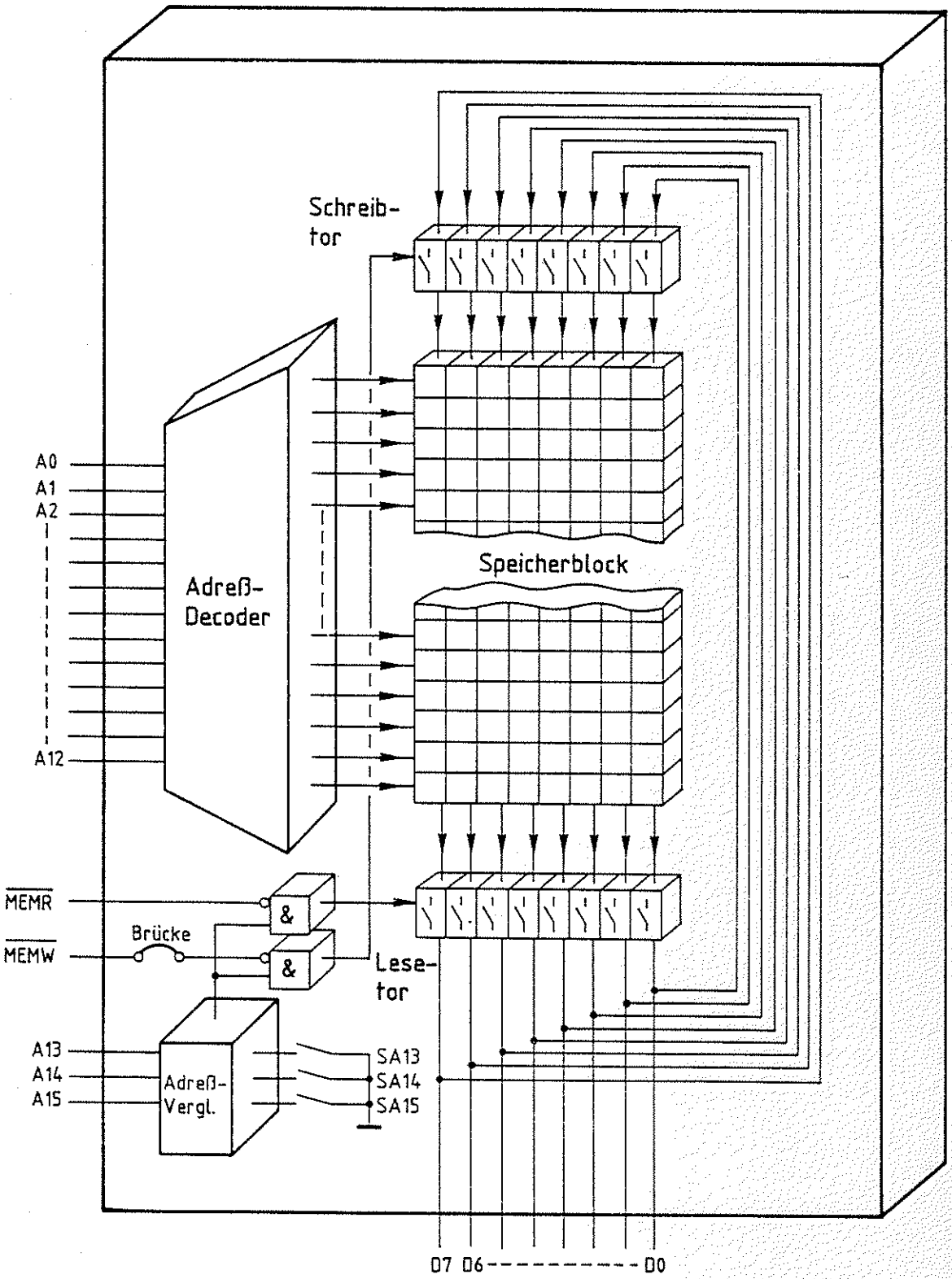


Bild 10: Aufbau einer 8-KByte-Speicherbaugruppe (8-KByte-Speicherblock)

Die Datenübernahme in eine Speicherzeile erfolgt dagegen mit der Freigabe des Schreibtors, wobei die Daten-Bus-Signale zur adressierten Speicherzeile durchgeschaltet werden und der alte Inhalt überschrieben wird. Die Schreibfreigabe wird durch den L-Pegel des Steuersignals MEMORY WRITE ($\overline{\text{MEMW}}$) ausgelöst. Voraussetzung ist, daß auf den Adreßleitungen A15, A14 und A13 die Block-Nummer ansteht.

Für die technische Realisierung dieser Baugruppe stehen zwar heute 8-KByte-Speicher-ICs zur Verfügung, jedoch verwendet man meist mehrere kleinere Speicherbausteine mit Speicherkapazitäten von 1, 2 oder 4 KByte. Falls es sich um RAM-Bausteine handelt, sind in ihnen Adreß-Decoder und Lese- und Schreibtor integriert. Einige Hersteller bieten pinkompatible RAM- und ROM-Bausteine an; sie sind bis auf das Schreibtor prinzipiell gleich aufgebaut und haben die gleiche Signalbelegung an den Anschlüssen. Damit kann man durch Austausch von RAM- und ROM-Bausteinen eine RAM-Baugruppe einfach in eine ROM-Baugruppe umwandeln. In diesem Fall wird das Steuersignal $\overline{\text{MEMW}}$ auf der Speicher-Baugruppe nicht benötigt und durch Öffnen einer Brücke unterbrochen (Bild 10).

pinkompatibel

Greift der Prozessor auf die Speicher-Baugruppe zu, so löst er die folgenden Schritte aus:
Der Prozessor...

- sendet die Adresse der gewünschten Speicherzeile auf dem Adreß-Bus aus.
- veranlaßt den Speicher, entweder durch das Steuersignal $\overline{\text{MEMR}}$, den Speicherinhalt auf den Daten-Bus zu schalten oder durch das Steuersignal $\overline{\text{MEMW}}$ den Signalzustand vom Daten-Bus in die Speicherzeile zu übernehmen.
- übernimmt beim Lesen mit dem Abschalten des Steuersignals $\overline{\text{MEMR}}$ die Daten vom Bus oder beendet den Schreibvorgang durch Wegschalten des Signals $\overline{\text{MEMW}}$.

1.4 Aufbau der Speicherzellen

Halbleiterspeicher

Wie schon erwähnt wurde, handelt es sich bei den verwendeten Speichern meist um Halbleiterspeicher, in denen die einzelnen Speicherzellen (für 1 Bit) hauptsächlich mit Dioden und Transistoren aufgebaut sind. Grundelement einer RAM-Speicherzelle z.B. ist eine bistabile Kippstufe, wie in Bild 11 dargestellt.

Speicherzelle

Bistabile Kippstufe (Flipflop)

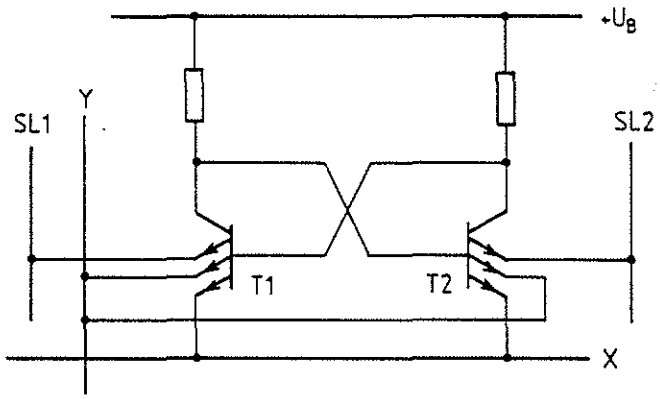


Bild 11: Prinzipieller Aufbau einer RAM-Zelle

Die beiden stabilen Zustände sind dadurch gekennzeichnet, daß entweder Transistor T1 leitet und T2 gesperrt ist oder T1 sperrt und T2 leitet. Den beiden Zuständen wird der Speicherinhalt "0" und "1" zugeordnet. Über die Leitungen X und Y wird die gewünschte Speicherzelle adressiert, über die Leitungen SL1 und SL2 (Schreib-Lese-Leitungen) kann der Inhalt der Speicherzelle gelesen oder ein neuer eingeschrieben werden.

Wegen des einfachen Herstellungsprozesses werden die Speicherzellen meist mit MOS-Transistoren aufgebaut, die außerdem höhere Packungsdichten ermöglichen und geringere Verlustleistung haben. Eine noch höhere Packungsdichte und geringere Verlustleistung erhält man bei einer Technik, bei der die Information in einem kleinen Kondensator gespeichert wird, und zwar in der Gate-Source- oder Gate-Drain-Kapazität eines MOS-Transistors. Das Prinzip dieser Speicherzelle zeigt Bild 12.

MOS-Transistor

Kondensator

Gate-Source-

Gate-Drain-

Kapazität

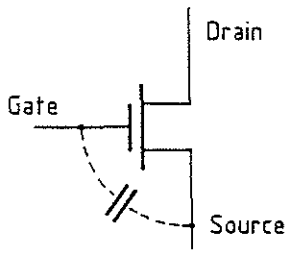


Bild 12: Prinzip der kapazitiven Informationsspeicherung in einem MOS-Transistor

Kapazitive Informationsspeicherung

Die beiden möglichen Signalzustände der Speicherzelle werden hier dem geladenen bzw. entladenen Kondensator zugeordnet. Bei dieser Speichertechnik ist allerdings eine sogenannte Auffrischschaltung erforderlich. Sie muß den ständigen Ladungsverlust der Kondensatoren ausgleichen, der aufgrund unvermeidbarer Leckströme in der Isolierschicht auftritt. Schaltungen dieser Art nennt man im englischen "refresh circuit".

Auffrischschaltung
refresh circuit

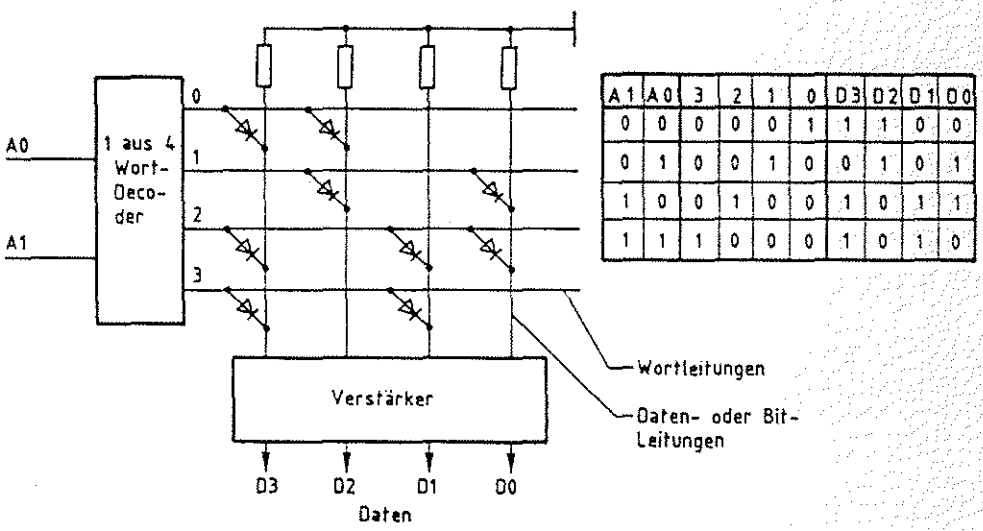
Weil ständig für diesen Auffrischvorgang auf die Speicherzellen zugegriffen werden muß (etwa alle 2msec), nennt man diese Speicher "dynamische RAMs" (Dynamik, Bewegung) im Gegensatz zu den Flipflop-Zellen, die man entsprechend "statische RAMs" (Statik, Ruhe) nennt.

Dynamisches RAM

Statisches RAM

Das Prinzip eines ROMs ist die Diodenmatrix (Bild 13). Jeder Kreuzungspunkt der horizontalen und vertikalen Leitungen der Matrix stellt eine Speicherzelle dar.

ROM
Diodenmatrix



A1	A0	3	2	1	0	D3	D2	D1	D0
0	0	0	0	0	1	1	1	0	0
0	1	0	0	1	0	0	1	0	1
1	0	0	1	0	0	1	0	1	1
1	1	1	0	0	0	1	0	1	0

Bild 13: Prinzip eines ROM-Speichers
Kapazität: 4x4Bit (4 Worte a' 4Bit)

Wort-Decoder

Wortleitung

Datenleitung

Der Wort-Decoder ist so aufgebaut, daß abhängig vom Signalzustand auf den Adreßleitungen, nur einer der Decoder-Ausgänge 1-Signal haben kann. Die ausgewählte Wortleitung legt nun alle Datenleitungen, die mit ihr über Dioden verbunden sind, auf H-Pegel. Die nicht mit ihr verbundenen Datenleitungen erhalten über die Widerstände L-Pegel. Die Tabelle in Bild 13 zeigt den ROM-Inhalt, abhängig von den möglichen Adressen. Die Programmierung des Speichers erfolgt also dadurch, daß während seiner Herstellung an gewünschten Kreuzungspunkten zwischen Wort- und Datenleitungen (oft auch Bit-Leitungen genannt) Dioden eingebaut werden. Je nach Herstellungsverfahren werden anstelle der Dioden auch Widerstände, bipolare Transistoren und MOS-Transistoren verwendet.

Programmierung des ROMs

1.5 Bausteine für Festwertspeicher (ROMs)

maskenprogrammiert

Der gewünschte ROM-Inhalt wird in einem der letzten Produktionsschritte in die Bausteine eingebracht. Diese Bausteine heißen "maskenprogrammierbare ROMs". Der Name "maskenprogrammiert" rührt von den Fertigungsschritten her, die sogenannte Masken für die Diffusionsprozesse erfordern. Dieses Verfahren ist teuer und wird nur bei hohen Stückzahlen angewandt. Außerdem kann ein fehlerhafter Inhalt nicht korrigiert werden. Nachteilig kommt hinzu, daß die notwendigen Fertigungseinrichtungen nicht jedem zugänglich sind. Daher hat man sogenannte PROM-Bausteine entwickelt. PROM heißt programmierbares ROM und bedeutet, daß der Anwender durch einen speziellen Vorgang die gewünschte Information in das ROM einbringen (programmieren) kann. Dazu liefert der Hersteller ROM-Bausteine, bei denen alle Speicherstellen entweder nur "0"- oder "1"-Informationen enthalten. In diesem Fall spricht man von unprogrammierten PROMs. Auch diese Bausteine haben den Nachteil, daß sie nicht wieder verwendbar sind, wenn eine falsche Information einprogrammiert wurde.

PROM-Bausteine

Fehler treten aber häufig in der Entwicklungsphase eines Programms auf, so daß man für den Anwender wiederverwendbare PROMs entwickelt hat, die man programmieren und wieder löschen kann. Diese Bausteine heißen EPROMs, d.h. lösch- (erasable) und programmierbare ROMs. Zum Löschen werden die ICs, die ein Quarzglasfenster besitzen, mit UV-Licht beleuchtet. Mittlerweile gibt es auch elektrisch löschbare (electrical erasable) und programmierbare ROMs, die man entsprechend EEPROM nennt.

EPROM

EEPROM

1.6 RAM-Speicher als Festwertspeicher

Schreib- Lese-Speicher (RAM) zieht man in Mikrocomputer-Systemen den Festwertspeichern immer dann vor, wenn man in der Programm-Entwicklungsphase das Programm und damit den Speicherinhalt ständig ändern muß.
Will man den Speicherinhalt von RAM-Zellen über längere Zeit erhalten, so muß man sie ständig mit Spannung versorgen, denn die Kondensatoren in dynamischen RAMs entladen sich, und Fliflops nehmen beim Einschalten der Spannung die Zustände 0 oder 1 in zufälliger Weise an. Zur Konservierung der Speicherinhalte verwendet man Batterien und spricht von Batteriepufferung und gepufferten RAMs.

gepuffertes RAM

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

SPEICHER-EINHEITEN

BFZ/MFA 10.3.

ÜBUNGSTEIL 1

Übungsteil 1

In den folgenden Arbeitsschritten werden Sie Messungen an einer RAM- und an einer ROM-Baugruppe durchführen.

Dazu benötigen Sie:

- 1 Baugruppenträger mit Busverdrahtung (BFZ/MFA 0.1.)
- 1 Bus-Abschluß (BFZ/MFA 0.2.)
- 1 Trafo-Einschub (BFZ/MFA 1.1.)
- 1 Spannungsregelung (BFZ/MFA 1.2.)

} zusammengebaut und geprüft nach FPÜ BFZ/MFA 1.2. Arbeitsblatt A7

- 1 Bus-Signalgeber (BFZ/MFA 5.1.)
- 1 Bus-Signalanzeige (BFZ/MFA 5.2.)
- 1 8-K-RAM/EPROM bestückt mit RAM (BFZ/MFA 3.1.)
- 1 8-K-RAM/EPROM bestückt mit EPROM MAT 85 (BFZ/MFA 3.1.)

Allgemeine Hinweise zur Durchführung der Übungen:

- Die Einschübe dürfen nur bei abgeschalteter Betriebsspannung gesteckt oder gezogen werden
- Aufgrund der Busverdrahtung können die Baugruppen in beliebige Steckplätze gesteckt werden
- Den logischen Signalen "0" und "1" sind die folgenden Pegel zugeordnet:

  log. "0" ≙ 0...0,8 V (LOW)

  log. "1" ≙ 2,4...5 V (HIGH)

- Alle zur Messung an den Baugruppen vorgegebenen Arbeitsblätter enthalten:
  = Angaben über den Sinn der jeweiligen Messung
  = Angaben über einzustellende Bedingungen (z.B. Schalterstellungen)
  = Aufgabenstellungen, ggf. mit Hinweisen zu möglichen Fehlern.

Bedienungshinweise:

8-K-RAM/EPROM-Baugruppe als RAM-Baugruppe:

- Die IC-Sockel IC8, IC9, IC10, IC11 müssen mit RAM-Bausteinen 6116P-3 bestückt sein
- Wenn keine vier Bausteine bestückt sind, müssen Sie mindestens einen Baustein in den Sockel IC8 stecken
- Die Lötpunkte P3-P4 und P5-P6 müssen gebrückt sein
- Mit den Schaltern SA13, SA14 und SA15 muß die Baugruppen-Nummer 0 eingestellt werden (siehe dazu folgende Seite)

8-K-RAM/EPROM-Baugruppe als EPROM-Baugruppe:

- Die IC-Sockel IC8 bis IC11 müssen mit EPROM-Bausteinen 2716 bestückt sein, die entsprechend der folgenden Tabelle beschriftet sind:

Sockel	IC11	IC10	IC9	IC8
beschriftet mit...	V1.8 IV	V1.8 III	V1.8 II	V1.8 I

- Die Lötpunkte P1-P2 und P7-P8 müssen gebrückt sein
- Die Baugruppen-Nummer 0 muß eingestellt sein

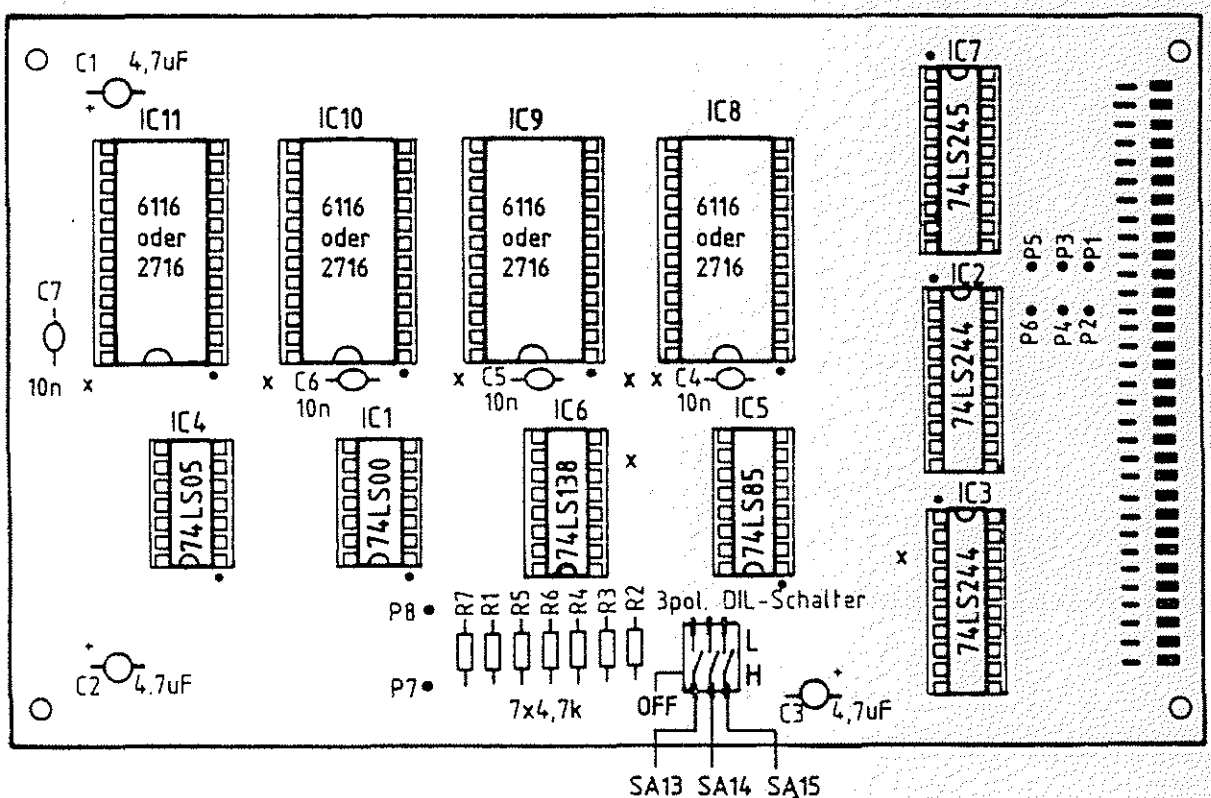


Tabelle zur Einstellung der Baugruppen-Nummern bzw. der Basis-Adressen der Speicher-Baugruppen.

Schalter offen ≙ H
Schalter zu ≙ L

SA15	SA14	SA13	Baugruppen-Nummer	Adreßbereich (H) von	bis
L	L	L	0	0000	1FFF
L	L	H	1	2000	3FFF
L	H	L	2	4000	5FFF
L	H	H	3	6000	7FFF
H	L	L	4	8000	9FFF
H	L	H	5	A000	BFFF
H	H	L	6	C000	DFFF
H	H	H	7	E000	FFFF

Basis-Adressen

Speicher-Einheiten Name:

Übungsteil 1 Datum:

A1

Überprüfen des Inhalts von RAM-Speicherzeilen

Die dazu notwendigen Adreß- und Steuersignale liefert der Bus-Signalgeber. Adressen und Speicherinhalte werden auf der Bus-Signalanzeige angezeigt.
Stecken Sie Bus-Signalgeber, Bus-Signalanzeige und RAM-Baugruppe in den Baugruppenträger und schalten Sie die Betriebsspannung ein.
Überprüfen Sie den Inhalt der ersten Speicherstellen im RAM, indem Sie nacheinander ab der Adresse 0000H einige Speicherstellen adressieren und das Steuersignal $\overline{\text{MEMR}}$ auslösen. Notieren Sie die Speicherinhalte und vergleichen Sie diese mit den Speicherinhalten, die Ihr Nachbar ermittelt hat. Die Inhalte sind zufällig und werden nicht übereinstimmen. Lesen Sie auch mehrmals eine gleiche Speicherstelle; der Inhalt darf sich nicht verändern.

Adresse	Inhalt
0000	
0001	
0002	
0003	
0004	
0005	
0006	
0007	
0008	
0009	
000A	

Speicher-Einheiten

Name:

Übungsteil 1

Datum:

A2

Eingabe eines kleinen Programms in den RAM-Speicher und Überprüfen des Speicher-Inhalts nach kurzzeitigem Abschalten der Betriebsspannung.

Übergeben Sie der RAM-Baugruppe ab der Speicheradresse 0000H die untenstehenden Daten. Sie laden damit ein Programm ins RAM, das in einer der späteren Übungen benötigt wird. Es veranlaßt den Prozessor fortwährend, den Signalzustand an den Eingangsleitungen einer Eingabe-Baugruppe zu lesen und ihn anschließend an einer Ausgabe-Baugruppe anzuzeigen.

Adresse	Inhalt
0000	DB
0001	01
0002	D3
0003	02
0004	C3
0005	00
0006	00

Kontrollieren Sie die im RAM gespeicherten Daten anschließend durch Lesen der Speicherstellen.

Schalten Sie nun kurz die Betriebsspannung aus und ermitteln Sie die Daten der Speicherstellen 0000H bis 0006H. Vergleichen Sie diese mit den zuvor eingegebenen Daten.

Adresse	Inhalt nach erneutem Einschalten der Betriebsspannung	Erkenntnis:
0000		
0001		
0002		
0003		
0004		
0005		
0006		



Speicher-Einheiten

Name:

Übungsteil 1

Datum:

A3

Lesen des Inhalts von ROM-Speicherzeilen bei verschiedenen Basisadressen der Baugruppe.

Entfernen Sie nun die RAM-Baugruppe und stecken Sie an ihre Stelle die EPROM-Baugruppe. (Nicht beide Baugruppen wegen der gleichen Baugruppen-Nummer gemeinsam betreiben!)
Ermitteln Sie den Inhalt einiger Speicherstellen im ROM ab der Adresse 0000H und tragen Sie die Daten in untenstehende Tabelle ein. Lesen Sie die Speicherinhalte nach kurzzeitigem Abschalten der Betriebsspannung erneut. Vergleichen Sie die Daten mit den Tabellenwerten. Versuchen Sie durch einen Schreibvorgang den Inhalt einiger Speicherstellen zu überschreiben.

Adresse	Inhalt	Inhalt nach erneutem Einschalten der Betriebsspannung
0000		
0001		
0002		
0004		
Erkenntnis:		

Stellen Sie nun die Baugruppen-Nummer 5 ein. Überprüfen Sie, ob die in obiger Aufgabe ermittelten Daten jetzt unter den Adressen des Blocks 5 (ab A000H) wiederzufinden sind. Kontrollieren Sie auch die Speicherstellen ab der Adresse 0000H.

Adresse	Inhalt	Erkenntnis:
A000		
A001		
A002		
A003		
A004		

Ende Übungsteil 1

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

SPEICHER-EINHEITEN

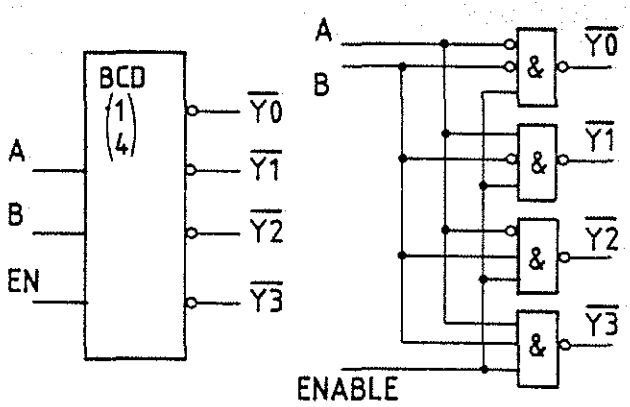
BFZ/MFA 10.3.

THEORIETEIL 2

2.1 Adreßdecodierung

Ebenso wie man den Systemspeicher eines Computers aus mehreren Speicherblöcken aufbaut, wird ein einzelner Block meist aus mehreren kleineren Speicher-Bausteinen zusammengesetzt. Auf der 8-K-RAM/EPROM-Baugruppe BFZ/MFA 3.1. besteht der 8-KByte-Block aus vier 2-KByte-Speicher-Bausteinen. Der technische Aufbau eines solchen Blocks ergibt sich aus einer ähnlichen Überlegung, wie sie im Teil 1 ausgeführt wurde:
Für einen 2-KByte-Speicher werden 11 Adreßleitungen für die Adressierung der 2048 Speicherstellen benötigt ($2^{11}=2^1 x 2^{10}=$ 2x1024. Von den 13 Adreßleitungen, die der 8-KByte-Speicherbaugruppe zugeführt werden, bleiben zwei Leitungen übrig (A11 und A12). Aus den vier möglichen Signalzuständen auf diesen beiden Adreßleitungen lassen sich vier Freigabesignale für die vier 2-KByte-Bausteine ableiten. Dafür verwendet man wieder einen (Adreß-)Decoder. Ein solcher Decoder ist mit seiner Innenschaltung in Bild 14 dargestellt.

8-K-Speicher-Block aufgebaut aus 4 2-K-Bausteinen



Eingänge			Ausgänge			
EN	B	A	$\overline{Y0}$	$\overline{Y1}$	$\overline{Y2}$	$\overline{Y3}$
H	L	L	L	H	H	H
	L	H	H	L	H	H
	H	L	H	H	L	H
	H	H	H	H	H	L
L	X	X	alle H			

X= beliebig (H oder L)

Bild 14: Schaltung und Funktionstabelle eines 1 aus 4- Decoders

Entsprechend dem Signalzustand an den Eingängen A und B wird immer ein Ausgang auf L-Signal (Aktiv Low-Signal) geschaltet. Über einen Freigabe-Eingang (ENABLE) können außerdem alle Ausgänge gesperrt werden. Dieser Decoder wird 1-aus 4-Decoder genannt, weil einer von vier möglichen Zuständen an den Ausgängen angezeigt wird.

1 aus 4-Decoder

Entsprechend gibt es auch 1 aus 8- und 1 aus 16-Decoder. Die Schaltung eines 8-KByte-Blocks unter Verwendung eines Adreßdecoders ist in Bild 15 dargestellt.

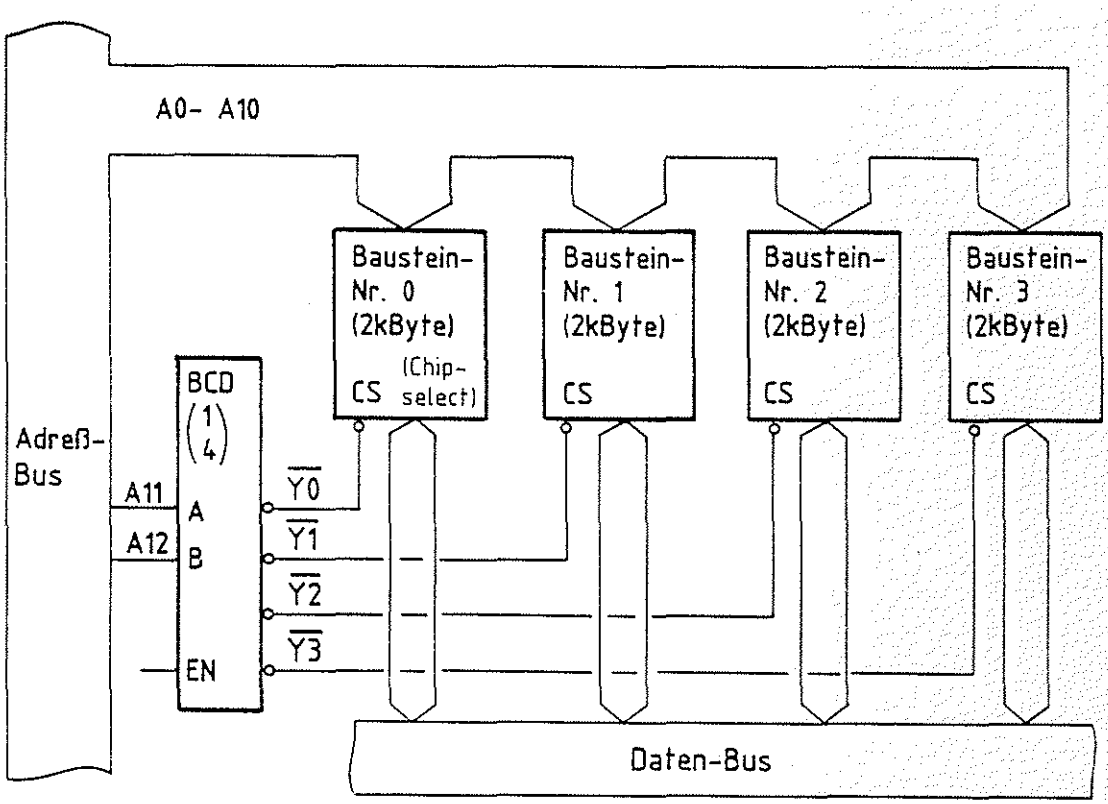


Bild 15: 8-KByte-Speicherblock aufgebaut mit vier 2-KByte-Speicher-Bausteinen

Die Adreßleitungen A0 bis A10 werden an alle Speicher-Bausteine angeschlossen. Über den Signalzustand auf diesen Leitungen wird in den Bausteinen eine ganz bestimmte Speicherstelle ausgewählt. Welcher Baustein schließlich Daten senden oder empfangen darf, wird vom Signalzustand auf den Adreßleitungen A11 und A12 bestimmt. Die Freigabesignale, die dazu vom Adreßdecoder geliefert werden, sind mit den sogenannten "Chip-Select-Eingängen" (chip-select = Bausteinauswahl) der Speicher-Bausteine verbunden. Diese Eingänge sind speziell für die hier geschilderte Anwendung vorgesehen. In der Tabelle Bild 16 sind die Adreßbereiche angegeben, die von den vier Bausteinen überdeckt werden. Hier wurde von der Baugruppe 0 bzw. der Basis-Adresse 0000 ausgegangen. Unberücksichtigt bleiben in dieser Tabelle

Chip-Select (CS)

die Adreßleitungen A13 bis A15, über die die Baugruppen-Nummer bzw. die Basis-Adresse festgelegt wird.

Adreßleitung													Adresse	Baustein
12	11	10	9	8	7	6	5	4	3	2	1	0		
L	L	L	L	L	L	L	L	L	L	L	L	L	0000	0
L	L	H	H	H	H	H	H	H	H	H	H	H	07FF	
L	H	L	L	L	L	L	L	L	L	L	L	L	0800	1
L	H	H	H	H	H	H	H	H	H	H	H	H	0FFF	
H	L	L	L	L	L	L	L	L	L	L	L	L	1000	2
H	L	H	H	H	H	H	H	H	H	H	H	H	17FF	
H	H	L	L	L	L	L	L	L	L	L	L	L	1800	3
H	H	H	H	H	H	H	H	H	H	H	H	H	1FFF	

Bild 16: Baustein-Adreßbereiche der vier Speicher-Bausteine der 8-KByte-Speicherbaugruppe

Das Anschlußbild eines 2-KByte-Speicher-Bausteins ist in Bild 17 dargestellt. Es handelt sich dabei um einen RAM-Baustein. RAM- und ROM-Bausteine unterscheiden sich nur durch die Write Enable-Leitung (Write Enable = Schreiben ermöglichen), die beim ROM-Speicher fehlt. Mit einem L-Signal auf dieser Leitung wird ein Datenwort in den Speicher geschrieben. Da die Speicher-Bausteine - ebenso wie die Eingabe-Baugruppen - Daten auf den Daten-Bus schalten, werden ihre Datenleitungen über Tristate-Gatter hinausgeführt. Der Chip-Select-Eingang ist bei diesen Bausteinen intern direkt mit dem ENABLE-Eingang der Tristate-Gatter verbunden.

Write-Enable-Leitung

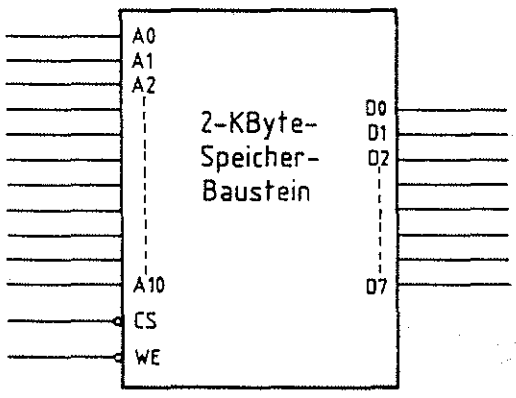


Bild 17: Anschlußbild eines 2-KByte-Speicher-Bausteins

2.2 Vereinfachte Schaltung einer Speicher-Baugruppe

Die Schaltung der Speicher-Baugruppe ist in Bild 18 abgebildet. Zur Verbesserung der Übersichtlichkeit sind die Daten-Bus-Leitungen vereinfacht dargestellt und einige auf der Baugruppe vorhandene Schaltungsdetails weggelassen worden.

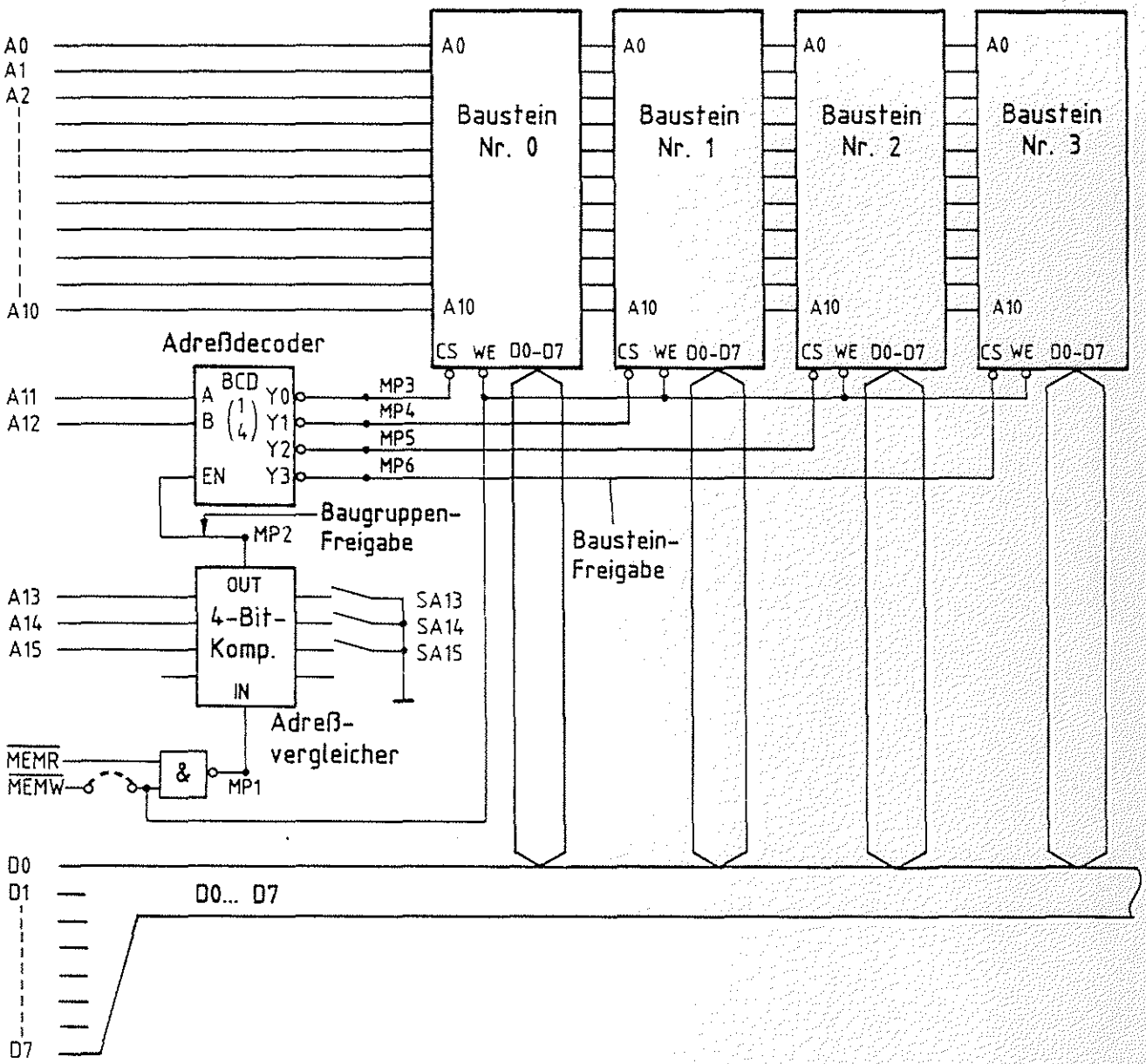


Bild 18: Vereinfachtes Schaltbild der Speicherbaugruppe

Der Adreßvergleicher zur Erkennung der Baugruppen-Nummer bzw. der Basis-Adresse auf den Adreßleitungen A13 bis A15 entspricht dem Adreßvergleicher auf den Ein- und Ausgabe-Baugruppen. Allerdings reicht hier ein 4-Bit-Komparator aus. Der Kaskadiereingang "IN" des Komparators erhält immer dann

H-Signal, wenn entweder $\overline{MEMR}$ oder $\overline{MEMW}$ L-Signal führen, d.h. der Prozessor den Speicher aktiviert. Der Ausgang des Adreßvergleichers (OUT) ist mit dem Freigabe-Eingang (ENABLE) des Adreßdecoders verbunden.

Bedingungen für Schreiben oder Lesen

Ein Speicher-Baustein wird nur dann freigegeben, wenn

- die Baugruppe über A13 bis A15 ausgewählt wird,
- das Steuersignal $\overline{MEMR}$ oder $\overline{MEMW}$ aktiviert wird und
- die entsprechende Baustein-Nummer an den Adreß-Leitungen A11 und A12 ansteht.

Handelt es sich um eine RAM-Baugruppe, so wird bei einem Schreibvorgang die WE-Leitung an den Bausteinen auf L-Signal geschaltet. Bei einer ROM-Baugruppe wird das Steuersignal $\overline{MEMW}$ durch eine Brücke von der Baugruppe getrennt. Die Signal- Zeit-Diagramme für einen Lese- und einen Schreibvorgang durch den Prozessor sind in den Bildern 19 und 20 dargestellt.

Zeitdiagramm Lesevorgang

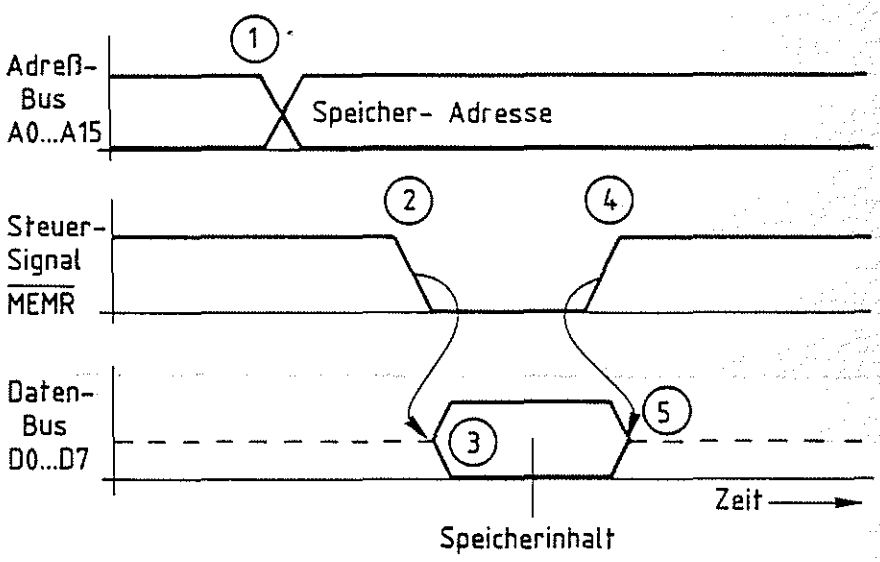


Bild 19: Lesen einer Speicherzeile

Lesevorgang:

Zum Zeitpunkt 1 sendet der Prozessor die Speicheradresse auf den Adreßleitungen A0 bis A15 aus. Zum Zeitpunkt 2 veranlaßt der Prozessor den Speicher, den Inhalt der adressierten Speicherzeile auf den Daten-Bus zu schalten (Zeit-

punkt 3). Mit dem L-H-Signalwechsel auf der Steuerleitung $\overline{\text{MEMR}}$ übernimmt der Prozessor den Signalzustand vom Daten-Bus und beendet gleichzeitig den Lesevorgang, so daß der Speicher die Daten vom Bus wegschaltet (Zeitpunkt 5). Der Vorgang ist identisch mit dem Lesevorgang einer Eingabe-Baugruppe.

Zeitdiagramm Schreibvorgang

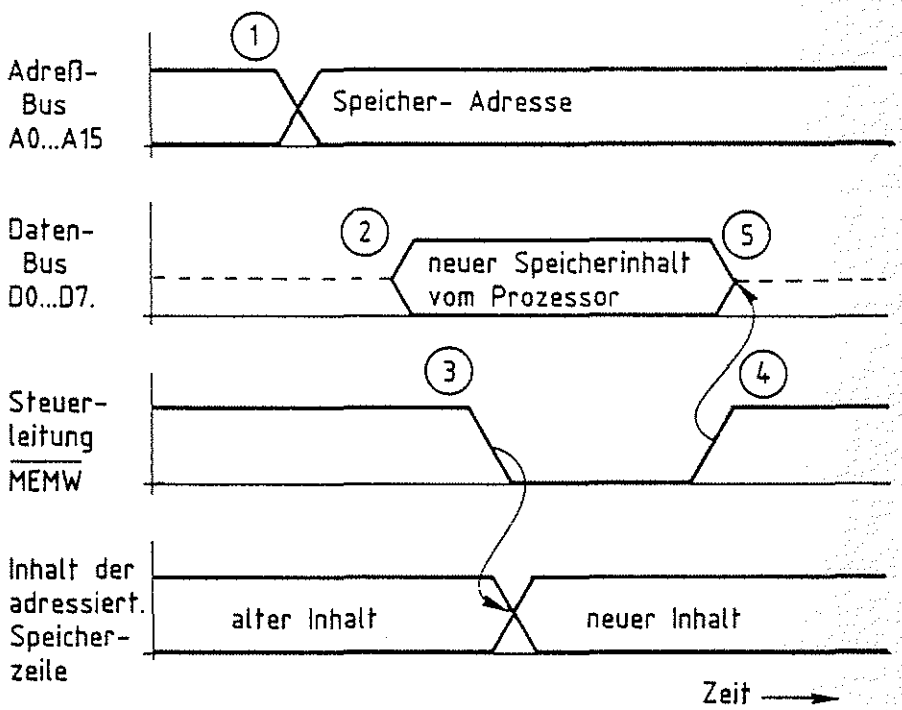


Bild 20: Schreiben in eine Speicherzeile

Schreibvorgang:
Zum Zeitpunkt 1 sendet der Prozessor die Speicheradresse auf den Adreßleitungen A0 bis A15 aus. Zum Zeitpunkt 2 stellt er den neuen Speicherinhalt am Daten-Bus bereit. Mit dem Steuersignal $\overline{\text{MEMW}}$ (Zeitpunkt 3) löst er die Datenübernahme in die adressierte Speicherzeile aus. Der Schreibvorgang wird mit H-Signal auf der Steuerleitung $\overline{\text{MEMW}}$ beendet (Zeitpunkt 4), so daß der Prozessor gleichzeitig die Daten vom Bus wegschalten kann (Zeitpunkt 5). Auch dieser Ablauf ist identisch mit einer Datenübergabe an eine Ausgabe-Baugruppe.

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

SPEICHER-EINHEITEN

BFZ/MFA 10.3.

ÜBUNGSTEIL 2

Übungsteil 2

In den folgenden Arbeitsschritten werden Sie auf der Speicher-Baugruppe die Signale für...

- die Freigabe der Baugruppe und
- die Auswahl der Speicher-Bausteine (CS)

meßtechnisch überprüfen.

Dazu benötigen Sie:

- 1 Baugruppenträger mit Busverdrahtung (BFZ/MFA 0.1.)
- 1 Bus-Abschluß (BFZ/MFA 0.2.)
- 1 Trafo-Einschub (BFZ/MFA 1.1.)
- 1 Spannungsregelung (BFZ/MFA 1.2.)

} zusammengebaut und geprüft nach FPÜ BFZ/MFA 1.2. Arbeitsblatt A7

- 1 Bus-Signalgeber (BFZ/MFA 5.1.)
- 1 Bus-Signalanzeige (BFZ/MFA 5.2.)
- 1 8-K-RAM/EPROM bestückt mit RAM (BFZ/MFA 3.1.)
- 1 Adapter 64polig (BFZ/MFA 5.3.)
- 1 Logiktester oder Vielfachmeßinstrument
- 2 Meßleitungen

Allgemeine Hinweise zur Durchführung der Übungen:

- Die Einschübe dürfen nur bei abgeschalteter Betriebsspannung gesteckt oder gezogen werden
- Aufgrund der Busverdrahtung können die Baugruppen in beliebige Steckplätze gesteckt werden
- Den logischen Signalen "0" und "1" sind die folgenden Pegel zugeordnet:

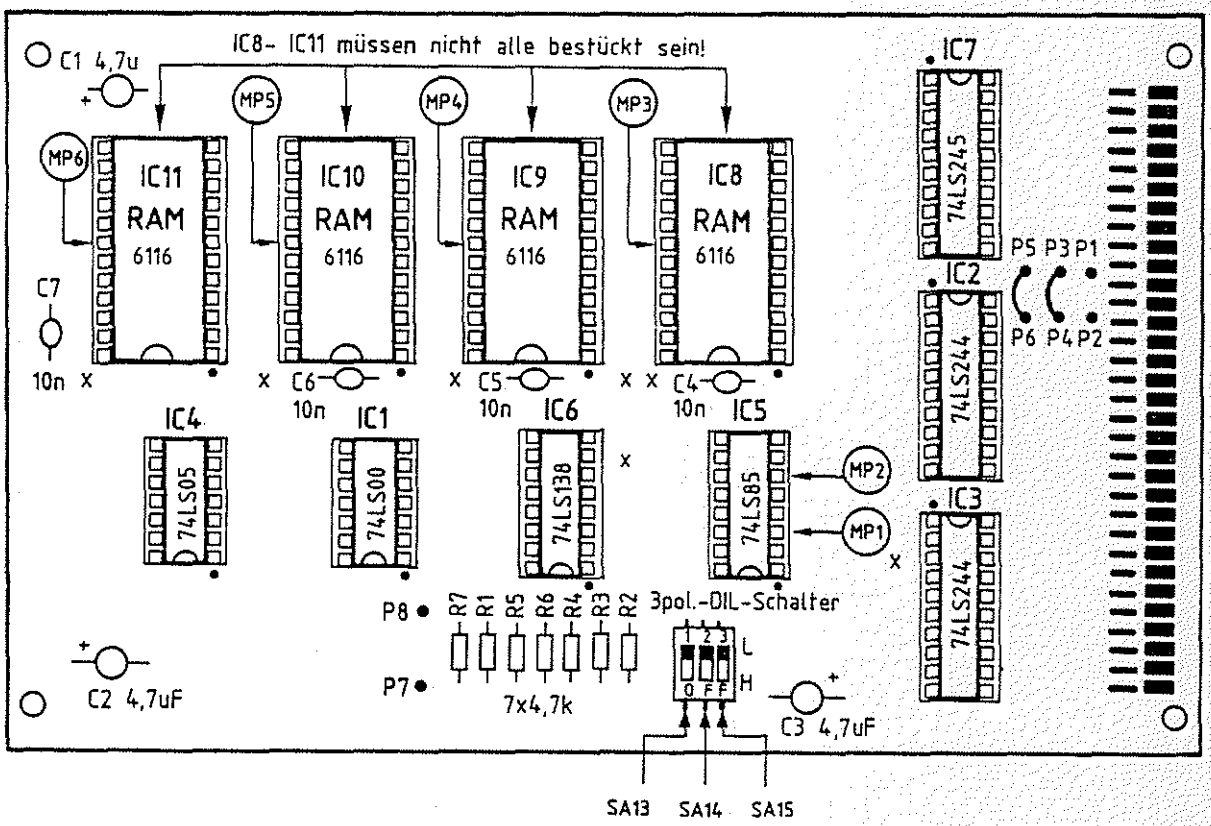
  log. "0" $\hat{=}$ 0...0,8 V (LOW)
  log. "1" $\hat{=}$ 2,4...5 V (HIGH)

- Alle zur Messung an den Baugruppen vorgegebenen Arbeitsblätter enthalten:
  = Angaben über den Sinn der jeweiligen Messung
  = Angaben über einzustellende Bedingungen (z.B. Schalterstellungen)
  = Aufgabenstellungen, ggf. mit Hinweisen zu möglichen Fehlern.

Bedienungshinweise:

8-K-RAM/EPROM-Baugruppe als RAM-Baugruppe:

Im unten abgebildeten Bestückungsplan der RAM-Baugruppe sind die in Bild 17 (vereinf. Schaltbild) eingetragenen Meßpunkte MP1 bis MP6 gekennzeichnet. Benutzen Sie dieses Schaltbild auch während der Messungen.



MP1 — Verknüpfung der Steuersignale $\overline{MEMR}$ u. $\overline{MEMW}$
MP2 — Baugruppen- Freigabe
MP3 — Bausteinfreigabe Nr.0 (CS0)
MP4 — " Nr1 (CS1)
MP5 — " Nr2 (CS2)
MP6 — " Nr3 (CS3)

Speicher-Einheiten

Name:

Übungsteil 2

Datum:

A1

Prüfen der Wirkung der Steuersignale am Eingang des Adreß-Vergleichers (MP1)

Stellen Sie mit SA13 - SA15 die Baugruppen-Nummer 0 ein (Basis-Adresse 0000H). Stecken Sie die Speicher-Baugruppe über die Adapter-Karte neben den Bus-Signalgeber und die Bus-Signalanzeige in den Baugruppenträger ein. Schalten Sie die Betriebsspannung ein.
Überprüfen Sie mit einem Logiktester oder Meßgerät die Wirkung der Steuersignale $\overline{\text{MEMR}}$ und $\overline{\text{MEMW}}$ am Eingang des Adreßvergleichers (MP1), indem Sie während der Messung die Steuersignale (Bus-Signalgeber) abwechselnd betätigen (nicht gleichzeitig).

Steuersignaltaste...		Pegel am MP1
MEMR	nicht betätigt	
	betätigt	
MEMW	nicht betätigt	
	betätigt	



Speicher-Einheiten

Name:

Übungsteil 2

Datum:

A2.1

Prüfen des Freigabesignals für die Speicher-Baugruppe (MP2)

Prüfen Sie das Freigabesignal für die Speicher-Baugruppe am Meßpunkt MP2, wenn Sie nacheinander am Bus-Signalgeber Adressen einstellen, die im und außerhalb des Adreßbereiches der Baugruppe liegen. Tragen Sie den Signalzustand in die Tabelle ein.

Schalter SA15	Schalter SA14	Schalter SA13	Baugruppen-Nummer	Adreßbereich von	Adreßbereich bis	Pegel am Meßpunkt MP2
L	L	L	0	0000	1FFF	
			1	2000	3FFF	
			2	4000	5FFF	
			3	6000	7FFF	
			4	8000	9FFF	
			5	A000	BFFF	
			6	C000	DFFF	
			7	E000	FFFF	

Speicher-Einheiten

Name:

Übungsteil 2

Datum:

A2.2

Stellen Sie mit den Schaltern SA13 bis SA15 die Baugruppen-Nummer 5 ein und wiederholen Sie die vorangegangenen Messungen.

Schalter SA15	Schalter SA14	Schalter SA13	Baugruppen-Nummer	Adreßbereich von	Adreßbereich bis	Pegel am Meßpunkt MP2
			0	0000	1FFF	
			1	2000	3FFF	
			2	4000	5FFF	
			3	6000	7FFF	
			4	8000	9FFF	
H	L	H	5	A000	BFFF	
			6	C000	DFFF	
			7	E000	FFFF	

Speicher-Einheiten Name:

Übungsteil 2 Datum:

A3

Prüfen der Baustein-Freigabe-Signale an den CS-Eingängen der Speicher-Bausteine

Stellen Sie die Baugruppen-Nummer auf 0.

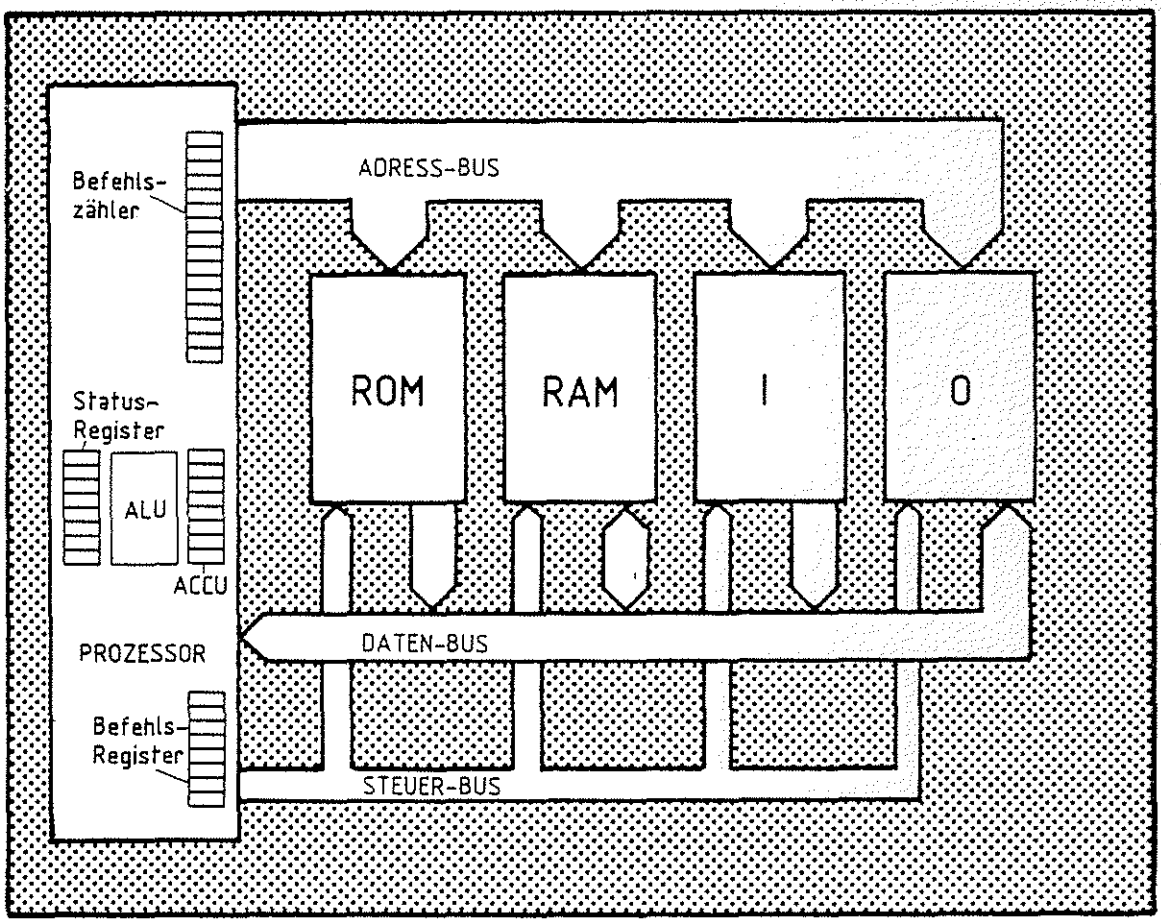
Stellen Sie nacheinander die unten aufgeführten Adressen mit dem Bus-Signalgeber ein und überprüfen Sie bei jeder Einstellung die vier Baustein-Freigabesignale (Chip Select) an den Meßpunkten MP3 bis MP6.

Baustein-Nummer	Einzustellende Adressen (Anfangs-und End-Adr. jedes Bausteins)	Baustein-Freigabesignale			
		(MP3 (CS0)	MP4 (CS1)	MP5 (CS2)	MP6 (CS3)
0	0000, 07FF				
1	0800, 0FFF				
2	1000, 17FF				
3	1800, 1FFF				

Ende der Übung.

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

MIKROPROZESSOR-MIKROCOMPUTER BFZ/MFA 10.4.



BFZ MFA

Diese Übung ist Bestandteil eines Mediensystems, das im Rahmen eines vom Bundesminister für Bildung und Wissenschaft, vom Bundesminister für Forschung und Technologie sowie der Bundesanstalt für Arbeit geförderten Modellversuches zum Einsatz der "Mikrocomputer-Technik in der Facharbeiterausbildung" vom BFZ-Essen e.V. entwickelt wurde.

Inhaltsverzeichnis

Theorieteil 1

1.1. Einleitung
1.2. Die Elemente des Mikroprozessors (CPU)
1.2.1. Befehlszähleinrichtung und Adreßregister
1.2.2. Befehlsregister und Befehlsdecoder
1.2.3. Ablaufsteuerung
1.2.4. Arithmetische und Logische Einheit und Akkumulator
1.3. Befehle und Befehlsabarbeitung
1.4. Befehlsarten
1.5. Schreibweise von Programmen

Übungsteil 1

A1 Verfolgen der Abarbeitung eines Programms im Einzelschrittbetrieb
A2-A4 Verfolgen der Wirkung verschiedener Befehle
A5 Einsatz und Aufbau von Warteschleifen
A6 Einsatz und Aufbau von Verzögerungsschleifen
A7 Programmierung einer einfachen Fußgängerampel

Theorieteil 2

2.1. Einleitung
2.2. Die Erzeugung des Taktsignals
2.3. Die Übertragung von Daten- und Adreßsignalen im "Zeitmultiplexbetrieb"
2.4. Die Erzeugung der Steuersignale $\overline{\text{MEMR}}$, $\overline{\text{MEMW}}$, $\overline{\text{IOR}}$ u. $\overline{\text{IOW}}$
2.5. Die Bearbeitung eines Befehls
2.6. Zusammenfassung
2.7. Messungen mit dem Oszilloskop am Mikrocomputer-Bus
2.7.1. Periodische Signalzustände (Zyklische Programme)
2.7.2. Die Ableitung von Triggersignalen
2.8. Untersuchung der Funktion der Befehlszähleinrichtung im "Free-Run-Mode"

Übungsteil 2

A1 Messen der Quarzfrequenz und des Systemtaktes mit dem Oszilloskop
A2 Prüfung der Prozessorbaugruppe im "Free-Run-Mode"
A3 Verfolgen der Bearbeitung eines Programms mit dem Oszilloskop

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

MIKROPROZESSOR-MIKROCOMPUTER

BFZ/MFA 10.4.

THEORIETEIL 1

1.1. Einleitung

Das zentrale Element eines Mikrocomputers ist der Mikroprozessor. Seine immer wiederkehrenden Aufgaben sind es ...

Aufgaben des Mikroprozessors

- den Datenverkehr auf den Datenleitungen in Verbindung mit Adreß- und Steuer-Bus zu steuern, um ...
- Bitkombinationen aus dem Speicher in den Prozessor zu transportieren, ...
- die diesen Bitkombinationen entsprechenden Befehle zu ermitteln (Befehlsentschlüsselung) und diese dann auszuführen, ...

Befehlsentschlüsselung

- alle hierzu notwendigen Daten (auch sie sind nur Bitkombinationen) aus dem Speicher zu holen oder Informationen im Speicher abzulegen ...
- sowie den Datenaustausch mit den Ein- und Ausgabe-Einheiten zu steuern.

Die grundsätzliche Arbeitsweise verschiedener Prozessortypen ist gleich. Im internen Aufbau (Prozessor-Architektur) können sie sich allerdings erheblich unterscheiden; es gibt jedoch einige Elemente, die in allen Mikroprozessoren wiederzufinden sind. Die Kenntnis der Bedeutung und Funktion dieser Elemente ist unbedingte Voraussetzung für das Verständnis der Arbeitsweise eines Mikrocomputers.

1.2. Die Elemente des Mikroprozessors (CPU)

Die interne Struktur eines Mikroprozessors ist in Bild 1 dargestellt. Die in jedem Prozessor vorhandenen Elemente sind ...

- Ablaufsteuerung,
- Befehlszähleinrichtung mit Adreßregister,
- Befehlsregister mit Befehlsdecoder,
- Arithmetische und logische Einheit mit Akkumulator und Statusregister.

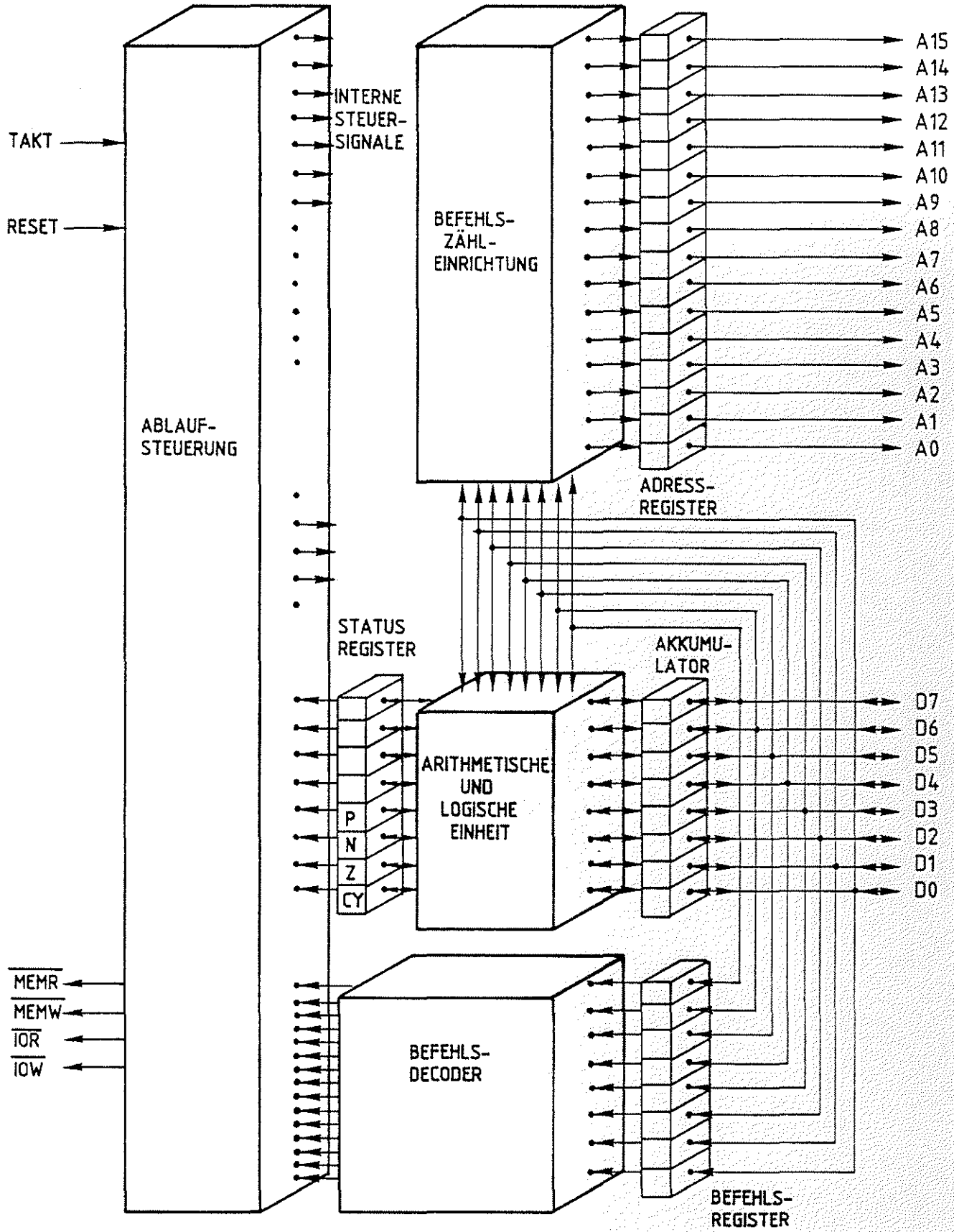


Bild 1: Interne Struktur eines Mikroprozessors

Die ersten drei Elemente gehören zur Control-Unit (CU = Steuerwerk) des Prozessors. Sie sorgen dafür, daß die Bitkombinationen aus dem Speicher geholt werden, und daß die ihnen entsprechenden Befehle erkannt und ausgeführt werden. Zur Arithmetischen und Logischen Einheit (ALU, Rechenwerk) gehört das Status-Register und der Akkumulator. Alle Speicher in einem Prozessor nennt man üblicherweise Register. Der Akkumulator (kurz Akku) ist ein besonderes Register. Es ist eng mit der ALU verknüpft und speichert die Daten, die in der ALU verarbeitet werden sollen bzw. die als Ergebnis einer Rechenoperation in der ALU anfallen. Darauf werden wir im folgenden noch eingehen (siehe 1.2.4.).

ALU
Statusregister
Register
Akkumulator

Ein 8-Bit-Prozessor besitzt einen 8-Bit-breiten Daten-Bus, über den er neben den Daten die auszuführenden Befehle aus dem Speicher holt. Ein durch eine Bitkombination verschlüsselter Befehl an den Prozessor kann daher nur 8-Bit-breit sein. In diesem Zusammenhang spricht man in der Computertechnik auch von Wortbreite oder Wortlänge. Die Wortbreite des in Bild 1 dargestellten Prozessors beträgt 8 Bit oder 1 Byte. Dagegen besitzt z.B. ein 16-Bit-Prozessor eine Wortbreite von 2 Byte. Wortbreite und Daten-Bus-Breite müssen nicht immer übereinstimmen.

Wortbreite

Die Wortbreite eines Prozessors bestimmt die Anzahl der möglichen Bitkombinationen, die er unterscheiden kann. Ein 8-Bit-Prozessor kann max. 256 (2^8) verschiedene Bitkombinationen unterscheiden. Welche Bitkombination welche Reaktion im Prozessor bewirkt, legt der Hersteller des Mikroprozessors fest. Zu jedem Mikroprozessor gehört daher eine Befehlsliste für den Anwender. Im folgenden werden wir einige Befehle aus der Befehlsliste für den Prozessortyp 8085 kennenlernen.

Befehlsliste

Unabhängig von der Art der Befehle vollziehen sich im Prozessor immer wieder die folgenden Schritte:

- Der Prozessor sendet auf den Adreßleitungen die Adresse der Speicherzeile aus, in der der nächste zu verarbeitende Befehl steht.

- Er liest mit dem Steuersignal $\overline{\text{MEMR}}$ diesen Befehl und speichert ihn im Befehlsregister.
- Er stellt fest, um welchen der ihm "bekannten" (vom Hersteller festgelegten) Befehle es sich handelt. Dies nennt man Entschlüsselung des Befehls. Abhängig von dieser Entschlüsselung holt er entweder noch zusätzlich erforderliche Daten für die Befehlsausführung aus dem Speicher oder führt den Befehl sofort aus.
- Wenn der Prozessor den Befehl ausgeführt hat, fährt er mit dem zuerst genannten Schritt fort.

Flußdiagramm

Dieser Vorgang läßt sich übersichtlich in Form eines Fluß-Diagramms darstellen (Bild 2).

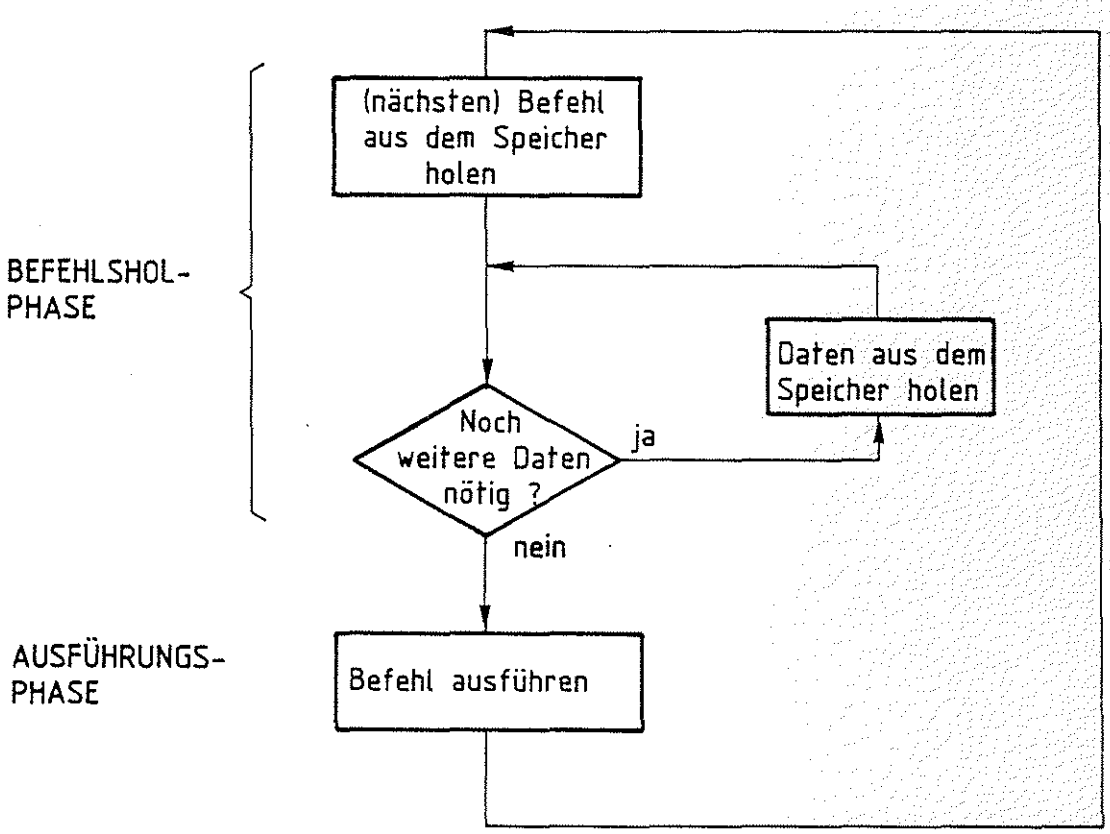


Bild 2: Flußdiagramm zur Befehlsabarbeitung

Befehlshol-Ausführungs-Phase

Jede Befehlsabarbeitung kann daher in zwei Phasen unterteilt werden, die Befehlshol- und die Ausführungs-Phase. Wie die einzelnen Prozessorelemente an diesem Ablauf beteiligt sind, wird im folgenden beschrieben.

1.2.1. Befehlszähleinrichtung und Adreßregister

Im Speicher sind die Befehle hintereinander abgespeichert. Die Befehlszähleinrichtung steuert die Reihenfolge, in der die Befehle abgearbeitet werden. Hierfür besitzt sie intern einen 16-Bit-Zähler, den man Befehls- oder Programmzähler (PC = Programm Counter) nennt.

Programmzähler

Der Zähler wird während jeder Befehlsausführung jeweils um Eins erhöht und liefert so die Adresse der Speicherzeile, deren Inhalt (Befehl oder Daten) verarbeitet wird.

Das Adreßregister wird zu Beginn einer jeden Befehlsholphase mit dem Inhalt des Programmzählers geladen. Während der Befehlsausführung kann bei bestimmten Befehlen der übernommene Progammzählerstand im Adreßregister überschrieben werden. Dies ist beispielsweise der Fall, wenn bei der Befehlsausführung eine Ein- oder Ausgabe-Baugruppe angesprochen wird und die Port-Adresse in das Adreßregister geladen werden muß.

Adreßregister

1.2.2. Befehlsregister und Befehlsdecoder

Das über den Programmzähler und das Adreßregister adressierte Befehlsbyte gelangt während der Befehlsholphase in das Befehlsregister. Im Befehlsdecoder wird ermittelt, welchem Befehl dieses Befehlsbyte entspricht. Das Ergebnis dieser Decodierung oder Entschlüsselung wird der Ablaufsteuerung mitgeteilt.

1.2.3. Ablaufsteuerung

Die Ablaufsteuerung erzeugt alle internen und externen Steuersignale für das Holen und Ausführen der Befehle. Jeder Befehl löst eine bestimmte Folge (Sequenz) von Steuerfunktionen aus, so daß man bei der Ablaufsteuerung auch von einem Steuer-Sequenzer spricht. Die Steuer-Sequenzen aller möglichen Befehle (max. 256) werden in der Ablaufsteuerung in einem sogenannten Mikroprogrammspeicher aufbewahrt. Dieser Mikroprogrammspeicher wird vom Hersteller der CPU programmiert.

Mikroprogramm-Speicher

Mikroprozessoren sind taktgesteuerte Schaltungen.

Taktsteuerung

Zur Erzeugung des Taktsignals besitzen einige Prozessoren einen internen Oszillator, bei anderen muß das Taktsignal extern erzeugt werden. Damit der Prozessor nach dem Anlegen der Betriebsspannung nicht willkürlich mit der Programmabarbeitung beginnt, muß er in den Grundzustand gebracht werden. Dafür besitzen die Prozessoren einen Rücksetz- (RESET-) Eingang. Ein kurzzeitiger Impuls an diesem Eingang bewirkt unter anderem, daß der Programmzähler gelöscht und die Befehlsholphase eingeleitet wird. Nach einem RESET beginnt der Prozessor also immer bei der Speicherstelle 0000H mit der Befehlsabarbeitung.

Taktimpuls-erzeugung

RESET

1.2.4. Arithmetische und Logische Einheit und Akkumulator

Die eigentliche Verarbeitung der Daten erfolgt in der Arithmetischen und Logischen Einheit (Rechenwerk). Man nennt die Daten, die verarbeitet werden, auch Operanden. Die ALU kann einen oder zwei Operanden verarbeiten. Wie und in welcher Form die Verarbeitung erfolgt, d.h. welche Operation in der ALU ausgeführt wird, hängt von dem gerade ausgeführten Befehl ab. Man unterscheidet zwischen arithmetischen und logischen Operationen. Arithmetische Operationen sind z.B. ...

Operand

Operation

arithmetische Operation

— Addition zweier Operanden:

```
  01010011
+ 10000100
----------
  11010111
```

— Subtraktion zweier Operanden:

```
  10011101
- 00101100
----------
  01110001
```

— Inkrementieren eines Operanden: (Um 1 erhöhen)

```
  01110111
+        1
----------
  01111000
```

— Dekrementieren eines Operanden: (eine Eins abziehen)

```
  10001000
-        1
----------
  10000111
```

logische Operation

Dagegen sind logische Operationen z.B. ...

— eine Bit-für-Bit UND-Verknüpfung zweier Operanden:

```
  10011101
  11000111
  --------
  10000101
```

— eine Bit-für-Bit ODER-Verknüpfung zweier Operanden:

```
  00101001
  11100001
  --------
  11101001
```

Für die Zuführung der beiden Operanden besitzt die ALU zwei Eingangskanäle. Der eine Operand gelangt über den Daten-Bus in die ALU, der andere über den Akkumulator. Das Ergebnis einer arithmetischen oder logischen Operation in der ALU wird im Akku abgelegt. Alle Mikroprozessoren besitzen Befehle, die das Laden (engl. load) des Akkus mit Daten aus dem Speicher bzw. das Ablegen des Akku-Inhalts im Speicher (engl. store) bewirken. Außerdem läuft der Datenverkehr von und zu den Ein- und Ausgabe-Einheiten über den Akkumulator ab. Der Akkumulator hat somit eine zentrale Funktion im Mikroprozessor.

Daten von und zu E/A-Einheiten über den Akku

Manchmal soll nach einer arithmetischen oder logischen Operation in der ALU aufgrund eines besonderen Ergebnisses eine Entscheidung getroffen werden (Programmverzweigung). Ein besonderes Ergebnis ist z.B. der Überlauf bei einer Addition, der zustande kommt, wenn die Summe aus den beiden Operanden eine Zahl ergibt, die sich mit acht Bit nicht mehr darstellen läßt.

```
                 11100011        (227)
               + 00100001      + ( 33)
               ----------      -------
Überlauf ⟶   1 00000100        (260)  > 255
```

Bei einer Subtraktion kann es vorkommen, daß beide Operanden gleich groß sind und als Ergebnis Null auftritt. Diese und andere Besonderheiten eines Ergebnisses werden nach einer Operation in der ALU im sogenannten STATUS-Register durch Setzen oder Löschen einzelner Bits angezeigt.

Statusregister

Nachfolgend sind Bezeichnung und Bedeutung einiger Status-Bits zusammengefaßt.

Carry-Bit (CY) = 1: Überlauf ist aufgetreten
= 0: kein Überlauf

Zero-Bit (Z) = 1: Ergebnis ist Null
= 0: Ergebnis ist ungleich Null

Negativ-Bit (N)* = 1: Ergebnis ist negativ (Bit7 im Akku ist 1)
= 0: Ergebnis ist positiv (Bit7 im Akku ist 0)

Parity-Bit (P) = 1: Anzahl der Bits mit dem Wert 1 ist geradzahlig
= 0: Anzahl der Bits mit dem Wert 1 ist ungerade

Flags

Die einzelnen Bits, die jeweils einen bestimmten Zustand anzeigen, nennt man auch Flags (Flagge). Der Zustand der Flags wird unmittelbar der Ablaufsteuerung zugeführt, da es Befehle für den Prozessor gibt, die in Abhängigkeit vom Zustand der Flags ausgeführt oder ignoriert werden.

* Wird bei der vorzeichenbehafteten Darstellung von Dualzahlen verwendet.

1.3. Befehle und Befehlsabarbeitung

Befehlsliste

Die Befehle, die ein Mikroprozessor ausführen kann, werden vom Bausteinhersteller in einer Befehlsliste beschrieben. Einige typische Prozessorbefehle und ihre Wirkungen werden im folgenden dargestellt. Dazu nehmen wir das kleine Programmbeispiel zu Hilfe, das Sie in der Übung "Speicher-Einheiten BFZ/MFA 10.3." in den Speicher eingegeben haben.

Speicheradresse (hex.)	Inhalt (hex.)
0000	DB
0001	01
0002	D3
0003	02
0004	C3
0005	00
0006	00

Dieses Programm veranlaßt den Mikroprozessor ...

- den Signalzustand an den Eingängen der Eingabe-Baugruppe mit der Port-Adresse 01 in den Akku zu holen,
- den Akku-Inhalt an die Ausgabe-Baugruppe mit der Port-Adresse 02 zu übergeben,
- die Befehlsabarbeitung an der Speicherstelle 0000 (1. Befehl) fortzusetzen.

Mehr-Byte-Befehle

Das Programm besteht aus drei Befehlen. Zur Speicherung dieser drei Befehle sind jedoch sieben Speicherzeilen erforderlich. Ein vollständiger Befehl besteht also aus dem Befehlsbyte und eventuell erforderlichen Zusatzangaben. Man unterscheidet daher Ein-, Zwei- und Drei-Byte-Befehle. Das obige Programm enthält z.B. zwei Zwei-Byte- und einen Drei-Byte-Befehl.

Die Abarbeitung der einzelnen Befehle vollzieht sich in den folgenden Schritten:

1. Befehl, Befehlsholphase:
   - Nach einem RESET lädt der Prozessor den Inhalt der Speicherzeile 0000H - also den Binärwert von DB - in das Befehlsregister und entschlüsselt den Befehl (Bild 3).
     (DBH ≙ 11011011) Diese Bitkombination veranlaßt den Prozessor, eine Eingabe-Baugruppe zu lesen. Zur Ausführung dieses Befehls benötigt er noch die Port-Adresse der Eingabe-Baugruppe. Vom Hersteller des Prozessors ist festgelegt, daß diese Port-Adresse in der dem Befehlsbyte folgenden Speicherzeile abgelegt sein muß (Der Programmierer muß dafür sorgen, daß sie dort auch zu finden ist).

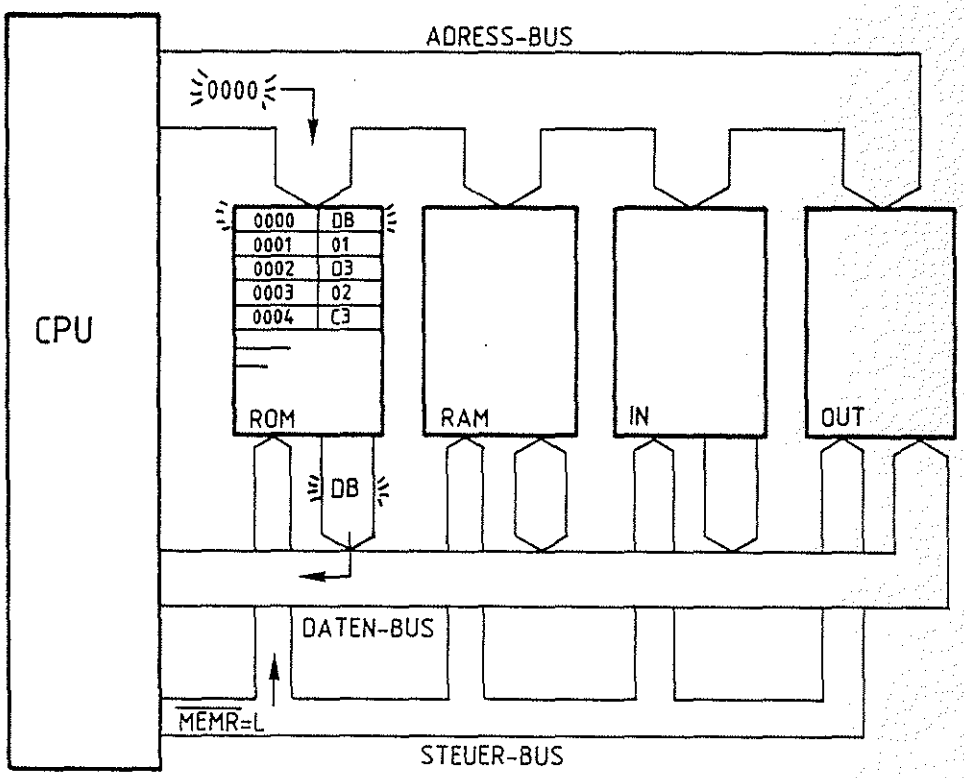


Bild 3: Lesen des Befehlsbytes DB aus dem Speicher

— Zum Lesen des Inhalts der nächsten Speicherzeile erhöht der Prozessor den Befehlszählerstand um Eins, transportiert ihn zum Adreßregister und aktiviert das Steuersignal $\overline{MEMR}$. Der Speicher übergibt daraufhin den Inhalt der Speicherzeile 0001 an den Akkumulator. Jetzt hat der Prozessor alle Informationen für die Ausführung des 1. Befehls (Bild 4).

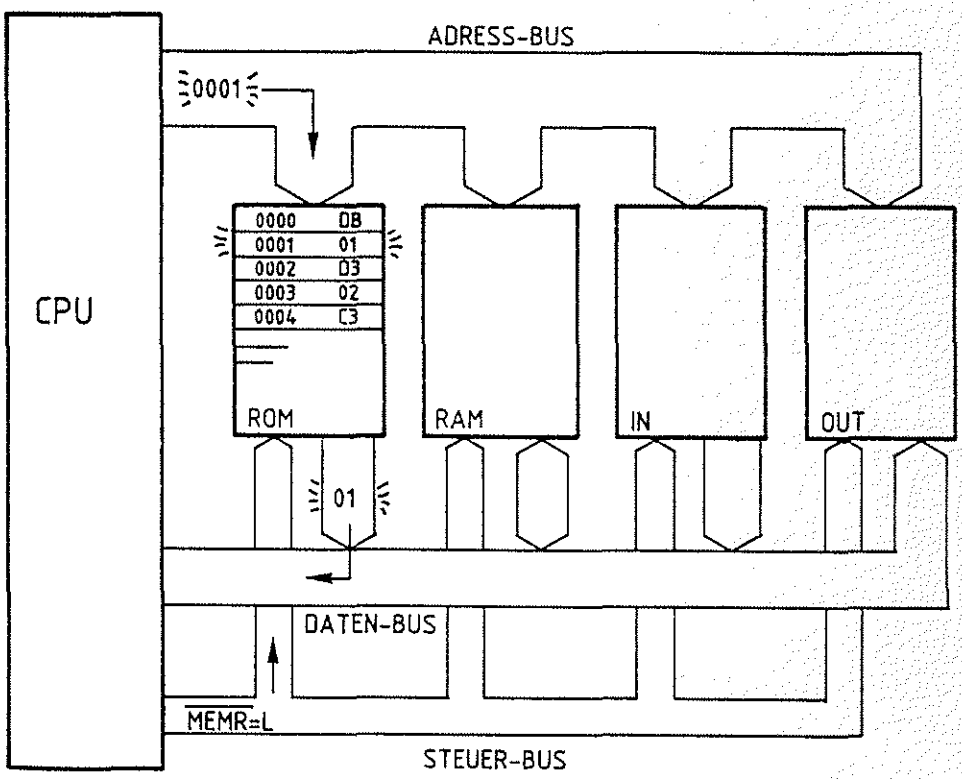


Bild 4: Lesen der Port-Adresse 01 aus dem Speicher

1. Befehl, Ausführungsphase:

   — Für die Befehlsausführung lädt der Prozessor die gerade gelesene Port-Adresse in das Adreßregister und aktiviert das Steuersignal $\overline{IOR}$. Die angewählte Eingabe-Baugruppe schaltet daraufhin den Signalzustand der Eingangsleitungen auf den Daten-Bus, den der Prozessor dann mit dem Wegschalten des Steuersignals in den Akku übernimmt. Der erste Befehl ist abgearbeitet (Bild 5).

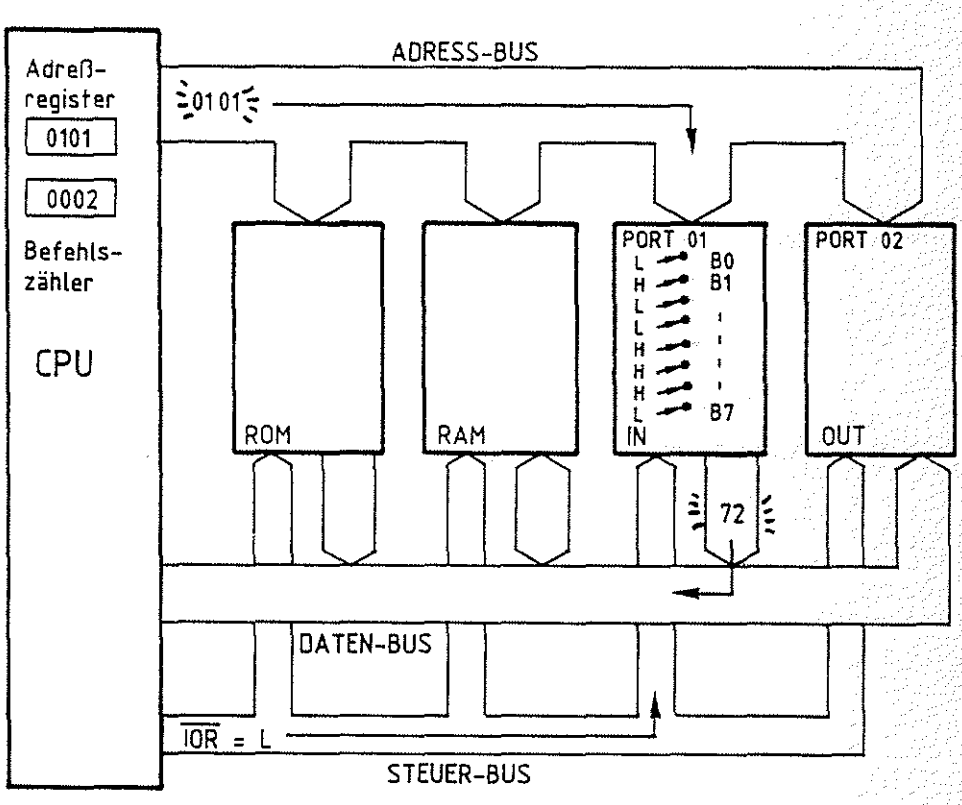


Bild 5: Lesen der Eingabe-Daten von Port 01
(Ausführen des Befehls DB)

Die Befehlszähleinrichtung des Prozessors erhöht während der Befehlsholphase den Befehlszähler jeweils mit dem Lesen einer Speicherstelle, so daß schon während der Befehlsausführung der Befehlszähler die Anfangsadresse des nächsten Befehls enthält. Man sagt auch, der Befehlszähler zeigt auf den nächsten Befehl (Bild 5).

2. Befehl, Befehlsholphase:

— Für das Lesen des nächsten Befehls lädt der Prozessor das Adreßregister mit dem Inhalt des Befehlszählers, aktiviert das Steuersignal $\overline{MEMR}$ und übernimmt das erste Byte des zweiten Befehls in das Befehlsregister. Die Befehlsdecodierung signalisiert der Ablaufsteuerung, daß Daten aus dem Akku an eine Ausgabe-Baugruppe übergeben werden sollen (D3H ≙ 11010011). Auch bei diesem Befehl muß die Port-Adresse der Baugruppe in der dem Befehl folgenden Speicherzeile stehen..

— Der Prozessor liest den Speicherinhalt unter der Adresse 0003H und transportiert ihn für die Befehlsausführung in das Adreßregister.

2. Befehl, Ausführungsphase:

— Für die Ausführung des Befehls schaltet der Prozessor noch den Akku-Inhalt auf den Daten-Bus und aktiviert das Steuersignal $\overline{IOW}$, mit dem die angewählte Ausgabe-Baugruppe die Daten übernimmt. Der zweite Befehl ist abgearbeitet.

Transportbefehle

Die beiden ersten Befehle, die Daten im Computer hin- und hertransportieren, gehören zur Gruppe der Transportbefehle. Der nächste auszuführende Befehl unter der Speicheradresse 0004H ist kein Transportbefehl. Dieser Befehl (C3H ≙ 11000011) veranlaßt den Prozessor, den Befehlszählerstand zu verändern, damit die Befehlsabarbeitung an einer anderen Speicherstelle fortgesetzt wird. Die Speicheradresse, mit der der Befehlszähler geladen werden soll, muß in den zwei, dem Befehlsbyte folgenden Speicherzeilen stehen (Adressen erfordern zwei Byte). Hinter dem Befehlsbyte (C3H) muß der untere Adreßteil (A0 bis A7) und danach der obere (A8 bis A15) abgelegt werden. Den unteren Adreßteil nennt man auch niederwertiges und den oberen höherwertiges Adreßbyte. Dieser Befehl, mit dem Sprünge im Speicher ausgeführt werden können, heißt Sprungbefehl (engl. jump). Er gehört zur Gruppe der Programmsteuerbefehle.

Niederwertiges u. höherwertiges Adreßbyte

Sprungbefehl

3. Befehl, Befehlsholphase:

— Der Prozessor liest die Speicherzeile 0004H und transportiert ihren Inhalt (C3H) in das Befehlsregister.

— Da er für die Ausführung des Sprungbefehls die Sprungadresse benötigt, liest er die nächste Speicherzeile (0005H), in der der niederwertige Adreßteil abgelegt sein muß.

— Für den höherwertigen Adreßteil greift der Prozessor noch einmal auf den Speicher zu und liest den Inhalt der Speicherzeile 0006H.

3. Befehl, Ausführungsphase:

— Für die Befehlsausführung überschreibt der Prozessor den Befehlszählerinhalt mit der gerade gelesenen Adresse. Der Befehl ist abgearbeitet.

Da der Befehlszählerinhalt mit der Adresse 0000H überschrieben wurde, beginnt der Prozessor wieder bei der Speicheradresse 0000H mit der Befehlsholphase. Das Programm wird fortlaufend abgearbeitet und hat zur Folge, daß der Signalzustand an den Eingängen der Eingabe-Baugruppe an den Ausgängen der Ausgabe-Baugruppe eingestellt wird.

Programmsteuerbefehle

1.4. Befehlsarten

Neben den Transport- und Programmsteuerbefehlen gibt es noch die wichtige Gruppe der Verarbeitungsbefehle, die Operationen in der ALU auslösen. Die drei folgenden Befehlslisten für ...

Verarbeitungsbefehle

— Transportbefehle
— Verarbeitungsbefehle
— Programmsteuerbefehle

enthalten nur einen kleinen Teil der beim Prozessor 8085 verfügbaren Befehle. Jede Liste enthält

— den binären und hexadezimalen Befehlscode,
— Angaben zur Anzahl der Bytes des Befehls,
— Angaben zur Wirkung des Befehls und
— den Mnemonischen Code des Befehls (Erklärung folgt).

T R A N S P O R T B E F E H L E				
Befehlsbyte			Bef.	
Mnemo.	hex.	binär	Länge	Wirkung/Beispiel
LDA	3A	00111010	3	Der Inhalt der Speicherzeile, deren Adresse in den beiden dem Befehl folgenden Speicherzeilen steht, wird in den Akku geladen. Beispiel: [3A] bewirkt, daß der Inhalt [04] der Speicherzeile 1304H [13] in den Akku geladen wird.
STA	32	00110010	3	Der Akku-Inhalt wird im Speicher abgelegt. Die Adresse der Speicherzeile steht in den beiden dem Befehl folgenden Speicherzeilen. Beispiel: [32] bewirkt, daß der Akku-Inhalt [FE] unter der Adresse 70FEH abge- [70] speichert wird.
MVI A	3E	00111110	2	Der Akku wird mit dem Inhalt der dem Befehl folgenden Speicherzeile geladen. Beispiel: [3E] bewirkt, daß der Akku mit [F9] dem Wert F9H geladen wird.
IN	DB	11011011	2	Der Akku wird mit dem Signalzustand der Eingabe-Baugruppe geladen, deren Port-Adresse in der folgenden Speicherzeile steht. Beispiel: [DB] bewirkt, daß der Signalzustand an [45] den Eingängen der Eingabe-Baugruppe mit der Adr. 45H in den Akku geladen wird.
OUT	D3	11010011	2	Der Akku-Inhalt wird an eine Ausgabe-Baugruppe übergeben. Die Port-Adresse steht in der dem Befehl folgenden Speicherzeile. Beispiel: [D3] bewirkt, daß der Akku-Inhalt an die [AC] Ausgabe-Baugruppe mit der Adr. ACH übergeben wird.

V E R A R B E I T U N G S B E F E H L E				
Befehlsbyte			Bef.	Wirkung/Beispiel
Mnemo.	hex.	binär	Länge	
CMA	2F	00101111	1	Jedes Bit des Akku-Inhaltes wird invertiert (aus 1 wird 0, aus 0 wird 1).
INR	3C	00111100	1	Der Akku-Inhalt wird um 1 erhöht, d.h. inkrementiert.
DCR	3D	00111101	1	Der Akku-Inhalt wird um 1 erniedrigt, d.h. dekrementiert.
ADI	C6	11000110	2	Das dem Befehl folgende Daten-Byte wird zum Akku-Inhalt addiert. Beispiel: C6 bewirkt, daß zum Akku-Inhalt 5BH 5B addiert wird.
SUI	D6	11010110	2	Das dem Befehl folgende Daten-Byte wird vom Akku-Inhalt abgezogen. Beispiel: D6 bewirkt, daß der Akku-Inhalt 3D um 3DH vermindert wird.
ANI	E6	11100110	2	Der Akku-Inhalt wird Bit-für-Bit mit demjenigen Daten-Byte UND-verknüpft, das in der dem Befehl folgenden Speicherzeile steht. Beispiel: E6 wenn vor der Befehlsausführung im 5A Akku 83H steht, so bewirkt der Befehl, daß im Akku der Wert 02H steht.
ORI	F6	11110110	2	Der Akku-Inhalt wird Bit-für-Bit mit demjenigen Daten-Byte ODER-verknüpft, das in der dem Befehl folgenden Speicherzeile steht. Beispiel: F6 wenn vor der Befehlsausführung im 0F Akku 83H steht, so bewirkt der Befehl, daß im Akku der Wert 8FH steht.

P R O G R A M M S T E U E R B E F E H L E				
Befehlsbyte			Bef.	Wirkung/Beispiel
Mnemo.	hex.	binär	Länge	
JMP	C3	11000011	3	Die Programmabarbeitung wird an der Speicherzeile fortgesetzt, deren Adresse in den beiden dem Befehl folgenden Speicherzeilen steht. Beispiel: C3 bewirkt, daß der nächste Befehl 12 von der Speicherzeile F712H F7 gelesen wird.
JZ	CA	11001010	3	Dieser Sprungbefehl zu der Speicheradresse, die in den beiden folgenden Speicherzeilen steht, wird nur ausgeführt, wenn die letzte Rechenoperation Null ergab. Beispiel: CA bewirkt, daß der nächste Befehl 12 nur dann von der Speicherzeile F7 F712H gelesen wird, wenn das Zero-Bit 1 ist. Wenn nicht, wird der Sprungbefehl ignoriert.
JNZ	C2	11000010	3	Dieser Sprungbefehl zu der Speicheradresse, die in den folgenden Speicherzeilen steht, wird nur dann ausgeführt, wenn die letzte Rechenoperation nicht Null (Zero-Bit = 0) ergab.
HLT	76	01110110	1	Dieser Befehl bewirkt, daß die Befehlsabarbeitung gestoppt wird. Nur ein RESET (oder Interrupt) kann die Befehlsabarbeitung wieder einleiten.

1.5. Schreibweise von Programmen

Maschinencode

Der Binär-Code der Befehle (und die hexadezimale Schreibweise) wird auch Maschinencode genannt, weil nur er vom Prozessor (der Maschine) verstanden wird. Diesen Code kann man sich jedoch schlecht merken und man braucht zum Lesen und Schreiben der Programme immer eine Befehlsliste. Daher hat man zu jedem Befehlsbyte eine dem Anwender verständliche Abkürzung (Merk- oder Mnemo-Code) eingeführt, die in den meisten Fällen direkt die Wirkung des jeweiligen Befehls erkennen läßt. Diese Abkürzungen sind in den vorangegangenen Befehlslisten in der Spalte "Mnemo." aufgeführt und heißen Assembler-Code (engl. assemble = zusammensetzen).

Mnemo-Code

Assembler-Code

Die folgenden Beispiele zeigen, wie diese Abkürzungen entstanden sind.

Transportbefehle

- LDA = Load ACCU direct,
  Lade den Akku direkt
- STA = Store ACCU direct,
  speichere den Akku direkt
- MVI A = Move immediate,
  bewege unmittelbar (in den Akku)
- IN = Input,
  Eingabe
- OUT = Output,
  Ausgabe

Verarbeitungsbefehle

- CMA = Complement ACCU,
  komplementiere (invertiere) Akku
- INR A = Increment Register,
  inkrementiere Register A (Akku)
- DCR A = Decrement Register,
  dekrementiere Register A (Akku)
- ADI = Add immediate to ACCU,
  addiere unmittelbar zum Akku
- SUI = Subtract immediate from ACCU,
  subtrahiere unmittelbar vom Akku
- ANI = And immediate with ACCU,
  UND unmittelbar mit Akku
- ORI = Or immediate with ACCU,
  ODER unmittelbar mit Akku

Programmsteuerbefehle

JMP = Jump unconditional,
springe unbedingt (immer)

JZ = Jump on Zero,
springe wenn Null (bedingt)

JNZ = Jump on no Zero
springe wenn nicht Null (bedingt)

HLT = Halt,
anhalten

Programme, die im Hex- oder Binär-Code vorliegen, heißen Maschinenprogramme und solche, die im Assembler-Code vorliegen, nennt man Assemblerprogramme.

Maschinenprogramme

Assemblerprogramme

Häufig schreibt man für die Dokumentation die Bytes eines Befehls in eine Zeile und gibt dabei nur noch die Speicheradresse des Befehlsbytes an. Dadurch werden auch die Maschinenprogramme für den Anwender übersichtlich. In Bild 6 sind die verschiedenen Schreibweisen für Programme gegenübergestellt.

Adresse	Inhalt
0000	DB
0001	01
0002	D3
0003	02
0004	C3
0005	00
0006	00

Maschinenprogramm

Adresse	Befehl
0000	DB 01
0001	D3 02
0004	C3 00 00
0007	..

Maschinenprogramm

Adresse	Befehl
0000	IN 01
0002	OUT 02
0004	JMP 0000
0007	...

Assemblerprogramm

Bild 6: Programmschreibweisen

Programmierer entwickeln ihre Programme zunächst nur im Assembler-Code, weil sie die Kürzel nach einiger Zeit wie die Worte unserer Sprache beherrschen. Man spricht häufig in diesem Zusammenhang auch von der Assemblersprache.

Assembler-Sprache

Nachdem ein Programm fertiggestellt ist, muß es in den Maschinen-Code übersetzt werden, d.h. statt der Kürzel wie IN, OUT, JMP usw. muß der entsprechende Hex-Code (DBH, D3H, C3H, ...) eingesetzt werden. Diese Arbeit nennt man assemblieren, man läßt sie meist von einem Computer ausführen. Das dafür notwendige Programm heißt auch Assembler.

assemblieren

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

MIKROPROZESSOR-MIKROCOMPUTER

BFZ/MFA 10.4.

ÜBUNGSTEIL 1

Übungsteil 1

In dieser Übung werden Sie die Befehlsabarbeitung eines Mikroprozessors verfolgen und gleichzeitig einige typische Prozessorbefehle in ihrer Wirkung kennenlernen. Dazu werden Sie einen vollständigen Mikrocomputer aus Prozessor-, Speicher- und Ein- und Ausgabe-Baugruppe betreiben und mit Hilfe des Bus-Signalgebers und der Bus-Signalanzeige kleine Testprogramme in den (RAM-)Speicher laden. Mit einer Einzelschrittsteuerung, die sich auf der Bus-Signalanzeige befindet, wird der Prozessor während der Befehlsabarbeitung gestoppt. Dadurch wird es möglich, den Signalfluß auf dem Bus-System zu verfolgen.
Zur Durchführung der Übung benötigen Sie:

- 1 Baugruppenträger mit Busverdrahtung (BFZ/MFA 0.1.)
- 1 Bus-Abschluß (BFZ/MFA 0.2.)
- 1 Trafo-Einschub (BFZ/MFA 1.1.)
- 1 Spannungsregelung (BFZ/MFA 1.2.)

} zusammengebaut und geprüft nach FPÜ BFZ/MFA 1.2. A7

- 1 Prozessor 8085 (BFZ/MFA 2.1.)
- 1 8-K-RAM/EPROM (BFZ/MFA 3.1.) bestückt mit mind. 2-K-RAM
- 1 8-Bit-Parallel-Ausgabe (BFZ/MFA 4.1.)
- 1 8-Bit-Parallel-Eingabe (BFZ/MFA 4.2.)
- 1 Bus-Signalgeber (BFZ/MFA 5.1.)
- 1 Bus-Signalanzeige (BFZ/MFA 5.2.)

Allgemeine Hinweise zur Durchführung der Übungen:

- Die Einschübe dürfen nur bei abgeschalteter Betriebsspannung gesteckt oder gezogen werden
- Aufgrund der Busverdrahtung können die Baugruppen in beliebige Steckplätze gesteckt werden
- Den logischen Signalen "0" und "1" sind die folgenden Pegel zugeordnet:

  log. "0" ≙ 0...0,8 V (LOW)

  log. "1" ≙ 2,4...5 V (HIGH)

- Alle zur Messung an den Baugruppen vorgegebenen Arbeitsblätter enthalten:
  = Angaben über den Sinn der jeweiligen Messung
  = Angaben über einzustellende Bedingungen
  = Aufgabenstellungen, ggf. mit Hinweisen zu möglichen Fehlern.

Bedienungshinweise:

Prozessor 8085:

Die Prozessor-Baugruppe ist mit dem Mikroprozessor 8085 aufgebaut. In der Frontplatte der Baugruppe befindet sich ein RESET-Taster, über den der Prozessor in den Grundzustand gebracht werden kann. Der Grundzustand wird auch mit dem Einschalten der Betriebsspannung eingenommen. Der Prozessor kann über bisher noch nicht beschriebene Steuereingänge so betrieben werden, daß er während der Befehlsabarbeitung nach jedem Funktionsschritt (Maschinenzyklus) stoppt, so daß man die Aktivitäten des Prozessors auf dem System-Bus mit der Bus-Signalanzeige verfolgen kann. Die dafür notwendigen Schaltungskomponenten (Einzelschrittsteuerung, Single Step) befinden sich auf der Baugruppe Bus-Signalanzeige.

Bus-Signalgeber:

Der Bus-Signalgeber wird in dieser Übung benötigt, um kleine Testprogramme in den RAM-Speicher zu laden. Befindet sich mit dem Bus-Signalgeber zusätzlich die Prozessor-Baugruppe im Baugruppenträger, so darf immer nur eine der beiden Baugruppen freigegeben werden, d.h. auf den System-Bus wirken. Die Freigabe der Baugruppen erfolgt über den Schalter ON/OFF am Bus-Signalgeber:

Stellung ON	-	Bus-Signalgeber frei, Prozessor gesperrt
Stellung OFF	-	Bus-Signalgeber gesperrt, Prozessor frei

Für die Programmeingabe wird der Prozessor gesperrt (Schalterstellung ON). An den Codierschaltern des Bus-Signalgebers werden die Speicheradressen und die Daten eingestellt und durch Betätigen der Steuersignaltaste MEMW in den Speicher geschrieben. Nach Eingabe des vollständigen Programms in den Speicher wird der Schalter ON/OFF in die Stellung OFF gebracht. Gleichzeitig mit dem Umschalten wird am Prozessor ein RESET ausgelöst, so daß die Programmabarbeitung bei der Speicherstelle 0000H beginnt.

Bus-Signalanzeige:

Der Schalter HLT/RUN und der Taster STEP werden im Einzelschrittbetrieb verwendet. Steht der Schalter HLT/RUN in der Stellung HLT (Halt), so wird die Befehlsabarbeitung des Prozessors gestoppt. Mit jedem Betätigen des Tasters STEP wird dann der Prozessor für genau einen Maschinenzyklus freigegeben. Immer dann, wenn der Prozessor das erste Byte eines Befehls liest (Befehlsholphase), leuchtet in der Frontplatte der Baugruppe die Leuchtdiode INSTRUCTION FETCH auf. Steht der Schalter HLT/RUN in Stellung RUN, so arbeitet der Prozessor ohne Unterbrechung. Der Umschalter ADDR. STOP-ON/OFF, über den der Einzelschrittbetrieb beim Auftreten einer ganz bestimmten Adresse auf dem Adreß-Bus aktiviert werden kann (Adressen-Stop), bleibt bei der Durchführung der Meßübungen in der Stellung OFF.

Übungsteil 1

Programmeingabe und Verfolgung des Ablaufs:

Gehen Sie bei der Eingabe der Programme und bei der Verfolgung des Ablaufs im Einzelschrittbetrieb in den folgenden Schritten vor:

- Schalter ON/OFF am Bus-Signalgeber in Stellung ON bringen.
- Schalter HLT/RUN an der Bus-Signalanzeige in Stellung HLT bringen (Schalter ADDR. STOP bleibt immer in Stellung OFF).
- Daten (Programm) mit Hilfe des Bus-Signalgebers in den Speicher eingeben.
- Die im Speicher eingegebenen Daten nochmals kontrollieren.
- Schalter ON/OFF am Bus-Signalgeber in Stellung OFF bringen (CPU aktiv).
- Für die Programmabarbeitung im Einzelschrittbetrieb die Taste STEP an der Bus-Signalanzeige betätigen.
- Für die Programmabarbeitung ohne Unterbrechung des Prozessors bringen Sie den Schalter HLT/RUN in Stellung RUN.

Mikroprozessor-Mikrocomputer

Name:

Übungsteil 1

Datum:

A1.1

Eingeben eines Programms in den RAM-Speicher mit dem Bus-Signalgeber und Verfolgung der Programmabarbeitung mit der Bus-Signalanzeige.

Einstellungen an den Baugruppen:

- Eingabe-Baugruppe: Port-Adresse 12H
- Ausgabe-Baugruppe: Port-Adresse 13H
- RAM-Baugruppe : Steckplatz IC8 mit 2-K-RAM bestückt, Basis-Adr. 0000H
- Bus-Signalgeber : Schalter ON/OFF→ON
- Bus-Signalanzeige: Schalter HLT/RUN→HLT
  Schalter ADDR.STOP→OFF

Stecken Sie die folgenden Baugruppen in den Baugruppenträger und schalten Sie die Betriebsspannung ein:

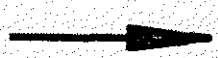
Prozessor 8085, RAM-Baugruppe, Eingabe-Baugruppe, Ausgabe-Baugruppe, Bus-Signalgeber und -Anzeige.

Geben Sie das folgende Programm in den RAM-Speicher ein und verfolgen Sie die Programmabarbeitung im Einzelschritt. Protokollieren Sie dabei den Signalzustand auf dem System-Bus (für jeden Maschinenzyklus).

Verfahren Sie bei der Durchführung der Übung wie auf der vorigen Seite beschrieben.

Programm:

Adresse	Inhalt	Ass.-Code
0000 0001	DB 12	IN 12
0002 0003	D3 13	OUT 13
0004 0005 0006	C3 00 00	JMP 0000



Arbeitsblatt BFZ / MFA 10.4. - 25

Mikroprozessor-Mikrocomputer Name:

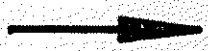
Übungsteil 1 Datum:

A1.2

Protokoll (LED ein ≙ * / LED aus ≙ -)								
STEP	Adreß-Bus	Daten-Bus	Steuer-Bus (LED) MEMW	MEMR	IOW	IOR	INSTR	Kommentar
0	0000	DB	-	*	-	-	*	Lesen des ersten Befehls
1	0001	12	-	*	-	-	-	Lesen der zugehörigen Port-Adresse
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								

Bemerkung:
Der Prozessor 8085 sendet beim Ansprechen einer Ein- und Ausgabe-Baugruppe die 8-Bit-Port-Adresse sowohl auf den unteren (A0 bis A7) als auch auf den oberen (A8 bis A15) acht Adreßleitungen aus.

Bringen Sie nun den Schalter HLT/RUN in Stellung RUN. Der Prozessor arbeitet jetzt das Programm ohne Unterbrechung ab. Für die drei Programmbefehle benötigt er etwa 15 Mikrosekunden, so daß der Signalzustand der Eingabe-Baugruppe für unser Auge unverzögert an der Ausgabe-Baugruppe eingestellt wird.



A2

Ändern der Funktion des Programms von A1 durch Einbau des Befehls CMA (Komplementiere den Akku-Inhalt).

Veranlassen Sie den Prozessor durch Einfügen eines CMA-Befehls (2FH) in das Programm der Aufgabe A1, den von der Eingabe-Baugruppe gelesenen Signalzustand vor der Ausgabe an die Ausgabe-Baugruppe zu invertieren.

Programm:

Adresse	Inhalt	Ass.-Code
0000 0001	DB 12	IN 12
0002	2F	CMA
0003 0004	D3 13	OUT 13
0005 0006 0007	C3 00 00	JMP 0000

Ändern der Funktion des Programms von A1 durch Einbau des Befehls INR A (Inkrementiere den Inhalt des Akkus) bzw. DCR A (Dekrementiere den Inhalt des Akkus).

Fügen Sie anstelle des CMA-Befehls den Inkrementiere- bzw. Dekrementiere-Befehl in das Programm ein. Stellen Sie während der Programmabarbeitung verschiedene Signalzustände an der Eingabe-Baugruppe ein, insbesondere beim Inkrementieren den Signalzustand FFH und beim Dekrementieren den Zustand 00H.

Programm:

a)

Adresse	Inhalt	Ass.-Code
0000 0001	DB 12	IN 12
0002	3C	INR A
0003 0004	D3 13	OUT 13
0005 0006 0007	C3 00 00	JMP 0000

b)

Adresse	Inhalt	Ass.-Code
0000 0001	DB 12	IN 12
0002	3D	DCR A
0003 0004	D3 13	OUT 13
0005 0006 0007	C3 00 00	JMP 0000



Mikroprozessor-Mikrocomputer Name:

Übungsteil 1 Datum:

A3.1

Einbau der Logischen Verarbeitungsbefehle ANI Konstante (Inhalt des Akkus mit dem Wert der Konstanten UND-verknüpfen) bzw. ORI Konstante (Inhalt des Akkus mit dem Wert der Konstanten ODER-verknüpfen).

Testen Sie in gleicher Weise wie in den vorangegangenen Aufgaben die logischen Verarbeitungsbefehle für die UND- und ODER-Verknüpfung des Akku-Inhaltes mit einer Daten-Konstanten (hier 0F).

Programm:

a)

Adresse	Inhalt	Ass.-Code
0000 0001	DB 12	IN 12
0002 0003	E6 0F	ANI 0F
0004 0005	D3 13	OUT 13
0006 0007 0008	C3 00 00	JMP 0000

b)

Adresse	Inhalt	Ass.-Code
0000 0001	DB 12	IN 12
0002 0003	F6 0F	ORI 0F
0004 0005	D3 13	OUT 13
0006 0007 0008	C3 00 00	JMP 0000

Mit dem ANI-Befehl (E6H) können einzelne Bits im Akku gelöscht (ausgeblendet) werden, wenn in der dem Befehl folgenden Daten-Konstanten das entsprechende Bit "0" ist. Die Datenkonstante nennt man auch Maske.

Beispiel:

Inhalt des Akkus: 10110110

Maske für die Bits B0 bis B7: 00000111

Inhalt des Akkus nach Ausführung des ANI-Befehls: 00000110

Diese Bits wurden ausgebl.

Diese Bits wurden nicht geändert

Mit dem ORI-Befehl können einzelne Bits im Akku gesetzt werden, wenn in der dem Befehl folgenden Konstanten das entsprechende Bit "1" ist.

Beispiel:

Inhalt des Akkus: 00100101

Konstante zum Setzen der Bits B6 u. B7: 11000000

Inhalt des Akkus nach Ausführung des ORI-Befehls: 11100101

Diese Bits wurden gesetzt

Diese Bits wurden nicht geändert



Mikroprozessor-Mikrocomputer Name:

Übungsteil 1 Datum:

A3.2

Einfügen der arithmetischen Befehle ADI Konstante (Addiere den Wert der Konstanten zum Akku-Inhalt) und SUI Konstante (Subtrahiere den Wert der Konstanten vom Akku-Inhalt) in das Programm von A1.

Fügen Sie anstelle der log. Befehle ANI und ORI nacheinander die beiden arithmetischen Befehle für die Addition einer Konstanten zum und die Subtraktion einer Konstanten vom Akku-Inhalt in das Programm ein. Stellen Sie an den Schaltern der Eingabe-Baugruppe Daten ein, die bei der Addition zu einem Überlauf und bei der Subtraktion zu einem Borgen führen.

Programm:

a)

Adresse	Inhalt	Ass.-Code
0000 0001	DB 12	IN 12
0002 0003	C6 05	ADI 05
0004 0005	D3 13	OUT 13
0006 0007 0008	C3 00 00	JMP 0000

b)

Adresse	Inhalt	Ass.-Code
0000 0001	DB 12	IN 12
0002 0003	D6 05	SUI 05
0004 0005	D3 13	OUT 13
0006 0007 0008	C3 00 00	JMP 0000

Mikroprozessor-Mikrocomputer

Name:

Übungsteil 1

Datum:

A4.1

Untersuchen der Transportbefehle ...

- MVI A, Konstante: Lade den Akku mit einer Konstanten
- STA Adresse : Speicher den Inhalt des Akkus in der Speicherzeile mit der angegebenen Adresse
- LDA Adresse : Lade den Akku mit dem Inhalt der Speicherzeile, die durch Adresse angegeben ist.

Testen Sie die Wirkung des STA-Befehls mit untenstehendem Programm, indem Sie den Prozessor im Einzelschrittbetrieb betreiben und den Signalzufluß protokollieren.
Die Daten-Konstante, die zunächst mit dem MVI-Befehl in den Akku geladen wird, wird anschließend unter der Adresse 0010H im Speicher abgelegt. Löschen Sie daher vor dem Start des Programms den Inhalt der Speicherzeile 0010H. Wenn der Prozessor den HLT-Befehl erreicht hat, testen Sie den Inhalt der Speicherzeile 0010H mit dem Bus-Signalgeber.

Programm:

Adresse	Inhalt	Ass.-Code
0000 0001	3E 55	MVI A,55
0002 0003 0004	32 10 00	STA 0010
0005	76	HLT

Protokoll (LED ein ≙ * / LED aus ≙ -)

STEP	Adreß-Bus	Daten-Bus	Steuer-Bus (LED) MEMW	MEMR	IOW	IOR	INSTR	Kommentar

Mikroprozessor-Mikrocomputer Name:

Übungsteil 1 Datum:

A4.2

Die Wirkung des LDA-Befehls können Sie mit folgendem Programm testen:

Adresse	Inhalt	Ass.-Code
0000 0001 0002	3A 10 00	LDA 0010
0003 0004	D3 13	OUT 13
0005	76	HLT

Laden Sie vor dem Programmstart die Speicherzeile 0010H mit dem Datum AAH. Die zugehörige Bitkombination muß nach dem OUT-Befehl an der Ausgabe-Baugruppe angezeigt werden.
Protokollieren Sie die Programmabarbeitung im Einzelschrittbetrieb.

P r o t o k o l l						(LED ein ≙ * LED aus ≙ -)		
STEP	Adreß-Bus	Daten-Bus	Steuer-Bus (LED) MEMW	MEMR	IOW	IOR	INSTR	Kommentar

Mikroprozessor-Mikrocomputer Name:

Übungsteil 1 Datum:

A5.1

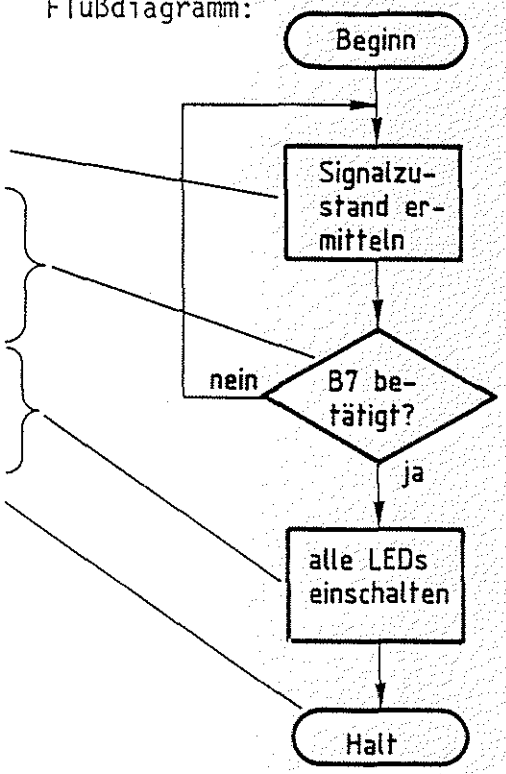
Einsatz und Aufbau von Warteschleifen

In technischen Anwendungen kommt es häufig vor, daß der Computer auf ein Ereignis, wie z.B. das Betätigen eines Schalters, wartet. Tritt das Ereignis ein, so muß eine Reaktion folgen, beispielsweise das Einschalten eines Motors. Das untenstehende Programm hat zur Folge, daß der Prozessor in einer sogenannten Warteschleife auf die Betätigung des Schalters B7 an der Eingabe-Baugruppe wartet. Wird Schalter B7 betätigt, werden alle LEDs an der Ausgabe-Baugruppe eingeschaltet. Realisiert wird die Warteschleife mit Hilfe der Maskierungstechnik (ANI-Befehl) und eines Sprungbefehls (JZ), der nur ausgeführt wird, wenn der Akku-Inhalt nach dem ANI-Befehl Null ergeben hat. Stellen Sie vor dem Start des Programms zunächst an allen Schaltern der Eingabe-Baugruppe L-Signal ein. Testen Sie dann das Verharren des Programms in der Warteschleife im Einzelschrittbetrieb. Vermerken Sie im Protokoll das Betätigen des Schalters B7, wodurch die Warteschleife verlassen wird.

Programm:

Adresse	Inhalt	Ass.-Code
0000 0001	DB 12	IN 12
0002 0003	E6 80	ANI 80
0004 0005 0006	CA 00 00	JZ 0000
0007 0008	3E FF	MVI A,FF
0009 000A	D3 13	OUT 13
000B	76	HLT

Flußdiagramm:



Arbeitsblatt

BFZ / MFA 10.4. - 32

Mikroprozessor-Mikrocomputer

Name:

Übungsteil 1

Datum:

A5.2

Protokoll (LED ein ≙ * / LED aus ≙ -)

STEP	Adreß-Bus	Daten-Bus	Steuer-Bus (LED) MEMW	MEMR	IOW	IOR	INSTR	Kommentar

Mikroprozessor-Mikrocomputer Name:

Übungsteil 1 Datum:

A6.1

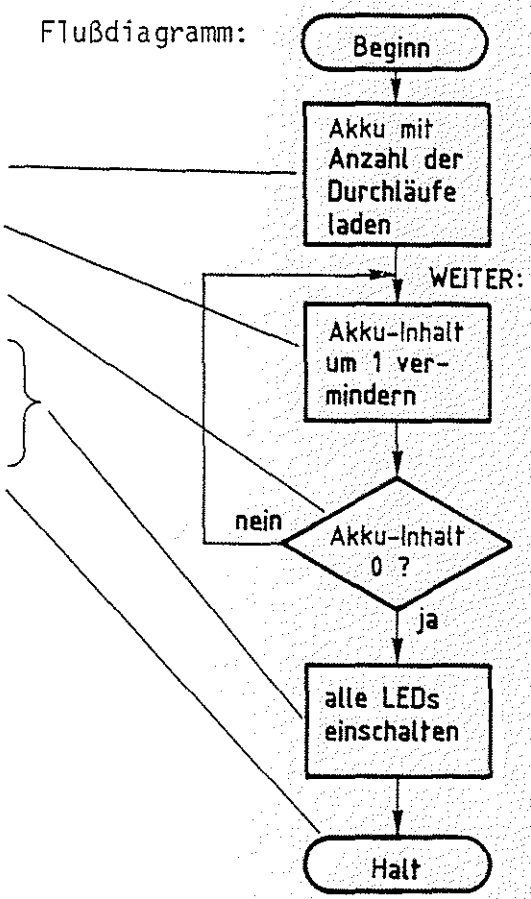
Einsatz und Aufbau von Verzögerungsschleifen

Ähnlich häufig wie die Warteschleife wird in Programmen eine Verzögerungsschleife benötigt. Das folgende kleine Programm enthält eine Verzögerungsschleife, die hier bewirkt, daß die LEDs der Ausgabe-Baugruppe nach dem Programmstart verzögert eingeschaltet werden. Das Programm führt den DCR-Befehl solange aus, bis der Akku-Inhalt Null ist. Dadurch wird der bedingte Sprungbefehl ignoriert und mit den folgenden Befehlen werden die LEDs an der Ausgabe-Baugruppe eingeschaltet. Testen Sie das Programm im Einzelschrittbetrieb und protokollieren Sie den Ablauf.

Programm:

Adresse	Inhalt	Ass.-Code
0000 0001	3E 02	MVI A,02
0002	3D	DCR A
0003 0004 0005	C2 02 00	JNZ 0002
0006 0007	3E FF	MVI A,FF
0008 0009	D3 13	OUT 13
000A	76	HLT

Flußdiagramm:



Arbeitsblatt

BFZ / MFA 10.4. - 34

Mikroprozessor-Mikrocomputer

Name:

Übungsteil 1

Datum:

A6.2

Protokoll (LED ein ≙ * / LED aus ≙ -)

STEP	Adreß-Bus	Daten-Bus	Steuer-Bus (LED) MEMW	MEMR	IOW	IOR	INSTR	Kommentar

Mikroprozessor-Mikrocomputer Name:

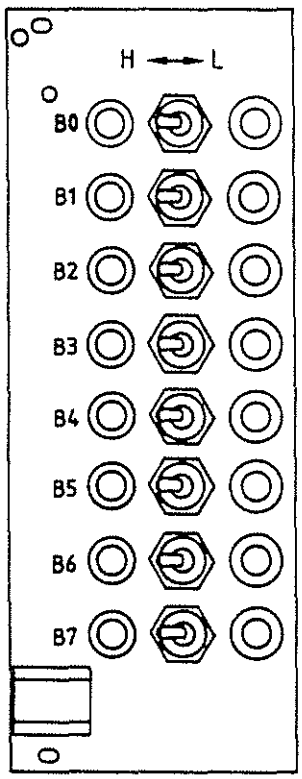
Übungsteil 1 Datum:

A7.1

Programmierung einer einfachen Fußgängerampel

In diesem Arbeitsschritt wird eine der möglichen Lösungen für die in der FTÜ 10.2. besprochene Fußgängerampel gezeigt. Den folgenden Abbildungen können Sie die Belegung der Ein- und Ausgänge des Mikrocomputers entnehmen.

Eingabe-Baugruppe

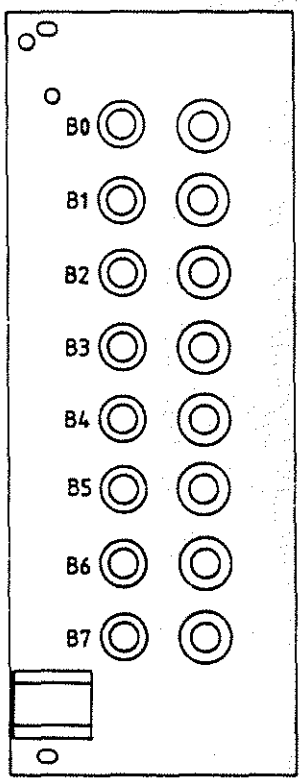


Anforderung Fußgänger

Straßenseite A

Straßenseite B

Ausgabe-Baugruppe



B0 ROT
B1 GELB
B2 GRÜN
} Ampel Autofahrer

B3 ROT
B4 GRÜN
} Ampel Fußgänger

In diesem Programm werden Befehle benutzt, die Ihnen noch nicht bekannt sind. Eine Einführung in den Befehlssatz des Prozessors erfolgt in den Fachtheoretischen Übungen 20. ... .

Zu den neuen Befehlen gehören zwei Befehle für die "Unterprogrammtechnik". Programmteile, die mehrfach in einem Programm benötigt werden, hier ein Zeitverzögerungsprogramm, brauchen dann nur einmal im Speicher abgelegt zu werden. Mit dem Befehl CALL (aufrufen, CDH) springt der Prozessor zum Unterprogramm und kehrt nach Ausführung des Unterprogramms mit dem Befehl RET (zurückkehren C9H) zum eigentlichen (Haupt-) Programm zurück.

Mikroprozessor-Mikrocomputer Name:

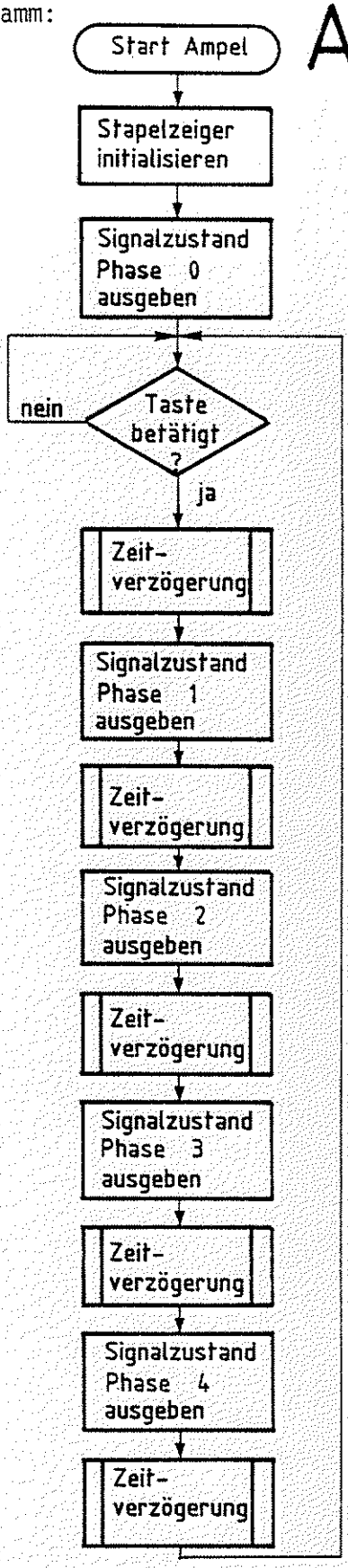
Übungsteil 1 Datum:

Programm:

Flußdiagramm:

A7.2

Adr.	Inhalt	Assembler-Code	
0000	31 00 02	LXI SP,0200	Hauptprogramm
0003	3E 0C	MVI A,0C	
0005	D3 13	OUT 13	
0007	DB 12	IN 12	
0009	E6 03	ANI 03	
000B	CA 07 00	JZ 0007	
000E	CD 30 00	CALL 0030	
0011	3E 0A	MVI A,0A	
0013	D3 13	OUT 13	
0015	CD 30 00	CALL 0030	
0018	3E 11	MVI A,11	
001A	D3 13	OUT 13	
001C	CD 30 00	CALL 0030	
001F	3E 0B	MVI A,0B	
0021	D3 13	OUT 13	
0023	CD 30 00	CALL 0030	
0026	3E 0C	MVI A,0C	
0028	D3 13	OUT 13	
002A	CD 30 00	CALL 0030	
002D	C3 07 00	JMP 0007	
0030	3E 18	MVI A,18	Unterprogramm "Zeitverzögerung"
0032	06 FF	MVI B,FF	
0034	0E FF	MVI C,FF	
0036	0D	DCR C	
0037	C2 36 00	JNZ 0036	
003A	05	DCR B	
003B	C2 34 00	JNZ 0034	
003E	3D	DCR A	
003F	C2 32 00	JNZ 0032	
0042	C9	RET	



Ende Übungsteil 1

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

MIKROPROZESSOR-MIKROCOMPUTER

BFZ/MFA 10.4.

THEORIETEIL 2

2.1. Einleitung

Im Theorieteil 1 wurde die interne Struktur eines Mikroprozessors und die Funktion wichtiger Elemente besprochen. Im MFA-Mikrocomputer-Baugruppensystem wird der Mikroprozessor "8085" eingesetzt, der neben den behandelten Registern (Akkumulator, Befehlsregister, Adreßregister und Statusregister) noch weitere besitzt, die jedoch für die grundsätzliche Arbeitsweise des Prozessors nicht von Bedeutung sind. Die Funktion dieser Register wird ausführlich in den "Anwender-Übungen FTÜ 20...." beschrieben. Im Theorieteil 2 wird nun auf die Baugruppe "Prozessor 8085" eingegangen. Hierbei werden einige Besonderheiten erklärt, die sich durch den Einsatz des o.g. Prozessors ergeben. Bild 7 zeigt die wichtigsten Elemente dieser Baugruppe (Prinzip). Vergleichen Sie die interne Struktur des Prozessors 8085 mit der in Bild 1 dargestellten Struktur eines allgemeinen Mikroprozessors.

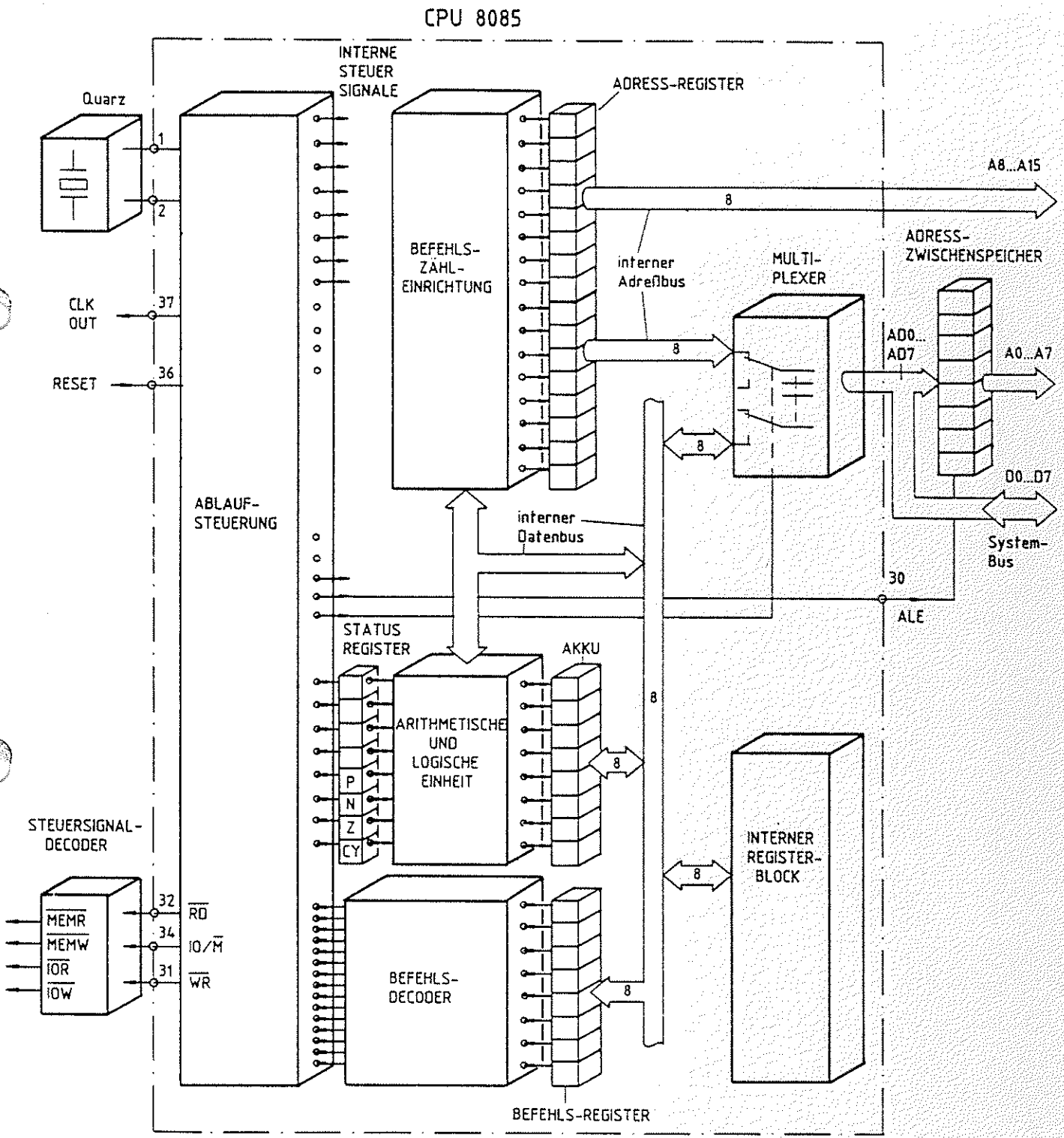


Bild 7: Baugruppe "Prozessor 8085" (Prinzip)

2.2. Die Erzeugung des Taktsignals

Für die Steuerung aller Abläufe innerhalb des Mikrocomputers ist die CPU verantwortlich. Dazu enthält sie einen internen Taktoszillator, dessen Frequenz von einem extern angeschlossenen Quarz (4-MHz) abgeleitet ist. Ein nachgeschalteter Schmitt-Trigger formt die sinusförmige Oszillatorspannung in ein Rechtecksignal um und ein Teiler halbiert die Oszillatorfrequenz. Den auf diese Weise gewonnenen 2-MHz-Takt nennt man Systemtakt. Bild 8 zeigt den für die Erzeugung des Systemtaktes wichtigen Schaltungsausschnitt und den Verlauf von sinusförmiger Oszillatorspannung und Rechteckspannung. Eine Periode der Rechteckspannung nennt man auch "Taktzyklus". Für Anwendungen innerhalb des Mikrocomputers steht der Systemtakt am CPU-Ausgang "CLK OUT" zur Verfügung.

Systemtakt

Taktzyklus

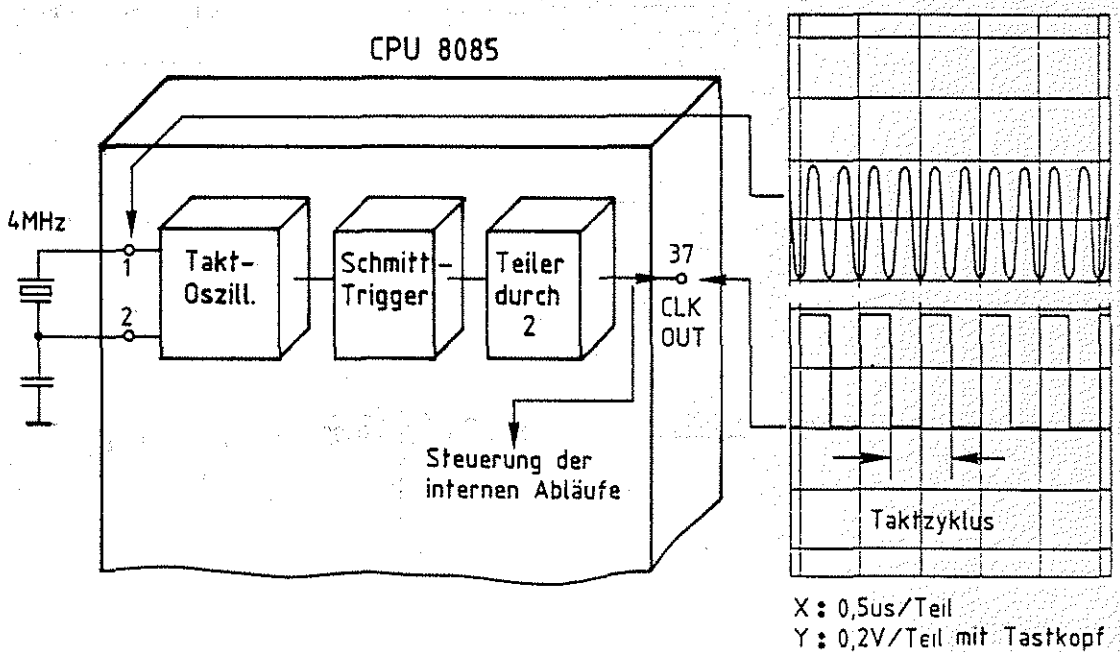


Bild 8: Takterzeugung und Darstellung des Taktes

2.3. Die Übertragung von Daten- und Adreßsignalen im "Zeitmultiplex-Betrieb"

Da die CPU 8085 einige Funktionseinheiten besitzt, die in anderen Prozessoren nicht vorhanden sind, die Anzahl ihrer Anschlüsse jedoch der anderer Prozessoren (40) entspricht, stehen für die Adreß- und Datensignale nicht wie üblich 24 (16 Adreß- und 8 Datenleitungen), sondern nur 16 Anschlußleitungen zur Verfügung. Damit über diese Leitungen trotzdem 16-Bit-Adressen und 8-Bit-Daten übertragen werden können, müssen einige Leitungen sowohl für den Daten- als auch für den Adreßverkehr verwendet werden. Dies geht nur, wenn man eine bestimmte Zeit lang alle 16 Leitungen für die Übertragung der 16 Adreßbits benutzt und danach 8 Leitungen für die Übertragung der 8 Datenbits. Dieses zeitlich versetzte Aussenden von Adreß- und Datensignalen nennt man "Zeitmultiplex-Betrieb". Zur Gewährleistung dieser Betriebsart wird innerhalb der CPU ein Umschalter (Multiplexer) benutzt. Er verbindet -gesteuert von der Ablaufsteuerung- 8 der 16 Leitungen für bestimmte Zeitabschnitte entweder mit dem CPU-internen Adreßbus oder mit dem CPU-internen Datenbus. Bild 7 zeigt die Anordnung dieses Multiplexers innerhalb der CPU (vergleichen Sie hierzu Bild 1). Die für Adreß- und Datensignale gemeinsam benutzten CPU-Leitungen sind durch die Bezeichnung AD0 bis AD7 gekennzeichnet. Über sie werden die niederwertigen Adreßbytes (A0 bis A7) und die Datenbytes übertragen.

Zeitmultiplexbetrieb
Multiplexer (Umschalter)

Welche Forderung ergibt sich nun aus dem Multiplexbetrieb an die Schaltung außerhalb der CPU? Zur Beantwortung dieser Frage soll zunächst mit Hilfe von Bild 9 daran erinnert werden, wie ein Prozessor ganz allgemein ein Datenbyte aus dem Speicher liest:

1. Speicheradresse auf den Adreßleitungen aussenden
2. Steuersignal $\overline{\text{MEMR}}$ erzeugen
3. Daten vom Datenbus in die CPU übernehmen

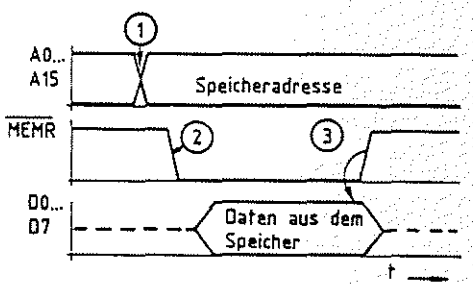


Bild 9: Lesen eines Datenbytes durch die CPU

Adreßzwischenspeicherung

Der zeitliche Verlauf der Signale zeigt, daß eine von der CPU ausgegebene Adresse solange auf dem Adreßbus zur Verfügung stehen muß, bis die CPU von der adressierten Speicherstelle Daten gelesen oder Daten an sie ausgegeben hat. Damit das niederwertige Adreßbyte auch während der Zeit des Datentransportes auf dem Adreßbus ansteht, muß es außerhalb der CPU zwischengespeichert werden. Der hierzu nötige Zwischenspeicher ist in Bild 7 schon berücksichtigt. Zur Übernahme des niederwertigen Adreßbytes in diesen Zwischenspeicher sendet die CPU jeweils bei Ausgabe einer neuen Adresse das Signal ALE aus. ALE steht für "Address-Latch-Enable" und bedeutet "Freigabe des Adressenspeichers". Mit diesem Signal muß die Übernahme der unteren acht Adreßbits in den Zwischenspeicher erfolgen. Bild 10 zeigt das Signal-Zeitdiagramm für diese Adreßzwischenspeicherung.

das ALE-Signal

1. Speicheradresse auf AD0-AD7 bzw. A8-A15 und ALE aussenden
2. Adreßteil A0-A7 in den Zwischenspeicher übernehmen
3. Steuersignal MEMR erzeugen
4. Daten vom Datenbus in die CPU übernehmen

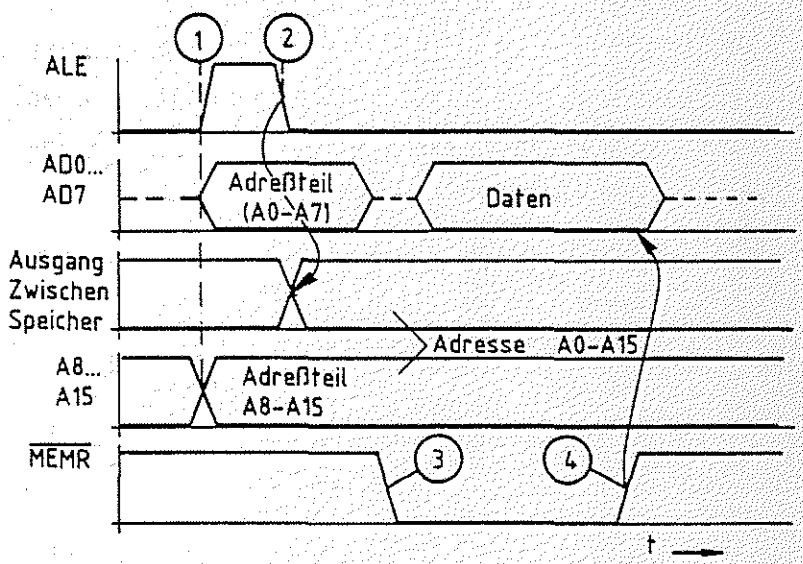


Bild 10: Lesen eines Datenbytes durch die CPU 8085 mit Adreßzwischenspeicherung

2.4. Die Erzeugung der Steuersignale $\overline{MEMR}$, $\overline{MEMW}$, $\overline{IOR}$ u. $\overline{IOW}$

die CPU-Steuerleitungen, $\overline{RD}$, $\overline{WR}$ u. IO/$\overline{M}$

Nachdem die CPU die Adresse für eine Speicherzeile oder eine Ein- oder Ausgabebaugruppe ausgegeben hat, zeigt sie mit den Pegeln auf den Steuerleitungen an, ob sie Daten in die adressierte Baugruppe schreiben, oder welche aus ihr lesen will. Hierzu stehen der CPU 8085 drei Leitungen zur Verfügung. Diese haben die Bezeichnungen $\overline{RD}$ (Read, Lesen), $\overline{WR}$ (Write, Schreiben) und IO/$\overline{M}$ (Input, Output/Memory, Ein-Ausgabe/Speicher).

Die folgende Tabelle (Bild 11) zeigt eine Zusammenstellung der verschiedenen CPU-Aktivitäten mit den zugehörigen Pegeln auf den drei Leitungen. Aus diesen drei Signalen müssen mit Hilfe eines Decoders die vier Steuersignale $\overline{MEMR}$, $\overline{MEMW}$, $\overline{IOR}$ und $\overline{IOW}$ abgeleitet werden.

CPU-Aktivität		IO/$\overline{M}$	$\overline{WR}$	$\overline{RD}$
Speicher lesen	($\overline{MEMR}$)	L	H	L
Speicher schreiben	($\overline{MEMW}$)	L	L	H
Ein/Ausgabe lesen	($\overline{IOR}$)	H	H	L
Ein/Ausgabe schreiben	($\overline{IOW}$)	H	L	H

Bild 11: CPU-Aktivitäten und Steuersignale $\overline{RD}$, $\overline{WR}$ u. IO/$\overline{M}$

Das Prinzip der Decodierungsschaltung und die zugehörige Funktionstabelle ist in Bild 12 dargestellt.

Eingänge			Ausgänge			
$\overline{WR}$	IO/$\overline{M}$	$\overline{RD}$	$\overline{MEMW}$	$\overline{IOW}$	$\overline{MEMR}$	$\overline{IOR}$
L	L	H	L	H	H	H
L	H	H	H	L	H	H
H	L	L	H	H	L	H
H	H	L	H	H	H	L

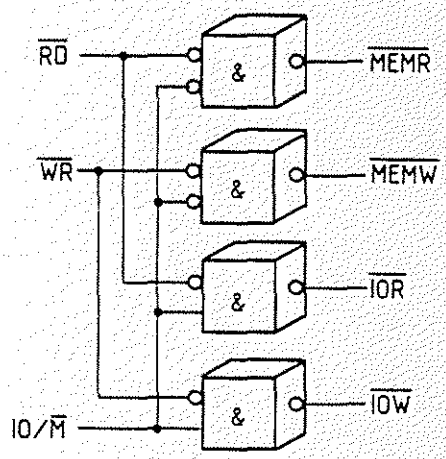


Bild 12: Decodierung der Steuersignale $\overline{RD}$, $\overline{WR}$ u. IO/$\overline{M}$ zur Erzeugung der System-Steuersignale mit zugehöriger Funktionstabelle

2.5. Die Bearbeitung eines Befehls

Im Theorieteil 1 wurde erklärt, daß die Bearbeitung eines Befehls in zwei Phasen durchgeführt wird (Bild 2). In der Befehlsholphase wird der Befehl zunächst aus dem Speicher gelesen und entschlüsselt. In der Befehlsausführungsphase erfolgt dann die Ausführung. Beide Phasen stehen in zeitlichem Zusammenhang zum Systemtakt. Bild 13 zeigt anhand des 2-Byte-Befehls "IN 12" (Daten des Eingabe-Ports mit der Port-Nr. 12 lesen) den Signalverlauf für die Ausführung des gesamten Befehls. Diesen Ablauf nennt man auch "Befehlszyklus".

Befehlszyklus

Es wird angenommen, daß das Befehlsbyte unter der Speicheradresse 0000H gespeichert ist und daß die Daten am Eingabe-Port durch Einstellung der Schalter den Wert E7H haben.

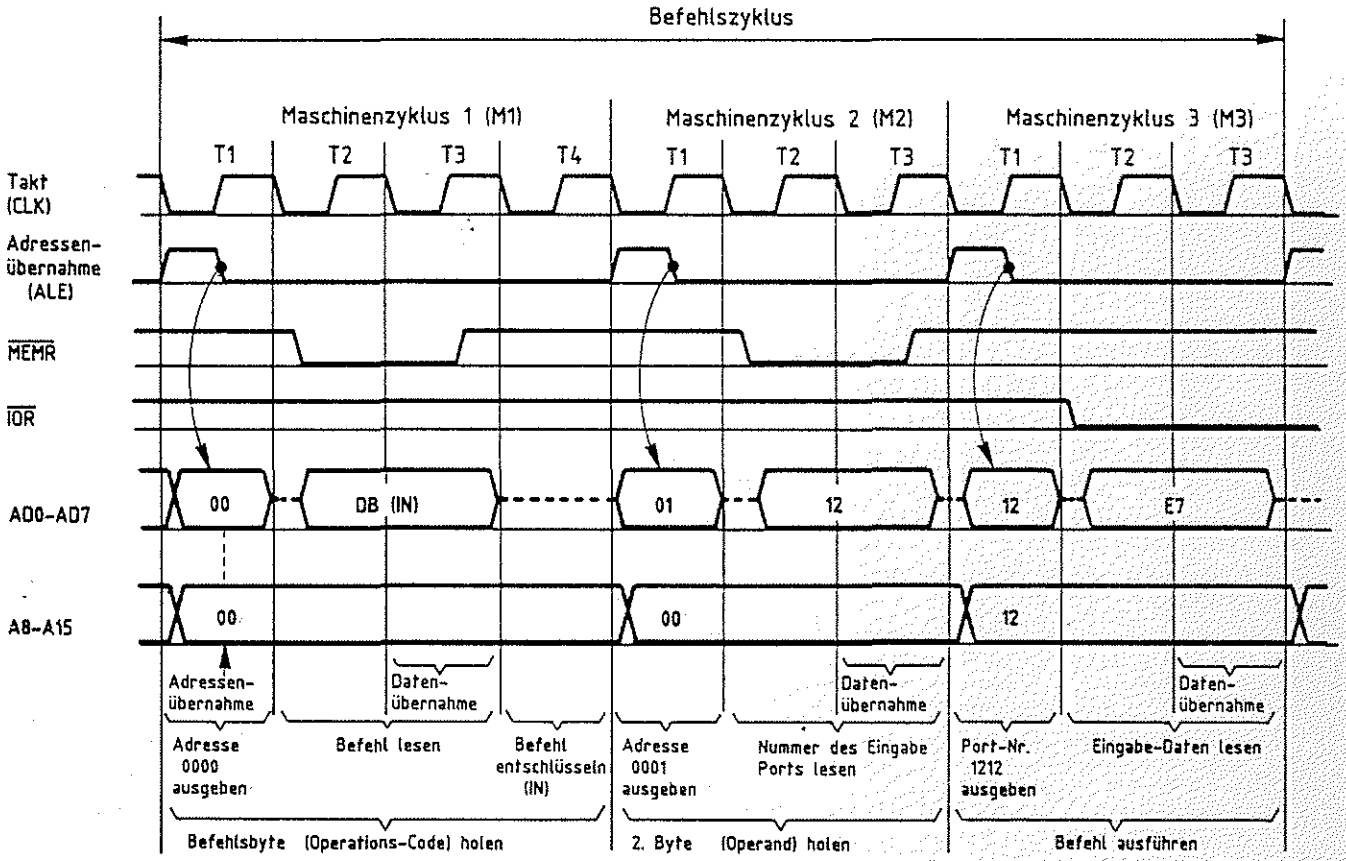


Bild 13: Verlauf der Steuer-, Adreß- und Datensignale für den Befehl "IN12"

Beschreibung des Signalverlaufes:

— Maschinenzyklus 1 (M1)

T1: Innerhalb dieser Zeit gibt der Prozessor die Adresse der Speicherzeile aus, die das zu bearbeitende Befehlsbyte enthält. Durch den Multiplexbetrieb steht das niederwertige Adreß-Byte auf den Leitungen AD0 bis AD7 für etwa einen Taktzyklus zur Verfügung, während das höherwertige Adreß-Byte alle vier Taktzyklen ansteht. Mit der fallenden Flanke des von der CPU ausgesendeten ALE-Signals wird das niederwertige Adreß-Byte außerhalb der CPU zwischengespeichert.

nieder- und höherwertige Adreßbytes

T2: Das Steuersignal "Speicher lesen" ($\overline{MEMR}$) wird auf L-Pegel geschaltet. Hierdurch wird der adressierte Speicher (Speicherstelle 0000) veranlaßt, Daten auf den Datenbus zu schalten. Auf den Adreß-Datenleitungen AD0 bis AD7 steht somit etwas verzögert (Laufzeiten) das Datenwort (DB) der adressierten Speicherzeile zur Verfügung.

T3: Innerhalb von T3 liest der Prozessor dieses Datenwort und speichert es in seinem Befehlsregister ab. Wenn das Steuersignal $\overline{MEMR}$ wieder H-Pegel führt, ist der angesprochene Speicherbaustein wieder gesperrt.

T4: In diesem Taktzyklus entschlüsselt die CPU das zuvor gelesene Befehlsbyte, d.h., sie stellt fest, um welchen der ihr "bekannten" Befehle es sich handelt. Aus dem Ergebnis dieser Entschlüsselung leitet sie die notwendigen Aktivitäten zur Ausführung des Befehls ab.

Die weitere Bearbeitung des Befehls hängt nun davon ab, welchen Befehl die CPU erhalten hat. In unserem Beispiel ist es ein Zwei-Byte-Befehl, der das Einholen eines weiteren Bytes erfordert, ehe er ausgeführt werden kann. Bei einem Drei-Byte-Befehl müßten noch zwei weitere Bytes aus dem Speicher gelesen werden.

Maschinenzyklus

Die einzelnen Schritte - Befehls-Byte lesen, zweites bzw. drittes Byte lesen und Befehl ausführen - nennt man "Maschinenzyklen" und bezeichnet sie mit M1, M2, M3 usw.. Der Maschinenzyklus M1 zum Einholen des Befehlsbytes besteht (bis auf wenige Ausnahmen) immer aus vier Taktzyklen. Die Anzahl der Taktzyklen der nachfolgenden Maschinenzyklen (M2, M3 usw.) hängt vom jeweiligen Befehl ab.

— Maschinenzyklus 2 (M2)

Die Nummer des Eingabe-Ports (der Operand) wird von der dem Befehlsbyte folgenden Speicherstelle (0001H) gelesen.

T1: Die nächste Speicheradresse (0001H), in der die Nummer des zu lesenden Ports gespeichert sein muß, wird ausgegeben. Mit Hilfe des ALE-Signals wird wieder das niederwertige Adreß-Byte der Speicheradresse (01H) außerhalb der CPU zwischengespeichert.

T2 u. T3: Die CPU liest erneut mit dem Steuersignal $\overline{MEMR}$ den Speicherinhalt (12H), der die Nummer des Eingabe-Ports darstellt und speichert ihn intern ab.

– Maschinenzyklus 3 (M3)

Der Befehl wird ausgeführt, d.h., es muß der Signalzustand an den Eingängen des Eingabe-Ports mit der Nr. 12 gelesen werden.

T1: Die CPU schaltet die zuvor gelesene Port-Nummer (12H) des Eingabe-Ports auf den Adreßbus und zwar sowohl auf die Leitungen AD0 bis AD7 als auch auf die Leitungen A8 bis A15. Mit dem ALE-Signal wird das niederwertige Adreß-Byte außerhalb der CPU zwischengespeichert.

T2: Nun veranlaßt die CPU mit dem Steuersignal $\overline{IOR}$ das adressierte Eingabe-Port, den Eingangssignalzustand (E7) auf den Datenbus zu schalten.

T3: Während des Taktzyklusses T3 wird dieses Datenwort mit der ansteigenden Flanke von $\overline{IOR}$ in den Akku der CPU übernommen. Wenn das Steuersignal $\overline{IOR}$ wieder H-Pegel führt, ist das Eingabe-Port vom Datenbus getrennt.

Der gesamte Befehlszyklus für den Befehl "IN" teilt sich damit in drei Maschinenzyklen und insgesamt 10 Taktzyklen auf. Zu Beginn eines jeden Maschinenzyklusses sendet die CPU für einen halben Taktzyklus den ALE-Impuls aus.

Zeit zur Bearbeitung eines Befehls

Bei einer Taktzykluszeit von 500 ns (2-MHz-Takt) benötigt der Prozessor für die Bearbeitung des Befehls 5 µs. Kennt man die Zahl der Taktzyklen für die verschiedenen CPU-Befehle, so kann man die Gesamtzeit berechnen, die der Prozessor zur Bearbeitung eines Programms benötigt.

Arbeitsgeschwindigkeit eines Mikrocomputers

Den beschriebenen Vorgang (Lesen und Ausführen eines Befehls) wiederholt der Prozessor für jeden auszuführenden Programmbefehl. Die dabei ausgelösten Funktionsschritte sind von der Art des Befehls abhängig. Somit ist auch die Bearbeitungszeit befehlsabhängig. Geht man bei diesem Prozessor von einer mittleren Bearbeitungszeit von ca. 8 µs für einen Befehl aus, so kann der Prozessor in einer Sekunde ca. 125.000 Befehle abarbeiten. Diese Zahl offenbahrt die Arbeitsgeschwindigkeit eines Mikrocomputers.

2.6. Zusammenfassung

Der hier beschriebene Aufbau einer Prozessorbaugruppe bezieht sich speziell auf den Mikroprozessor 8085. Andere Prozessortypen erfordern entsprechend andere Schaltungskomponenten. Obwohl bei allen Prozessortypen die grundsätzliche Arbeitsweise für die Bearbeitung eines Befehls gleich ist, können sie sich doch in den Steuersignalen und der Taktsteuerung unterscheiden. Je nach der Ausbaustufe eines Mikrocomputers wird man insbesondere die Adreß- Daten- und Steuersignale eines Prozessors über spezielle Treiberbausteine betreiben, die auch auf der hier beschriebenen Baugruppe zum Einsatz kommen. Da sie die Signale jedoch nicht verändern und für das Verständnis der Funktion unwesentlich sind, werden sie in den Erklärungen nicht berücksichtigt. Näheres hierzu können Sie aber in der Fachpraktischen Übung BFZ/MFA 2.1. nachlesen.

Treiber für Adreß- Daten- u. Steuersignale

2.7. Messungen mit dem Oszilloskop am Mikrocomputer-Bus

Sollen zur Überprüfung der Arbeitsweise eines Mikrocomputers mit einem Oszilloskop Messungen durchgeführt werden, so ist zweierlei zu beachten:

- die Signalzustände der darzustellenden Signale müssen sich periodisch wiederholen, um ein stehendes Bild auf dem Oszilloskop zu erhalten
- durch ein geeignetes Triggersignal muß dafür gesorgt werden, daß der Zeitbezug, der zwischen den zu messenden Signalen vorhanden ist, bei der schrittweisen Aufnahme der Oszillogramme erhalten bleibt.

2.7.1. Periodische Signalzustände (Zyklische Programme)

Daher muß die CPU mit Hilfe eines Programms veranlaßt werden, eine sich periodisch wiederholende Befehlsfolge zu durchlaufen. Eine solche Befehlsfolge nennt man Programmschleife oder kurz Schleife. Im einfachsten Fall kann eine Schleife aus einem einzigen Befehl im Programmspeicher bestehen, nämlich dem Drei-Byte-Befehl JMP 0000 (C3 00 00). Steht dieser Befehl unter der Adresse 0000 im Speicher, so setzt der Prozessor mit der Ausführung dieses Sprungbefehls die Befehlsbearbeitung stets bei dieser Adresse fort (unendliche Schleife). Längere

Programmschleife zur Verfolgung der Arbeitsweise der CPU mit dem Oszilloskop

Programme werden in ähnlicher Weise periodisch wiederholt, wenn am Programmende ein Rücksprungbefehl zum Programmanfang eingefügt wird.

2.7.2. Ableitung von Triggersignalen

Alle Signale auf den Adreß- Daten- und Steuerleitungen stehen zeitlich zueinander in Bezug. Dieser Zeitbezug wird aber nicht dargestellt, wenn man mit einem Oszilloskop die Signalverläufe auf den Leitungen nacheinander aufnimmt (und sie z.B. in ein Signal-Zeit-Diagramm einträgt). Grund hierfür ist die Triggereinrichtung des Oszilloskops, die ihren Triggerimpuls intern vom jeweils gemessenen Signal ableitet. Bild 14 zeigt, wie z.B. die beiden Adreßsignale A0 und A1 auf dem Bildschirm des Oszilloskops angezeigt werden können.

Triggerimpuls

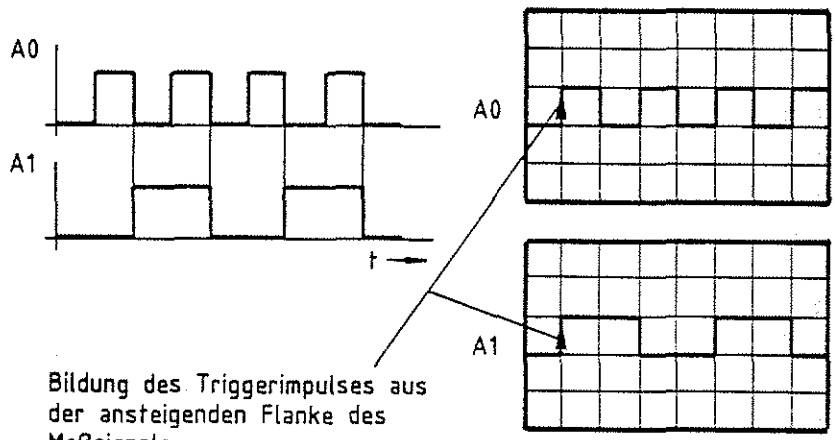


Bild 14: Darstellung von Signalen ohne deren Zeitbezug

Um nun den Zeitbezug in der Darstellung herzustellen, muß man dafür sorgen, daß alle aufgenommenen Signale auf dem Bildschirm den gleichen Zeitbeginn haben. Dies erreicht man durch Ableitung eines Triggerimpulses aus dem zu untersuchenden System, mit dem der Schreibvorgang auf dem Bildschirm ausgelöst werden muß (Externe Triggerung). In Mikrocomputer-Systemen wird stets mit der Änderung von Adreßsignalen ein neuer Befehlszyklus eingeleitet, so daß man häufig von diesen Adreßsignalen ein Triggersignal ableitet. Dazu bedient man sich eines Adreßvergleichers, wie Sie ihn bei den Speicher- und Ein/Ausgabebaugruppen kennengelernt haben. Bild 15 zeigt das Prinzip der Schaltung.

Ableitung eines Triggerimpulses aus den Systemsignalen

Externe Triggerung

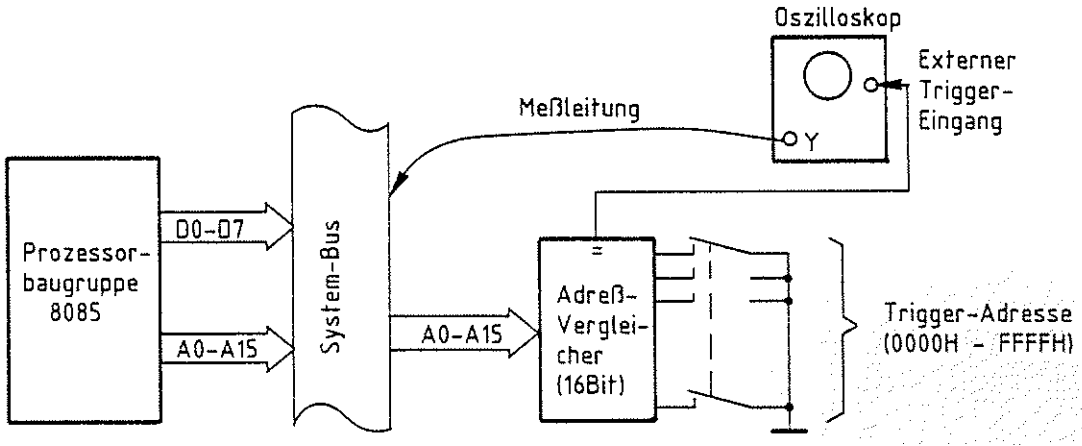


Bild 15: Ableitung eines Triggersignals durch Adreßvergleich

Stimmt die an den Schaltern vorgewählte Adresse mit dem Signalzustand auf dem Adreßbus überein, so liefert der Adreßvergleicher ein Ausgangssignal, durch das der Schreibvorgang auf dem Oszilloskop ausgelöst wird. Nun kann nacheinander Signal für Signal auf dem Systembus aufgenommen werden, wobei alle Signale den gleichen zeitlichen Beginn haben. Durch Ändern der Schalterstellung, d.h. durch Ändern der Triggeradresse kann dieser Zeitpunkt verändert werden. Dies entspricht bildlich gesehen einer Verschiebung des Signalausschnittes auf der Zeitachse. Die Art dieser Triggerung nennt man Worttriggerung. Sie kommt auch bei allen Logikanalysatoren zur Anwendung, mit denen gleichzeitig die Signalverläufe mehrerer Signale aufgenommen werden können. Logikanalysatoren geben aber nicht den exakten Signalverlauf wieder, sondern sie zeigen lediglich an, ob ein Signal im jeweils betrachteten Zeitbereich H- oder L-Pegel führt. Bild 16 zeigt die Signaldarstellung eines solchen Logikanalysators.

Triggeradresse wählt Bildausschnitt

Worttriggerung Logikanalysator

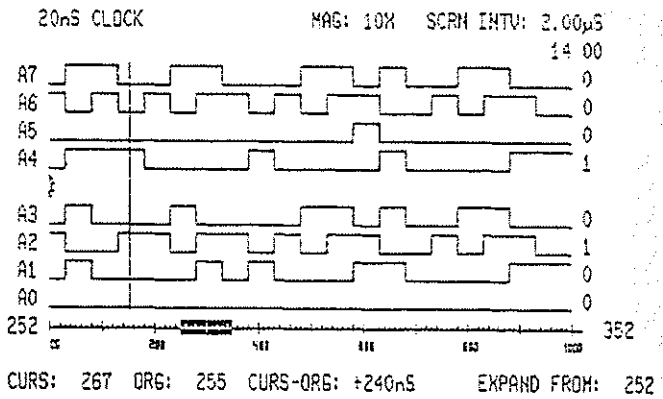


Bild 16: Diagramm eines Logikanalysators

2.8. Untersuchung der Funktion der Befehlszähleinrichtung im "Free-Run-Mode"

Free-Run-Mode

Um überprüfen zu können, ob der Prozessor nach einem RESET nacheinander Befehl für Befehl aus dem Speicher liest und ausführt, müßte ein vollständiger Speicherausbau zur Verfügung stehen.
Um dies zu umgehen, kann man den Prozessor durch einen Trick überlisten, indem ihm in jeder Befehlsholphase ein Befehl vorgetäuscht wird. Dies erreicht man, wie in Bild 17 dargestellt, durch eine Brücke zwischen der Steuerleitung $\overline{MEMR}$ und der Datenleitung D7.

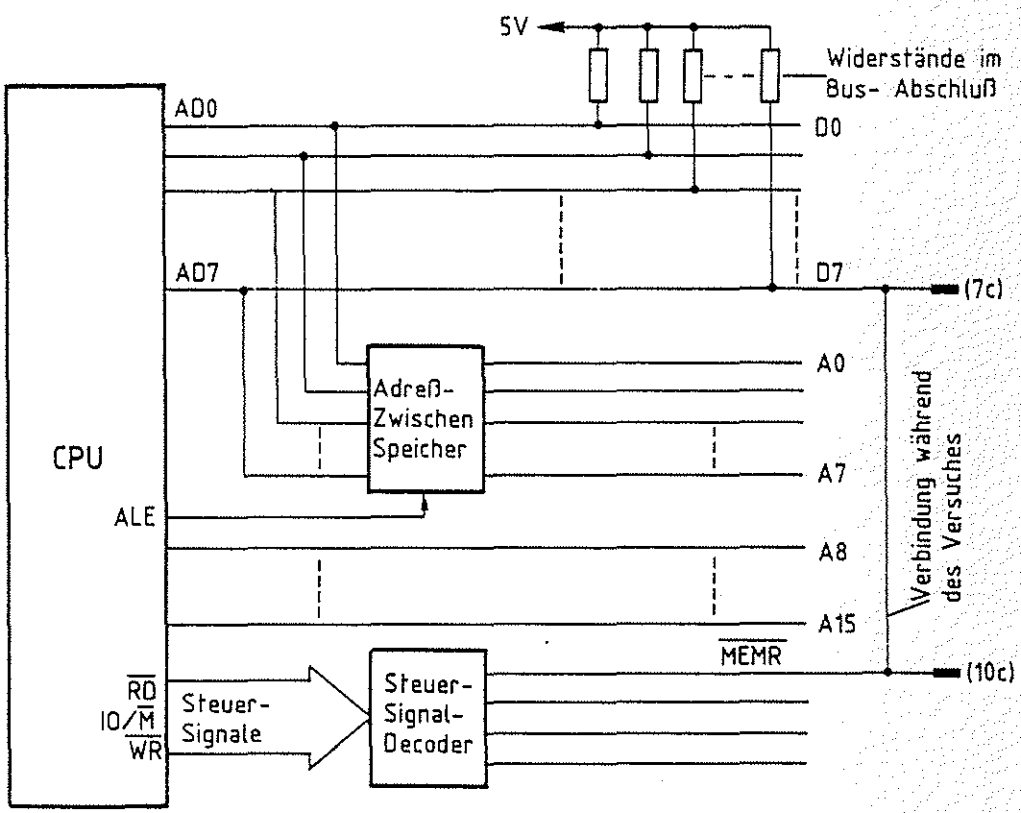


Bild 17: Prozessor im Free-Run-Mode

Beginnt der Prozessor nach einem RESET mit dem Lesen des ersten Befehls, so erhält er mit dem Steuersignal $\overline{MEMR}$ (L-Pegel) das Befehlsbyte 0111 1111 (7FH), weil lediglich die Datenleitung D7 durch das aktive Steuersignal $\overline{MEMR}$ L-Pegel führt, während alle anderen Datenleitungen über die Pull-up-Widerstände (auf dem Busabschluß) auf H-Pegel liegen. Das gelesene Befehlsbyte ist ein 1-Byte-Befehl (MOV A,A), der im Prozessor keine

Veränderungen zur Folge hat.
Der Vorteil dieser Vorgehensweise ist, daß der Prozessor unabhängig von der ausgesendeten Adresse stets den gleichen Befehl erhält und den Befehlszähler nach jedem gelesenen Befehl um Eins erhöht (Ein-Byte-Befehl).
Mit dem Oszilloskop kann man nun die Signale auf den Adreßleitungen A0 bis A15 und das Zählerverhalten des Programmzählers untersuchen. Dieses Verfahren nennt man Free-Run-Mode.
Es wird häufig bei der Funktionskontrolle und auch in Verbindung mit speziellen Meßgeräten (Signaturanalysatoren) angewendet.

FACHTHEORETISCHE ÜBUNG MIKROCOMPUTER – TECHNIK

MIKROPROZESSOR-MIKROCOMPUTER

BFZ/MFA 10.4.

ÜBUNGSTEIL 2

Übungsteil 2

Im Übungsteil 2 werden Sie u.a. die Bearbeitung eines kleinen Programms durch den Prozessor mit einem Oszilloskop beobachten. Hierzu wird ein Programm mit dem Bus-Signalgeber in den RAM-Speicher geladen. Das zur Triggerung des Oszilloskops notwendige Signal liefert der Adreßvergleicher, der sich auf der Bus-Signalanzeige befindet.
Zur Durchführung der Übung benötigen Sie:

- 1 Baugruppenträger mit Busverdrahtung (BFZ/MFA 0.1.)
- 1 Bus-Abschluß (BFZ/MFA 0.2.)
- 1 Trafo-Einschub (BFZ/MFA 1.1.)
- 1 Spannungsregelung (BFZ/MFA 1.2.)

} zusammengebaut und geprüft nach FPÜ BFZ/MFA 1.2. Arbeitsblatt A7

- 1 Prozessor 8085 (BFZ/MFA 2.1.)
- 1 8-K-RAM/EPROM (BFZ/MFA 3.1.) bestückt mit mind. 2-K-RAM
- 1 Bus-Signalgeber (BFZ/MFA 5.1.)
- 1 Bus-Signalanzeige (BFZ/MFA 5.2.)
- 1 Adapter 64polig (BFZ/MFA 5.3.)
- 1 Ein- oder Zweikanal-Oszilloskop mit externem Triggereingang und Tastköpfen 10:1

Allgemeine Hinweise zur Durchführung der Übungen:

- Die Einschübe dürfen nur bei abgeschalteter Betriebsspannung gesteckt oder gezogen werden
- Aufgrund der Busverdrahtung können die Baugruppen in beliebige Steckplätze gesteckt werden
- Den logischen Signalen "0" und "1" sind die folgenden Pegel zugeordnet:

  log. "0" ≙ 0...0,8 V (LOW)
  log. "1" ≙ 2,4...5 V (HIGH)

- Alle zur Messung an den Baugruppen vorgegebenen Arbeitsblätter enthalten:
  = Angaben über den Sinn der jeweiligen Messung
  = Angaben über einzustellende Bedingungen
  = Aufgabenstellungen, ggf. mit Hinweisen zu möglichen Fehlern
  = Diskussion der Meßergebnisse

Bedienungshinweise:

Prozessor 8085:

Die Prozessor-Baugruppe ist mit dem Mikroprozessor 8085 aufgebaut. In der Frontplatte der Baugruppe befindet sich ein RESET-Taster, über den der Prozessor in den Grundzustand (Befehlszähleinrichtung auf 0000) gebracht werden kann. Dieser Grundzustand wird auch mit dem Einschalten der Betriebsspannung eingenommen.

Bus-Signalgeber:

Der Bus-Signalgeber wird in dieser Übung dazu benötigt, um ein Programm in den RAM-Speicher zu laden. Da sich hierbei gleichzeitig die Prozessor-Baugruppe am System-Bus befindet, muß diese während der Programmeingabe vom Bus getrennt werden. Dies erfolgt über den Schalter ON/OFF am Bus-Signalgeber.

Stellung ON : Bus-Signalgeber frei, Prozessor gesperrt

Stellung OFF : Bus-Signalgeber gesperrt, Prozessor frei

An den Codierschaltern des Bus-Signalgebers werden die Speicheradressen und die Daten eingestellt und durch Betätigen der Steuersignal-Taste MEMW in den Speicher geladen. Nach Eingabe des vollständigen Programms in den Speicher wird der Schalter ON/OFF in die Stellung OFF gebracht. Hierdurch wird u.a. am Prozessor ein RESET ausgelöst, so daß Programmbearbeitung durch den Prozessor bei der Speicheradresse 0000H beginnt.

Bus-Signalanzeige:

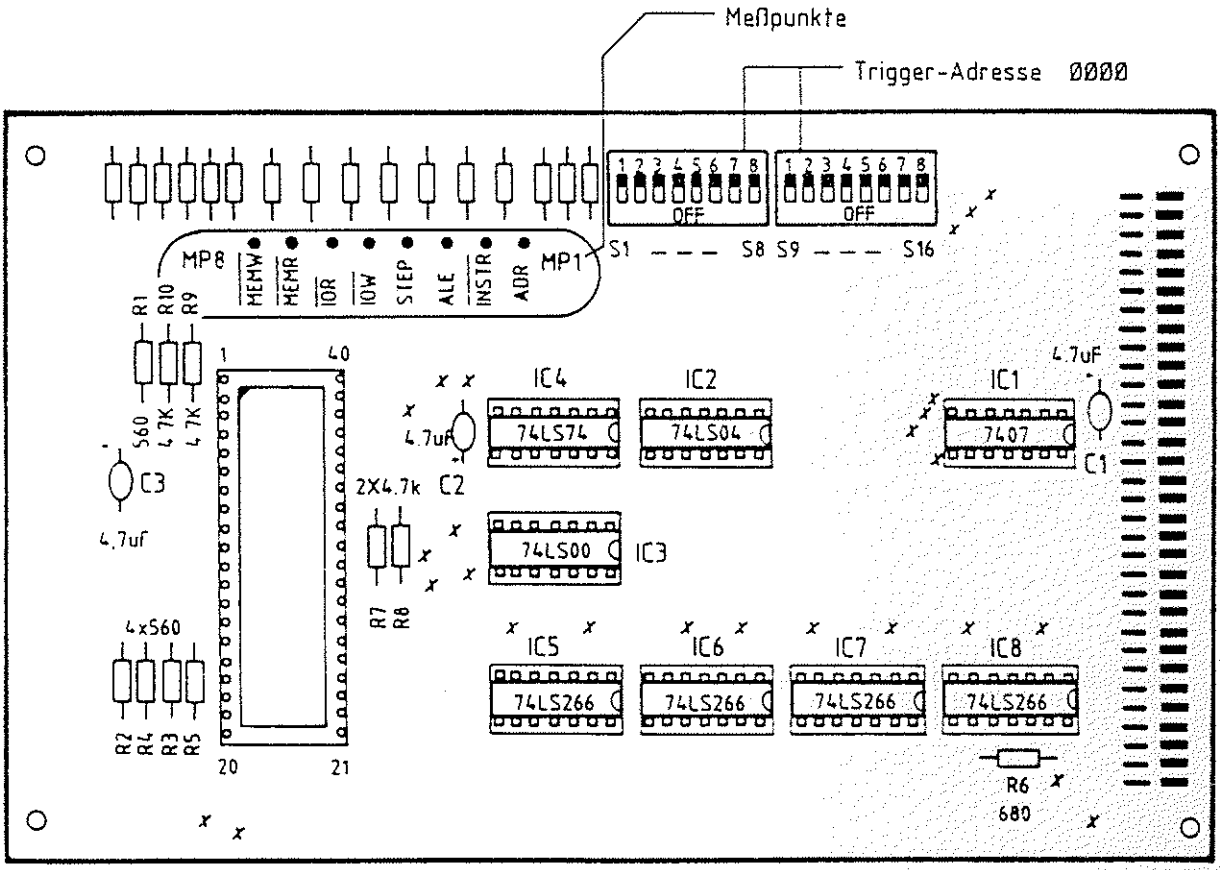
Schalter RUN/HLT und Taster STEP werden für den Einzelschrittbetrieb verwendet. Da der Prozessor in den folgenden Arbeitsschritten mit normaler Arbeitsgeschwindigkeit arbeiten soll, muß der Schalter RUN/HLT in die Stellung RUN gebracht werden.

Der Umschalter ADDR.STOP ON/OFF, über den der Einzelschrittbetrieb beim Auftreten einer ganz bestimmten Adresse auf dem Adreß-Bus aktiviert werden kann, bleibt bei den folgenden Messungen in der Stellung OFF.

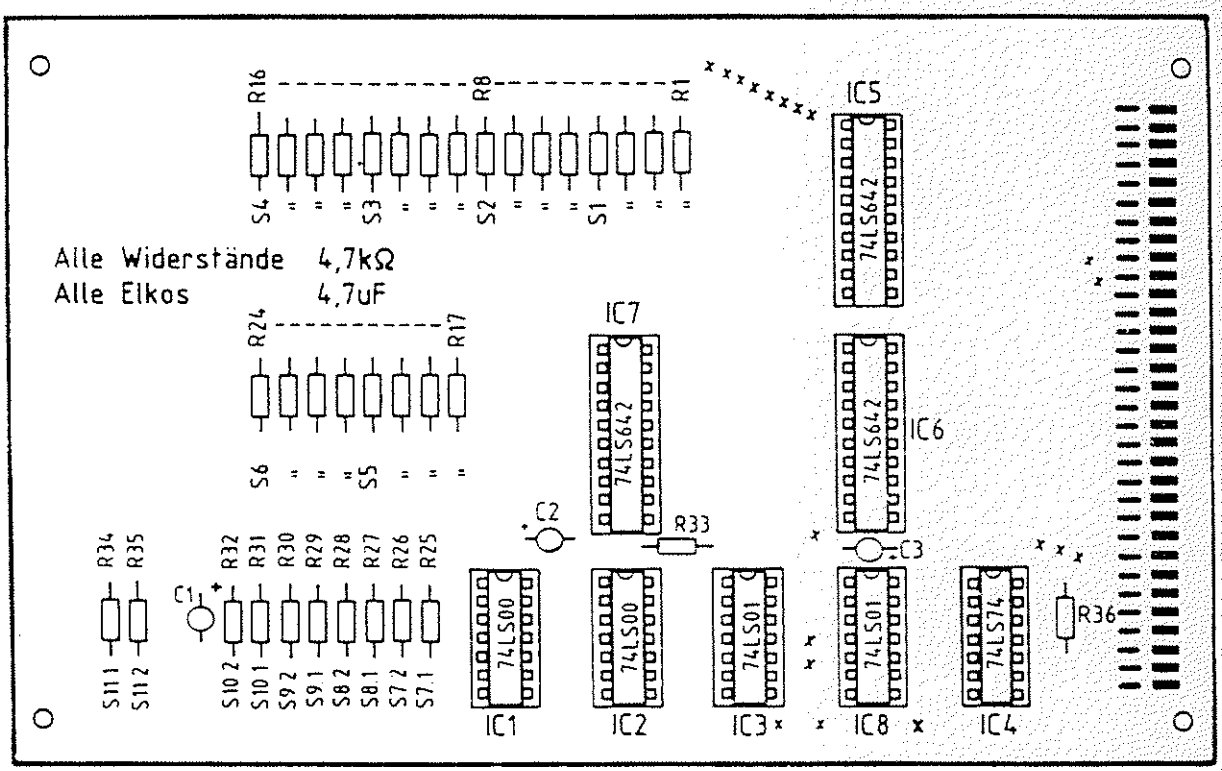
Auf der Baugruppe befinden sich einige Meßstifte mit Signalen, die Sie oszilloskopieren sollen. Deshalb ist die Baugruppe über eine Adapterkarte zu betreiben. Die 16 DIL-Schalter auf der Baugruppe müssen auf Stellung ON geschaltet werden. Hierdurch erscheint am Meßpunkt MP1 (ADR) immer dann ein H-Signal, wenn auf dem Adreßbus die Adresse 0000 ansteht (für Triggerungen).

Die folgenden beiden Seiten zeigen zu Ihrer Information die Bestückungspläne aller verwendeten Baugruppen.

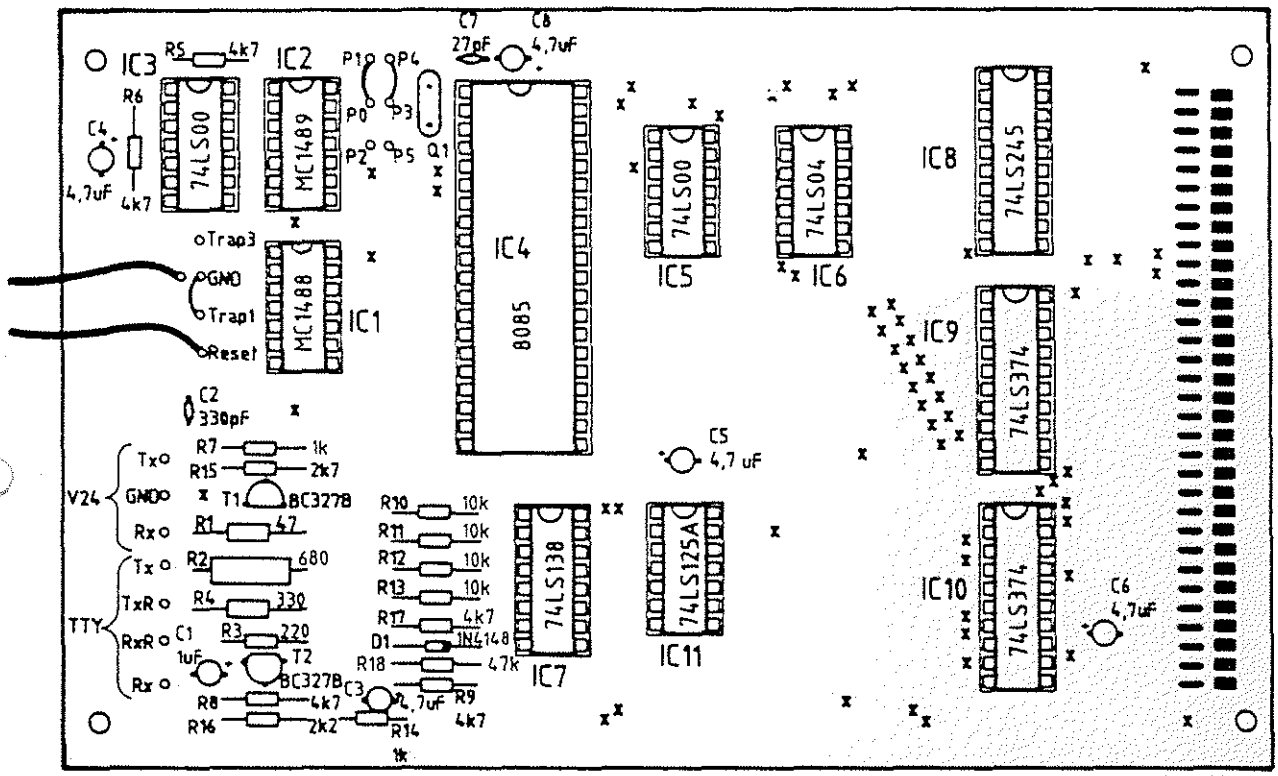
Bus-Signalanzeige



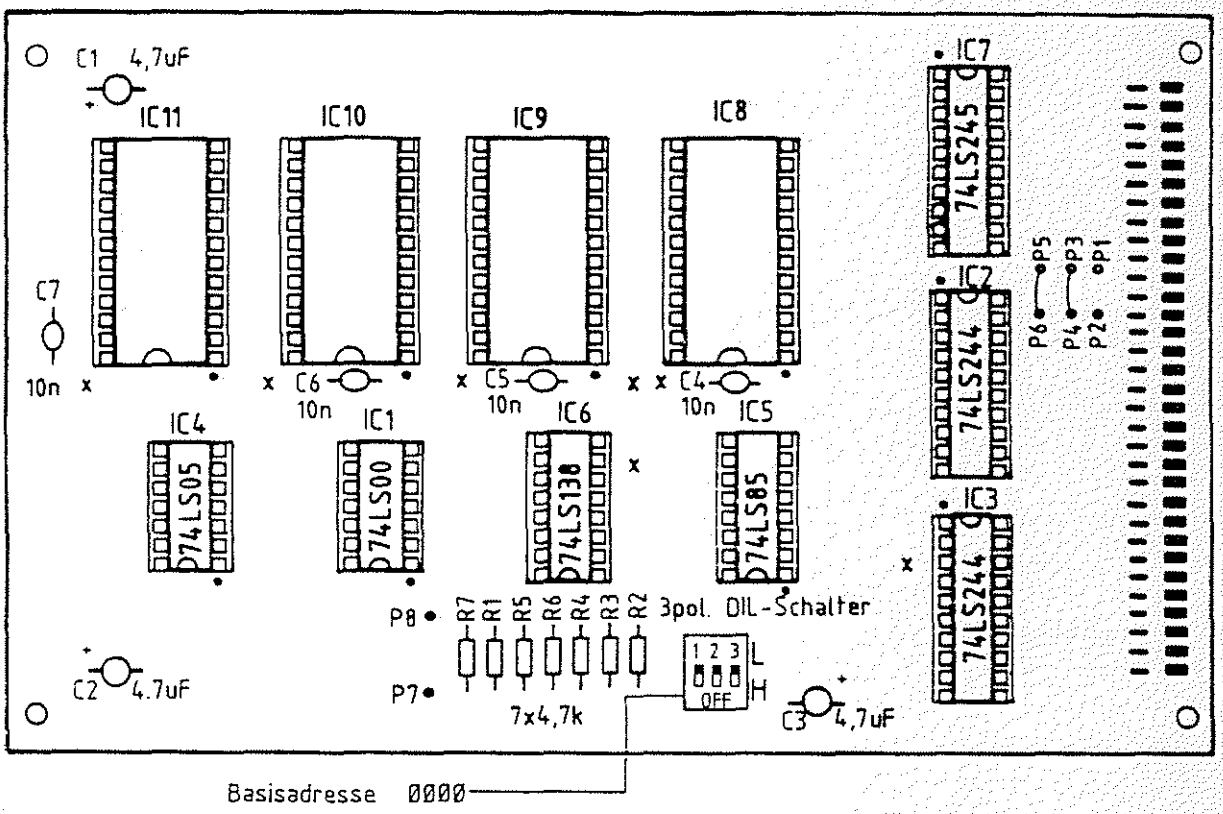
Bus-Signalgeber



Prozessor 8085



RAM - Baugruppe



Mikroprozessor-Mikrocomputer

Name:

Übungsteil 2

Datum:

A1

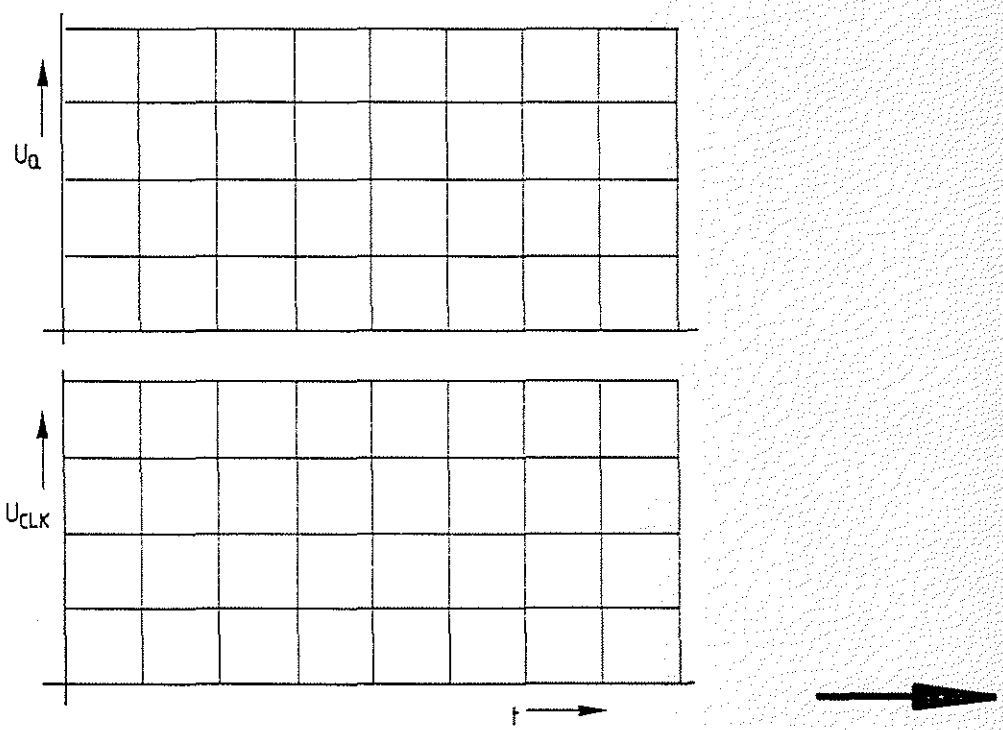
Messen der Quarzfrequenz und des Systemtaktes mit dem Oszilloskop

Insbesondere im Fehlerfall oder bei der Inbetriebnahme müssen an der Prozessorbaugruppe meßtechnische Untersuchungen vorgenommen werden, anhand derer man die grundsätzliche Arbeitsweise beurteilen kann. Hierzu gehört als erstes die Überprüfung des Systemtaktes.

Stecken Sie die Baugruppe "Prozessor 8085" über die Adapterkarte in den Baugruppenträger und schalten Sie die Betriebsspannung ein. Messen Sie mit dem Oszilloskop (mit 10:1-Tastkopf) die Quarzfrequenz am Pin 1 der CPU und die Frequenz des Systemtaktes am Pin 37. Tragen Sie Ihre Meßwerte in die folgende Tabelle ein.

	Quarzfrequenz (1)	Systemfrequenz (37)
Ihre Meßwerte		
Kontroll - Werte	4 MHz	2 MHz

Skizzieren Sie den Verlauf der beiden Spannungen in zeitlichem Zusammenhang



Mikroprozessor-Mikrocomputer Name:

Übungsteil 2 Datum:

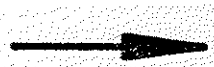
A2.1

Prüfung der Prozessorbaugruppe im "Free-Run-Mode"

Verbinden Sie (am besten auf der Adapterkarte) die Busleitungen 10c ($\overline{\text{MEMR}}$) und 7c (D7).
Stecken Sie zusätzlich die Bus-Signalanzeige in den Baugruppenträger;

Schalter RUN/HLT ⟶ RUN
Schalter ADDR.STOP ⟶ OFF

Anweisungen	Anzeigen; Bemerkungen
Betriebsspannung Ein.	Alle Leuchtpunkte der ADDRESS-Anzeige der Bus-Signalanzeige leuchten; dies ist ein Zeichen dafür, daß sich die Signalzustände auf den Adreß-Leitungen ständig ändern, und daß die CPU grundsätzlich arbeitet. LED's MEMR und INSTR leuchten; die CPU liest "Daten" und interpretiert diese als Befehl. Alle Leuchtpunkte der DATA-Anzeige leuchten; die Datenwerte ändern sich ständig.
Schalter RUN-HLT auf HLT. Mehrmals STEP betätigen.	CPU "bleibt stehen", die ADDRESS-Anzeige zeigt eine "feste" Adresse an; ihr Wert hängt vom Zufall ab. ADDRESS-Anzeige wird mit jedem STEP um Eins erhöht.
RESET betätigen	ADDRESS-Anzeige springt auf 0000; die RESET-Funktion der CPU arbeitet richtig.
Schalter RUN-HLT auf RUN	Die CPU arbeitet mit Normalgeschwindigkeit (Anzeigen wie oben).



Mikroprozessor-Mikrocomputer

Name:

Übungsteil 2

Datum:

A2.2

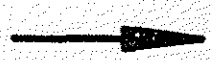
Messen Sie mit einem Oszilloskop die Periodendauer der Spannungen an den Adreßleitungen A0 bis A13. Rechnen Sie diese dann in die jeweilige Frequenz um.

Adreßleitung	Pin	T	f
A0	16c		
A1	17a		
A2	17c		
A3	18a		
A4	18c		
A5	19a		
A6	19c		
A7	20a		
A8	20c		
A9	21a		
A10	21c		
A11	22a		
A12	22c		
A13	23a		

Oszilloskopieren Sie (zeitrichtig) die Signale folgender Adreßleitungen und tragen Sie die Signalverläufe in das vorbereitete Diagramm ein.

Signal	Messbar an ...	Bemerkungen
ALE	MP3-Signalanzeige	Kanal 1
A0	16c-Adapter	Kanal 2
A1	17a-Adapter	Kanal 2
A2	17c-Adapter	Kanal 2
U_{MP1}	MP1-Signalanzeige	Ext. Trigger

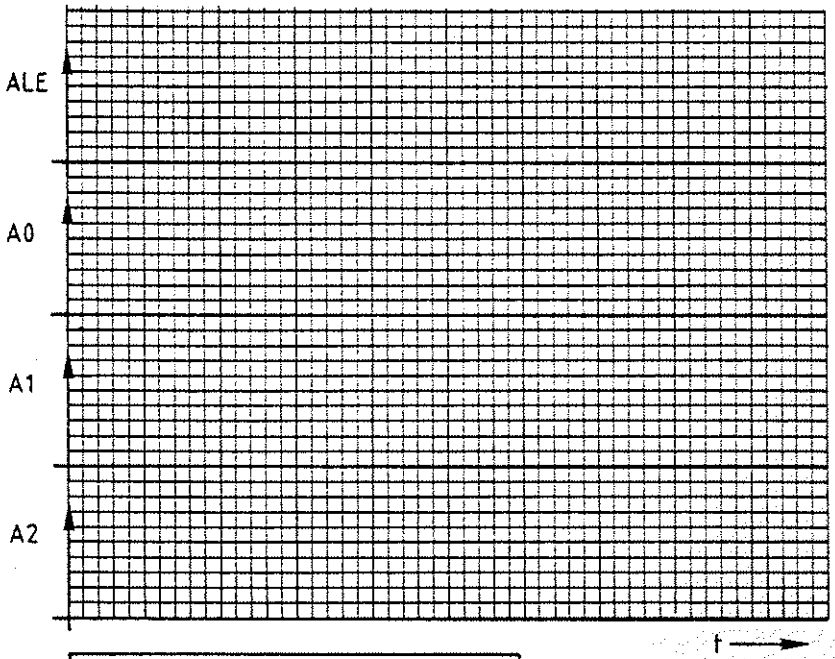
Wenn Ihr Oszilloskop keine "Ext. Trigger"-Einrichtung hat oder diese wegen der kurzen Triggerimpulse nicht richtig arbeitet (Bandbreite), legen Sie Signal A2 auf Kanal 2 und messen alle anderen Signale mit Kanal 1. Die Triggerung muß dann auf Kanal 2 geschaltet werden.



Mikroprozessor-Mikrocomputer Name:

Übungsteil 2 Datum:

A2.3



Y-Ablenkung:
X-Ablenkung:

Diskussion der Meßergebnisse:

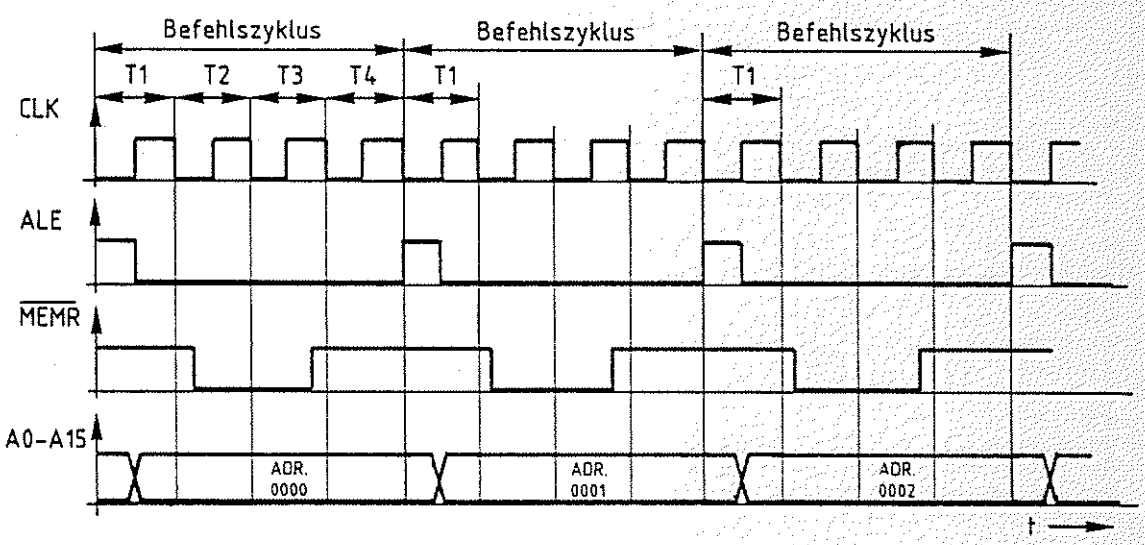


Bild A2.3: Zur Erklärung des "Free-Run-Modes"

A2.4

Bild A2.3 zeigt Ihnen noch einmal den zeitlichen Verlauf der wichtigsten Signale zum durchgeführten Versuch "Prozessorbaugruppe im "Free-Run-Mode".

Innerhalb der Taktzeit T1 gibt die CPU über die Ausgänge AD0 bis AD7 und A8 bis A15 eine Adresse aus, nehmen wir an, die Adresse 0000 (Hexadezimal). Mit der fallenden Taktflanke des ALE-Signals wird diese Adresse in den Adreßzwischenspeicher übernommen und auf den System-Bus geschaltet. Da sich der Prozessor zuerst einen Befehl holen muß, schaltet er zu Beginn der Taktzeit T2 das Steuersignal $\overline{\text{MEMR}}$ auf L-Pegel. Durch die Verbindung der Leitung $\overline{\text{MEMR}}$ mit der Datenleitung D7 findet der Prozessor auf dem Datenbus das Datenwort 7F vor, weil die Datenleitungen D0 bis D6 über die Bus-Abschlußwiderstände auf H-Pegel liegen, und die Datenleitung D7 durch das $\overline{\text{MEMR}}$ Signal auf L-Pegel gezogen wird.

Innerhalb der Taktzeit T3 wird das Befehlsbyte 7F in die CPU übernommen. Das $\overline{\text{MEMR}}$-Signal wird wieder auf H-Pegel geschaltet.

Während der Taktzeit T4 wird das gelesene Befehlsbyte in der CPU entschlüsselt und als 1-Byte-Befehl "MOV A,A" erkannt.

MOV A,A bedeutet für die CPU die Anweisung "Transportiere (Move = transportieren, übertragen) das Datenwort, das zur Zeit im Akkumulator steht, in den Akkumulator". Der Akkumulator ist ein 8-Bit-Register (Speicher) in der CPU. Der Befehl bewirkt keinerlei Veränderung der CPU.

Die "Bearbeitung" des Befehls ist also mit dem Ende von T4 abgeschlossen, es kann der nächste Befehl gelesen werden. Dazu gibt die CPU nun die Adresse 0001 aus, schaltet $\overline{\text{MEMR}}$ wieder auf L-Pegel und liest erneut das Befehlsbyte 7F.

Mit jedem weiteren Befehlszyklus - die CPU benötigt dazu jeweils vier Takte - wird die Befehlszähleinrichtung der CPU um 1 erhöht. Wenn der höchste Wert FFFF erreicht ist, beginnt die CPU wieder bei der Adresse 0000.

Oszilloskopiert man die Spannungen der Adreßleitungen, beginnend bei Leitung A0, so muß man Rechteckspannungen mit jeweils doppelter Perioden-dauer messen können. Die kleinste Periodendauer muß dem 8-fachen der Taktzeit des CLK-Signals (Pin 37, CPU) entsprechen. Die Ausführungszeit dieses Befehls läßt sich auch aus dem Abstand zweier ALE-Impulse bestimmen.



Mikroprozessor-Mikrocomputer

Name:

Übungsteil 2

Datum:

A3.1

Verfolgen der Bearbeitung eines Programms mit dem Oszilloskop

Mit Hilfe des Bus-Signalgebers wird das Programm in den RAM-Speicher eingegeben. Hierzu wird die CPU vom System-Bus getrennt. Nach der Programmeingabe wird die Bearbeitung des Programms durch die CPU bei normaler Arbeitsgeschwindigkeit mit einem Oszilloskop verfolgt.

Vorbereitungen:

- RAM-Baugruppe: Basisadresse mit den DIL-Schaltern auf 0000H stellen. RAM-Baustein in die Fassung IC8 stecken (sofern nicht vier RAMs vorhanden sind).

Folgende Baugruppen in den Baugruppenträger stecken:
Bus-Signalgeber
Bus-Signalanzeige über Adapterkarte
Prozessor 8085
RAM-Baugruppe

Betriebsspannung einschalten!



Mikroprozessor-Mikrocomputer Name:

Übungsteil 2 Datum:

A3.2

Anweisung	Anzeige; Kommentare
Signalgeber: ON/OFF ⟶ ON Signalanzeige: RUN/HLT ⟶ RUN ADDR. STOP ⟶ OFF	ADDRESS- und DATA-Anzeigen entsprechen den Einstellungen der ADDRESS- und DATA-Schalter. Die CPU ist vom Bus getrennt. In dieser Schalterstellung können Programme in den RAM-Speicher geladen werden. Vorbereitung für norm. Arbeitsgeschwindigkeit der CPU.
Laden Sie in den RAM-Speicher: ADDRESS / DATA 0000 / C3 0001 / 00 0002 / 00 Signalgeber: ON/OFF ⟶ OFF	Dies ist das Programm. Es besteht aus dem einzigen Befehl "JMP 0000". Die CPU arbeitet jetzt mit "Normalgeschwindigkeit".

Oszilloskopieren Sie mit einem Zweistrahl-Oszilloskop der Reihe nach die in folgender Tabelle angegebenen Signale.
Tragen Sie die Signalverläufe in das vorbereitete Diagramm ein.

Signal	Messbar ...	Bemerkungen
U_{MP1}	MP1-Signalanzeige	Eingang "Ext.Triggerung"
$\overline{CLK}$	2a-Adapter	Kanal 1, Systemtakt
ALE	MP3-Signalanzeige	Kanal 2
A0	16c-Adapter	Kanal 2
A1	17a-Adapter	Kanal 2
$\overline{MEMR}$	MP7-Signalanzeige	Kanal 2
D0	4a-Adapter	Kanal 2
D1	4c-Adapter	Kanal 2
D2	5a-Adapter	Kanal 2
D3	5c-Adapter	Kanal 2
D4	6a-Adapter	Kanal 2
D5	6c-Adapter	Kanal 2
D6	7a-Adapter	Kanal 2
D7	7c-Adapter	Kanal 2

Arbeitsblatt

BFZ / MFA 10.4. - 62

Mikroprozessor-Mikrocomputer

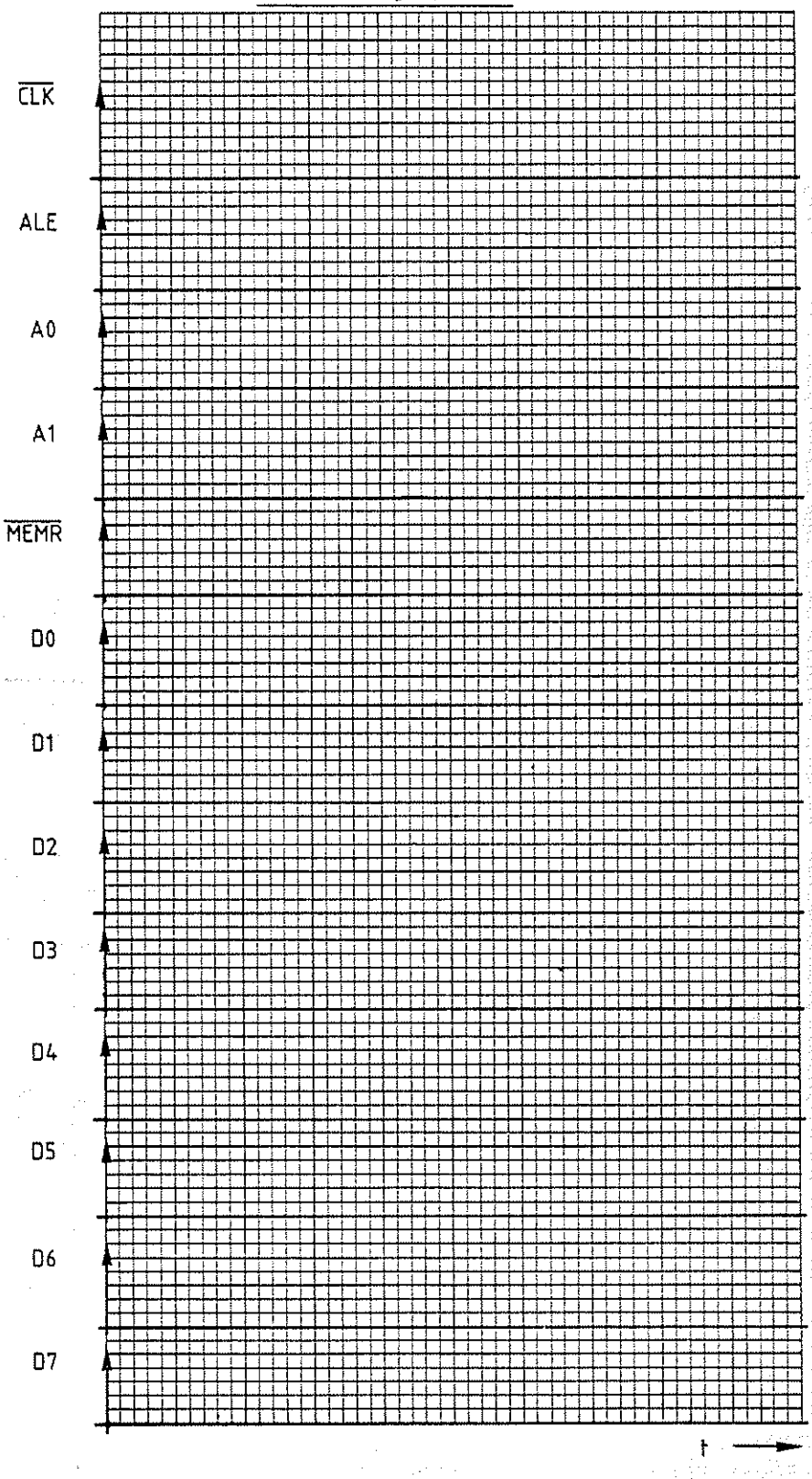
Name:

Übungsteil 2

Datum:

A3.3

Zeitablenkung: 1µs/Div.



A3.4

Diskussion der Meßergebnisse

Das Signal am Meßpunkt 1 der Signalanzeige wechselt immer dann auf H-Pegel, wenn der Adreßbus die Adresse 0000 führt. Diese Adresse haben Sie dem Adreßvergleicher auf der Bus-Signalanzeige mit den 16 DIL-Schaltern vorgegeben. Das Signal am Meßpunkt 1 ist das Ausgangssignal dieses Adreßvergleichers. Es wurde zur externen Triggerung des Oszilloskops verwendet. Dadurch haben alle aufgenommenen Signale den richtigen Zeitbezug zueinander.

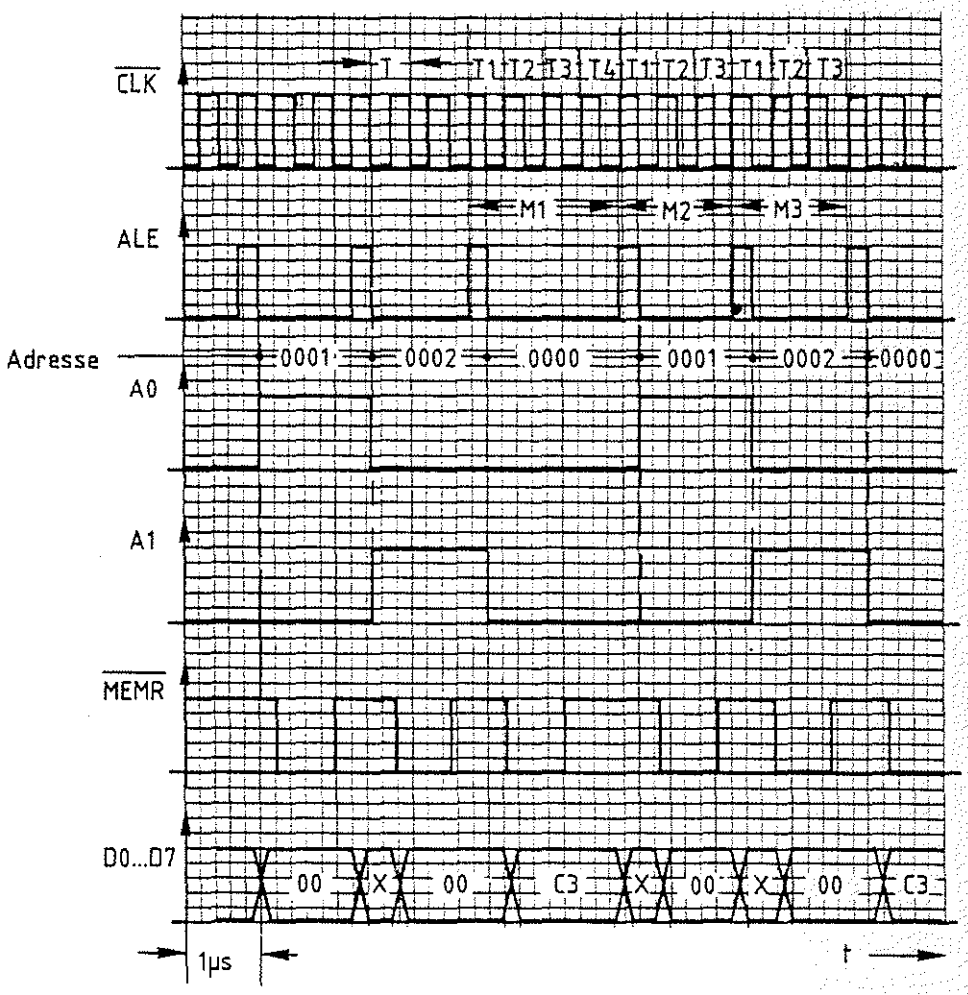


Bild A3.4: Oszillosgramme zum Arbeitschritt A3

Mikroprozessor-Mikrocomputer Name:

Übungsteil 2 Datum:

A3.5

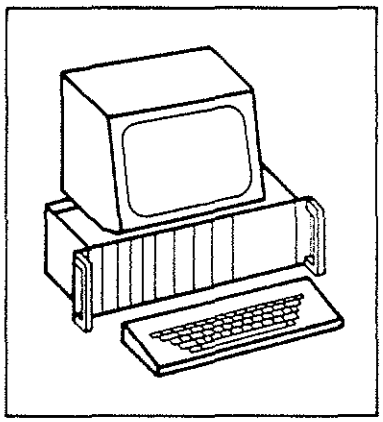
Das ALE-Signal nimmt jeweils zu Beginn eines neuen Maschinenzyklusses für eine halbe Taktperiode ($\overline{CLK}$) H-Pegel an. Sie können deutlich erkennen, daß der Maschinenzyklus M1 vier Taktperioden (T1-T4) andauert und die Zyklen M2 und M3 nur je drei. Ursache dafür ist die Befehlsentschlüsselung, die während des Taktzyklusses T4 erfolgt. Die Bearbeitungszeit für den gesamten Befehl (oder hier das Programm) beträgt 10 Taktzyklen bzw. 5 µs.

Aus dem Verlauf der Adreßsignale (A0 u. A1) lassen sich - jeweils für die Dauer eines Maschinenzyklusses - die vom Prozessor ausgegebenen Adressen bestimmen. Die ermittelten Adreßwerte sind in das Diagramm eingetragen.

Ein L-Pegel des Steuersignals MEMR zeigt an, daß die CPU aus dem Speicher liest. Um den Drei-Byte-Befehl zu lesen, sind drei Speicherzugriffe erforderlich. Das jeweils gelesene Datenwort läßt sich aus den Verläufen der einzelnen Datensignale ermitteln. Die so bestimmten Datenworte sind in die vereinfachte Darstellung der Datensignale eingetragen. Zwischen den Datenworten C3, 00 und 00 ergaben sich durch die Prozessoraktivität noch andere Signalzustände auf den Datenleitungen (mit X bezeichnet), die jedoch für den Programmablauf bedeutungslos sind.

Ende der Übung

FACHPRAKTISCHE ÜBUNG MIKROCOMPUTER-TECHNIK



MAT 85

BFZ/MFA 7.1.

Diese Übung ist Bestandteil eines Mediensystems, das im Rahmen eines vom Bundesminister für Bildung und Wissenschaft, vom Bundesminister für Forschung und Technologie sowie der Bundesanstalt für Arbeit geförderten Modellversuches zum Einsatz der "Mikrocomputer-Technik in der Facharbeiterausbildung" vom BFZ-Essen e.V. entwickelt wurde.

Inhaltsverzeichnis Seite

System-Informationen (Überblick)		
1.	Einleitung, Speicherbelegung	2
2.	Aufbau des Systems	3
3.	Arbeitsweise des Betriebsprogramms	4
3.1.	Kaltstart	4
3.2.	Kommando-Eingabe	5
3.3.	Reset-Betätigung, Warmstart	5
3.4.	Bildschirm-Modus	5
3.5.	Bedienerführung	6
4.	Struktur des Betriebsprogramms	6
4.1.	Kommando-Kurzbeschreibung	7
4.1.1.	Monitor-Kommandos	7
4.1.2.	Assembler/Disassembler-Kommandos	8
4.1.3.	Tracer-Kommandos	8
4.1.4.	Ordnung der Kommandos nach Einsatzgebieten	9

Gebrauch der Kommandos (Ausführl. Kommando-Beschreibung)		
—	Inhaltsverzeichnis	10
—	Hinweise zur Beschreibung der Kommandos	11
—	Beschreibung der Kommandos und Übungen dazu	14

Anhang (Techn. Daten, Arbeitsblätter, Progr. Beispiele)		
1.	Anschluß einer Datensichtstation	62
1.1.	Bedingungen für eine fehlerfreie Datenübertragung	62
1.2.	Anschlußplan	63
2.	Druckermodus, TTY-Betrieb	64
3.	Anschluß eines Matrix-Druckers	64
3.1.	Betrieb des Matrix-Druckers	65
4.	ASCII-Code-Tabelle	67
5.	Die Tastatur Cherry G 80-0177	69
6.	Häufig verwendete Symbole für Flußdiagramme	70
7.	Unterprogramme des Betriebsprogramms	71
8.	Beispiele für den Gebrauch von Unterprogrammen aus dem Betriebsprogramm	74
8.1.	Verzögerungszeit n x 0,24s	74
8.2.	Bildschirm löschen und Textausgabe	75
8.3.	Steuerung des Cursors per Programm	76
—	8085-Befehlsliste	80

System-Informationen

1. Einleitung, Speicherbelegung

Das Betriebsprogramm MAT 85*) gestattet mit Hilfe von 14 Kommandos das Ein- und Ausgeben, das Testen und das Verfolgen der Wirkungsweise von Anwenderprogrammen.
Das Betriebsprogramm ist in vier 2-KByte-EPROM's vom Typ 2716 gespeichert und belegt den Adreßraum ab Adresse 0000 bis 1FFFH. An Schreib-Lesespeicher benötigt es 1 KByte, so daß dem Anwender bei einer Bestückung der RAM-Karte mit einem 2-KByte-RAM-Baustein ein Speicherbereich von 1 KByte zur Verfügung steht. Der Schreib-Lesespeicher muß am Ende des adressierbaren Speicherbereiches liegen. Die erforderliche Speicherbelegung ist in Bild 1 dargestellt.

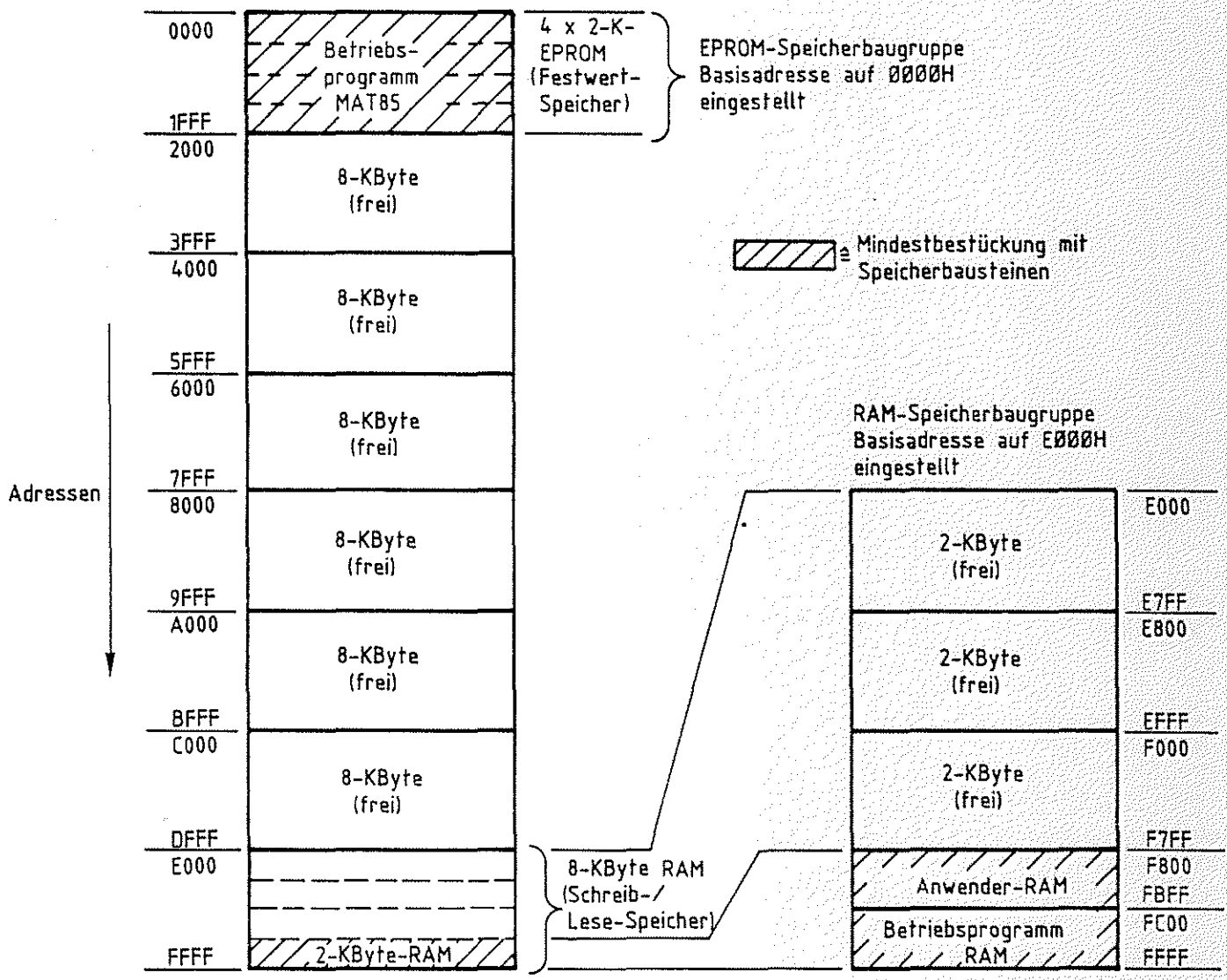


Bild 1: Speicher-Belegung

*) MAT 85 = Abkürzung für Monitor-Assembler-Tracer für Prozessor-Baugruppe 8085.

2. Aufbau des Systems

Für den Aufbau des Systems benötigen Sie die folgenden Baugruppen:

- Baugruppenträger mit Busverdrahtung BFZ/MFA 0.1.
- Busabschluß BFZ/MFA 0.2.
- Trafo-Einschub BFZ/MFA 1.1.
- Spannungsregelung BFZ/MFA 1.2.
- Prozessor 8085 BFZ/MFA 2.1.
- 8-K-RAM/EPROM BFZ/MFA 3.1. bestückt mit MAT 85
- 8-K-RAM/EPROM BFZ/MFA 3.1. bestückt mit mind. 2-K-RAM
- Video-Interface BFZ/MFA 8.2.
- ASCII-Tastatur BFZ/MFA 8.1. } Datensichtstation
- Monitor mit Cinch-Anschluß

In Bild 2 ist der Aufbau des Mikrocomputers aus diesen Baugruppen dargestellt.

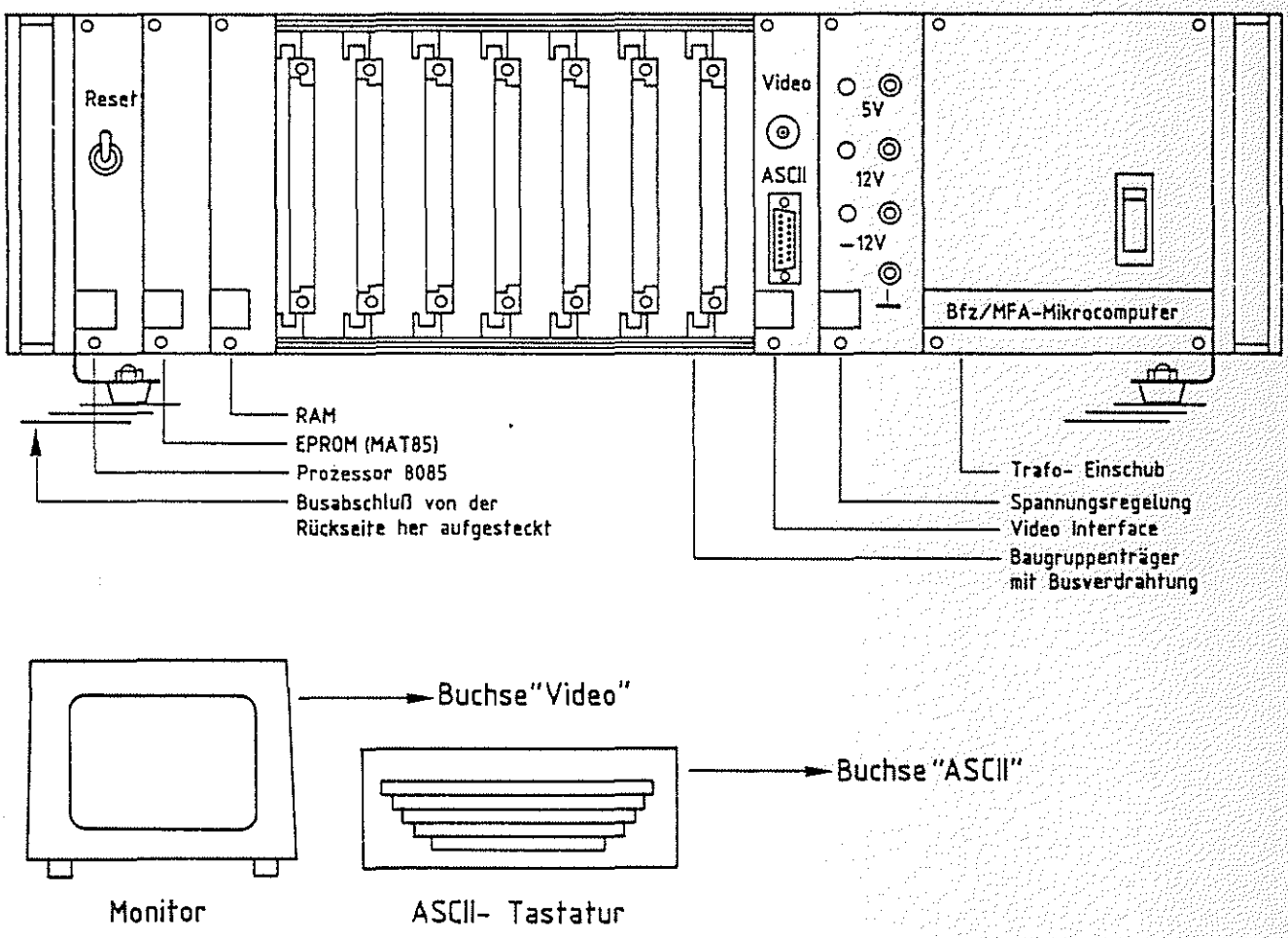


Bild 2: Aufbau des Mikrocomputers

Soll anstelle der dargestellten Datensichtstation eine andere oder ein Fernschreiber (Teletype, TTY) verwendet werden, so müssen zunächst die Anschlüsse für diese Geräte vorbereitet werden. Hinweise hierzu finden Sie im Anhang.

3. Arbeitsweise des Betriebsprogramms

3.1. Kaltstart

Mit dem Einschalten der Betriebsspannung (Kaltstart) wird das Betriebsprogramm gestartet und die Übertragungsgeschwindigkeit (Baud-Rate) des angeschlossenen Dialoggerätes (Datensichtstation bzw. TTY) erfaßt, um die eigene Obertragungsgeschwindigkeit an die des Dialoggerätes anzupassen. Dazu ist es erforderlich, daß ein bestimmtes Zeichen vom Dialoggerät an den Mikrocomputer gesendet wird.

Daher muß die SPACE-Taste kurz betätigt werden, worauf sich das Betriebsprogramm mit der Versionsnummer und dem Ausdruck aller zur Verfügung stehenden Bediener-Kommandos meldet (Bild 3). Die Überschrift mit der Versionsnummer wird auf dem Bildschirm nur kurzzeitig angezeigt.

```
ASSEMBLER
BREAKPOINT
DISASSEMBLER
GO
HELP
IN
LOAD TAPE
MEMORY
NEXT INSTRUCTION
OUT
PRINT
REGISTER
SAVE
TRACE INTERVAL

KMD >_
```

Bild 3: Ausdruck der verfügbaren Monitor-Kommandos nach einem Kaltstart

3.2. Kommando-Eingabe

Die Bereitschaft zur Annahme eines Kommandos vom Bediener wird durch den Ausdruck "KMD>_" angezeigt (Kommando-Modus). Jedes der aufgelisteten Kommandos kann durch Eingabe seines ersten Buchstabens und durch anschließendes Betätigen der Taste "RETURN" bzw. "CR" (Wagenrücklauf) oder "SPACE" (Leertaste) aufgerufen werden. Daraufhin druckt das Betriebsprogramm den vollständigen Kommandonamen aus und fordert eventuell zusätzlich erforderliche Informationen an. Soll das Kommando abgebrochen werden, so muß die Taste "ESC" (Escape = flüchten) betätigt werden. Das Betriebsprogramm quittiert diese Eingabe durch ein akustisches Signal und fordert durch den Ausdruck "KMD>_" ein neues Kommando an.

3.3. Reset-Betätigung, Warmstart

Im Gegensatz zum Kaltstart erfolgt nach Betätigung der RESET-Taste (Warmstart oder warmer RESET) keine Erfassung der Übertragungsgeschwindigkeit und auch kein Auflisten der Bediener-Kommandos, sondern die Ausgabe

```
*** RESET ***
```

und die Aufforderung zur Kommando-Eingabe "KMD>_".

3.4. Bildschirm-Modus

Das Betriebsprogramm unterscheidet je nach gemessener Übertragungsgeschwindigkeit zwischen einem Bildschirm- und einem Drucker-Modus (siehe Anhang Kapitel 2.)

Im Bildschirm-Modus können falsch eingegebene Zeichen (Kommandos, usw.) durch Betätigung der Taste "DEL" (Delete = streichen) oder "RUBOUT" (ausradieren) gelöscht werden.
Bei längeren Protokollen (z.B. beim PRINT-Kommando) wird nach jeder Bildschirmseite (16 Zeilen, zu je maximal 64 Zeichen) der Ausdruck gestoppt und der Text "==> SPACE" ausgegeben. Der Bediener erhält damit die Möglichkeit, die Protokollierung auch bei hohen Übertragungsgeschwindigkeiten zu verfolgen. Der Ausdruck wird fortgesetzt, wenn die SPACE-Taste kurz betätigt wird.

3.5. Bedienerführung

Unabhängig vom Bildschirm- bzw. Drucker-Modus wird der System-Bediener vom Betriebsprogramm geführt, indem es eventuell zusätzliche Informationen für die Kommando-Ausführung (z.B. Adressen usw.) anfordert. Dabei erfolgt sofort eine Kontrolle, ob die Eingabedaten dem notwendigen Format entsprechen (SYNTAX-Prüfung). Ist dies nicht der Fall, wird der Bediener durch ein akustisches Signal auf seinen Fehler aufmerksam gemacht. Solch ein Signal ertönt z.B. dann, wenn das Betriebsprogramm eine Adresse angefordert hat und das eingegebene Zeichen kein Hex-Zeichen ist.
Im Bildschirm-Modus wird das falsch eingegebene Zeichen angezeigt, indem der CURSOR (Schreibstellen-Zeiger, Schreibmarke auf dem Bildschirm) auf dieses Zeichen zeigt. Im Drucker-Modus werden falsche Zeichen vom Betriebsprogramm ignoriert.

4. Struktur des Betriebsprogramms

Das Betriebsprogramm MAT 85 ist in drei Programmblöcke unterteilt. Jedem dieser Blöcke ist eine bestimmte Aufgabe und ein Teil der Kommandos zugeordnet. Bild 4 zeigt diese Struktur.

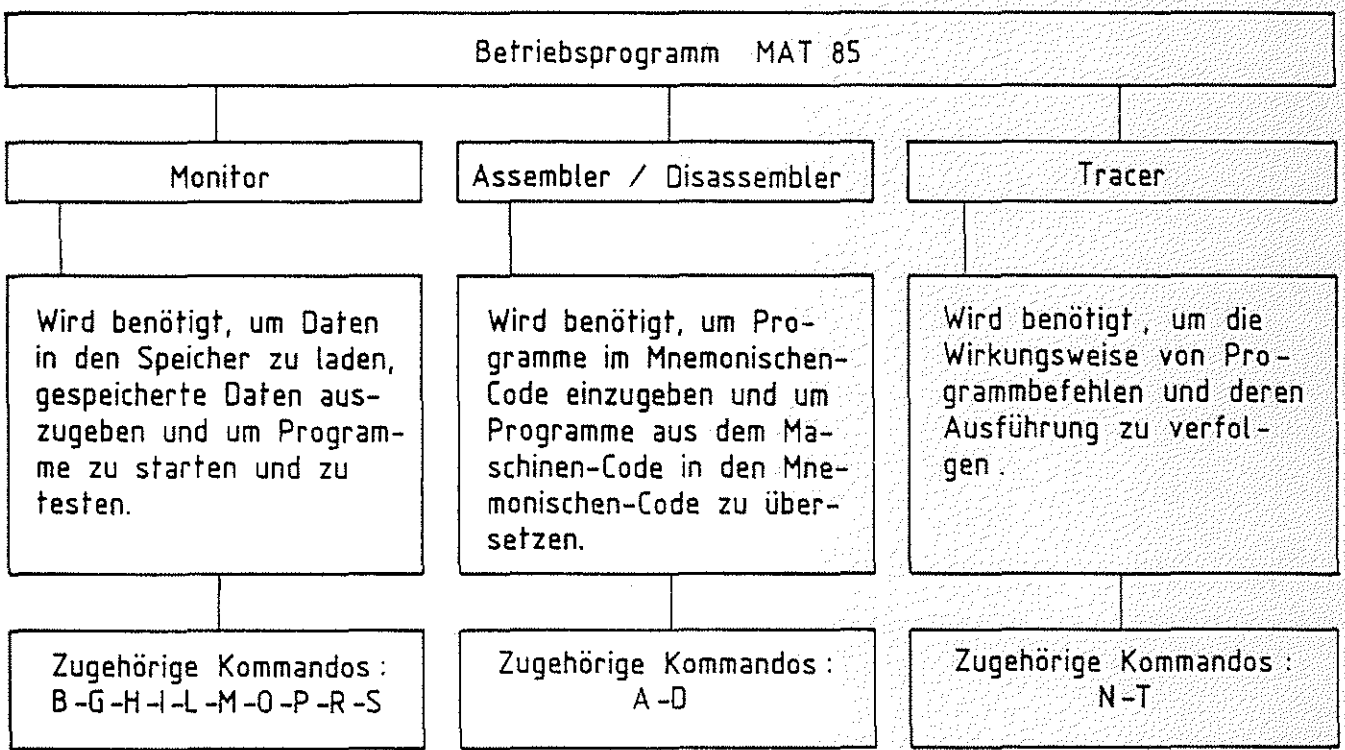


Bild 4: Struktur des Betriebsprogramms MAT 85

4.1. Kommando-Kurzbeschreibung

4.1.1. Monitor-Kommandos

BREAKPOINT: Dieses Kommando ermöglicht es, mit dem GO-Kommando Unterbrechungspunkte einzugeben. Unterbrechungspunkte (engl. Breakpoints) sind Adressen aus dem Speicherbereich des Anwenderprogramms, an denen die Programmabarbeitung unterbrochen werden soll.
Nach der Unterbrechung werden die Inhalte der CPU-Register angezeigt.

GO: Mit diesem Kommando können eingegebene Programme gestartet werden.

HELP: Dient dazu, alle verfügbaren Kommandos des Betriebsprogramms anzuzeigen.

IN: Dieses Kommando dient dazu, Daten von Eingabe-Ports zu lesen und anzuzeigen.

LOAD TAPE: Lädt Daten von einer Magnetband-Kassette in den Speicher des Mikrocomputers. Hierzu wird das Kassetten-Interface BFZ/MFA 4.4.a benötigt.

MEMORY: Mit diesem Kommando lassen sich die Inhalte von Speicherzeilen in verschiedenen Formaten ausdrucken und ändern.

OUT: Dient dazu, Daten an Ausgabe-Ports zu senden.

PRINT: Mit diesem Kommando können die Inhalte von Speicherzeilen in verschiedenen Formaten (Binär, Hexadezimal, Dezimal, ASCII) formatiert (pro Zeile max. 8 Inhalte) ausgedruckt werden.

REGISTER: Mit diesem Kommando können die Anfangswerte der CPU-Register, z.B. vor einem Testlauf des Anwenderprogramms, vorgegeben werden.

SAVE: Dient dazu, Daten auf einem Kassetten-Recorder zu speichern. Hierzu wird das Kassetten-Interface BFZ/MFA 4.4.a benötigt.

4.1.2. Assembler/Disassembler-Kommandos

ASSEMBLER_____: Mit diesem Kommando wird ein Programm aufgerufen, das es ermöglicht, Anwendungsprogramme im Mnemo-Code (8085-Intel-Format) einzugeben. Der eingegebene Code wird Zeile für Zeile in den zugehörigen Maschinen-Code übersetzt und im RAM-Speicher abgelegt.

DISASSEMBLER____: Mit diesem Kommando können Programme, die im Maschinen-Code gespeichert sind, in den Assembler-Code übersetzt werden.

4.1.3. Tracer-Kommandos

NEXT INSTRUCTION_: Mit diesem Kommando wird ein Tracer (Verfolger) aktiviert, der es ermöglicht, die Ausführung und Wirkungsweise einer vorgegebenen Anzahl von Programmbefehlen zu verfolgen. Dazu wird nach jedem Befehl (engl. Instruction) die Programmbearbeitung kurz unterbrochen und die Inhalte aller CPU-Register werden protokolliert.

TRACE INTERVAL___: Dieses Kommando bewirkt eine Protokollierung der Registerinhalte immer dann, wenn diejenigen Programmbefehle abgearbeitet werden, die in einem vorher zu bestimmenden Speicherbereich liegen.

4.1.4. Ordnung der Kommandos nach Einsatzgebieten

Die in Bild 5 dargestellte Grafik zeigt die beschriebenen Kommandos nach Einsatzgebieten geordnet.

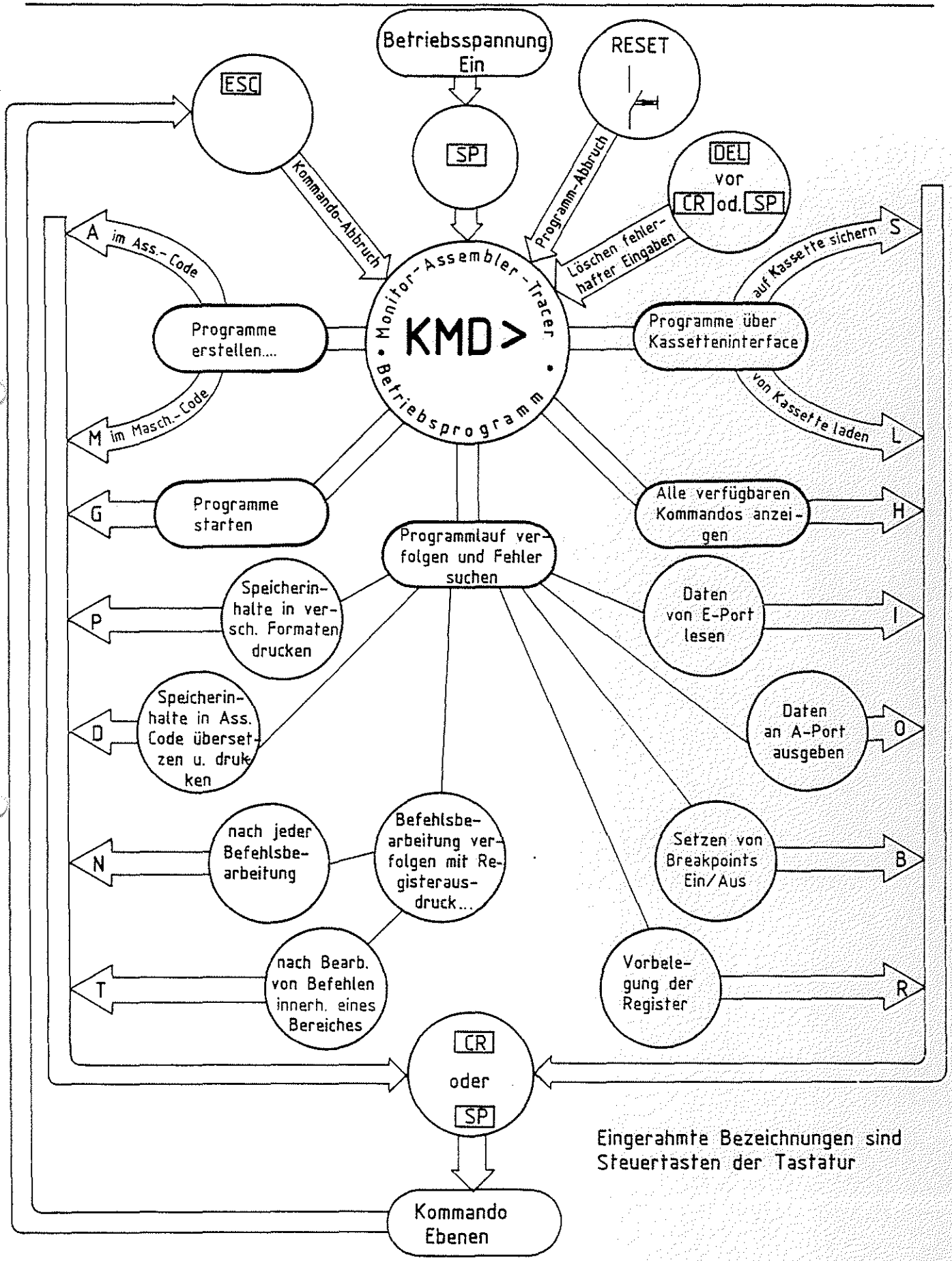


Bild 5: Zuordnung der Kommandos zu Einsatzgebieten

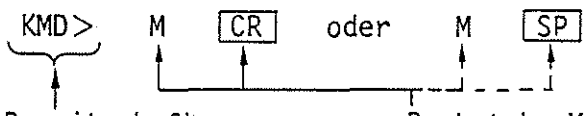
Inhaltsverzeichnis

	Seite
Hinweise zur Beschreibung der Kommandos	11
HELP	14
MEMORY	15
PRINT	18
GO	20
DISASSEMBLER	23
NEXT INSTRUCTION	25
ASSEMBLER	27
REGISTER	44
BREAKPOINT	46
TRACE INTERVAL	53
IN	58
OUT	59
SAVE	60
LOAD	61

— Hinweise zur Beschreibung der Kommandos

Unter Kapitel 3.2 der "System-Informationen" wurde kurz beschrieben, wie der Mikrocomputer seine Bereitschaft zur Annahme eines Kommandos anzeigt, wie ein Kommando aufgerufen wird und wie man ein Kommando abbricht.
Im folgenden werden Aufruf und Verwendung der einzelnen Kommandos ausführlich beschrieben. Anhand von Bildschirmausdrucken und Kommentaren kann die Anwendung eines jeden Kommandos nachvollzogen werden. Übungsaufgaben dienen dazu, das Erlernte zu vertiefen.
Um Tastatureingaben, Bildschirmausdrucke und die Kommentare dazu übersichtlich und allgemeingültig zu gestalten, werden einige Abkürzungen und Darstellungsweisen verwendet, die am Beispiel des MEMORY-Kommandos zunächst erklärt werden sollen:

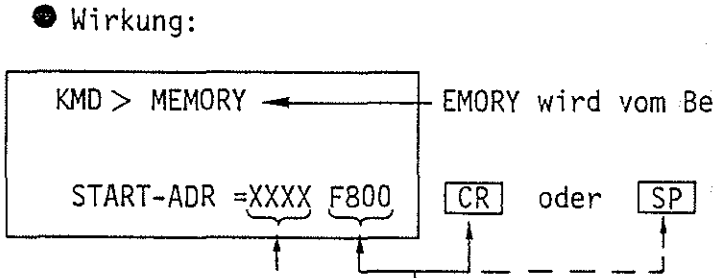
● Aufruf des MEMORY-Kommandos (wenn KMD >_ angezeigt wird):



Bereitschafts-meldung

Buchstabe M gefolgt von der Wagenrücklauftaste oder der Leertaste eintippen.
Eingerahmte Zeichen sind Tasten mit einer Steuerfunktion!

● Wirkung:

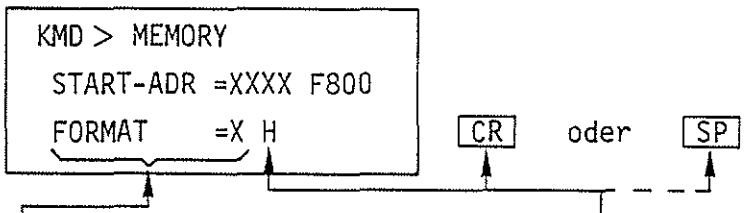


Hier steht die Vorgabe-Adresse des Computers (Vorschlag). Wird ihr Wert akzeptiert, müssen Sie die CR - oder SP - Taste betätigen, wenn nicht,

müssen Sie die hexadezimale Adresse derjenigen Speicherzeile eintippen, die als erste bearbeitet werden soll. Hier wird diese Adresse zu F800 gewählt, indem hintereinander die Zeichen F800 eingetippt werden, gefolgt von der Betätigung der CR - oder SP - Taste. Allgemeingültig wird diese Adresse "Neu-Adresse" genannt und durch "YYYY" gekennzeichnet.

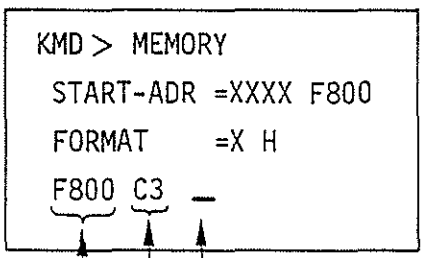
Gebrauch der Kommandos

● Wirkung:



Das Betriebsprogramm schlägt vor, die Daten in dem bei X angegebenen Format darzustellen bzw. zu bearbeiten. Wird die Vorgabe akzeptiert, müssen Sie CR oder SP betätigen. Wenn nicht, müssen Sie einen der Buchstaben A, B, D oder H gefolgt von CR oder SP eintippen. Hier wird H für Hexadezimale Darstellung der Daten eingegeben.
Allgemeingültig wird das Format "Neu-Format" genannt und durch "Y" gekennzeichnet. Die Bedeutung der verschiedenen Formate wird bei der Beschreibung des MEMORY-Kommandos gezeigt.

● Wirkung:



Die gewünschte Adresse wird angezeigt.

Der Cursor (Schreibmarke) erwartet weitere Eingaben, die unter der Beschreibung des MEMORY-Kommandos gezeigt werden.

Dies ist der Inhalt der Speicherzeile mit der angegebenen Adresse im gewünschten Format. Der hier ausgegebene Wert C3 ist vom Zufall abhängig, bei Ihrem Computer kann ein anderer Wert angezeigt werden.

In Bild 6 sind die oben beschriebenen Arbeitsschritte in gekürzter Form dargestellt. Diese Art der Darstellung wird bei der Beschreibung der Kommandos verwendet.

Schirmbild (oft Ausschnitt)	Tastatureingaben und Kommentare zum Schirmbild
KMD > MEMORY START-ADR =XXXX YYYY FORMAT =X Y	M [CR] oder M [SP] eintippen "EMORY" wird ergänzt XXXX = Vorgabe; Neu: YYYY [CR] oder [SP] Vorgabe: [CR] oder [SP] X = Vorgabe; Neu: Y[CR] oder [SP] Vorgabe: [CR] oder [SP] Y = A: ASCII (Druckbare Zeichen) B: Binär (0,1) D: Dezimal (0...9) H: Hexadezimal (0...F)

Bild 6: Kurzform der Darstellung des Schirmbildes, von Tastatureingaben und Kommentaren zum Schirmbild

- Alle weiteren vom Betriebsprogramm vorgegebenen, oder vom Benutzer zu verändernden Werte sind sinngemäß zu handhaben.

- Fehlerhafte Eingaben können vor Kommando-Abschluß durch die [CR]- oder [SP]-Taste mit der [DEL]-Taste (Delete = löschen) gelöscht und dann entsprechend korrigiert werden.

- Die Rückkehr aus den Kommandoebenen in das Betriebsprogramm erfolgt durch Betätigen der [ESC]-Taste (Escape = flüchten). Siehe hierzu auch Bild 5.

H-Kommando

H

Mit dem Help-Kommando lassen sich die Namen aller zulässigen Kommandos des Betriebssystems MAT 85 in alphabetischer Reihenfolge ausdrucken.

Aufruf und Handhabung:

KMD> HELP	H[CR] oder H[SP] eintippen "ELP" wird ergänzt
(Kommando-Ausführung)	
KMD>_	Nächstes Kommando

Zur Kommando-Ausführung:

- Nach dem Ausdruck aller Kommandonamen (die obere Zeile "KMD> HELP" wird überschrieben) erfolgt ein Rücksprung in die Kommando-Routine (KMD>_).

- Zum Aufruf eines der Kommandos muß nur der 1. Buchstabe, gefolgt von der Taste [CR] (Carriage returm = Wagen-Rücklauf) oder der Taste [SP] (Space = Leerzeichen) eingegeben werden.
  Andernfalls erfolgt eine Fehlermeldung ohne Annahme der Eingabe.

- Eingaben, die vor Betätigung der [CR]- oder [SP]-Taste erfolgen, können mit der Taste [DEL] (Delete = streichen) gelöscht werden.

M-Kommando **M1**

Mit dem Memory-Kommando lassen sich die Speicherinhalte in verschiedenen Formaten byte-weise anzeigen und ändern.

Aufruf und Handhabung:

```
KMD > MEMORY

 START-ADR = XXXX YYYY

 FORMAT    = X Y
```

M [CR] oder M [SP] eintippen
"EMORY" wird ergänzt

XXXX = Vorgabe; Neu: YYYY [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X = Vorgabe; Neu: Y [CR] oder [SP]
Vorgabe: [CR] oder [SP]

Y = A: ASCII (Druckbare Zeichen)
= B: Binär (0,1)
= D: Dezimal (0...9)
= H: Hexadezimal (0...F)

— Beispiel für Adresse = F800 und Format = H:

Schirmbild	Eingabe	Wirkung
F800 C3 ←zufällige Inhalte	[SP]	Inh. unverändert, ADR + 1
F801 20 22	22 [SP]	Inh. verändert, ADR + 1
F802 44	[SP]	Inh. unverändert, ADR + 1
F803 55 -	[-]	Inh. unverändert, ADR - 1
F802 44 54-	54 [-]	Inh. verändert, ADR - 1
F801 22 } eingegebene Werte	[+]	Inh. unverändert, ADR + 1
F802 54 }	[CR]	Inh. unverändert, Fertig
KMD >_		Nächstes Kommando kann eingegeben werden.

— Beispiel für Adresse = F850 und Format = A:

Schirmbild	Eingabe	Wirkung
F850 A4		Wenn Speicherinhalt kein ASCII-Zeichen, wird Hex-Code angezeigt.
	[SP]	Inh. unverändert, ADR + 1
F851 B7 R	R [SP]	Inh. verändert, ADR + 1
F852 A6 A	A [SP]	Inh. verändert, ADR + 1
F853 BA M-	M [-]	Inh. verändert, ADR - 1
F852 .A U		ASCII-Zeichen sind durch "." gekennz.
	U [SP]	Inh. verändert, ADR + 1
F853 .M	[CR]	Inh. unverändert, Fertig
KMD >_		Nächstes Kommando

— Beispiel für Adresse = F860 und Format = D:

Schirmbild	Eingabe	Wirkung
F860 164	SP	Inh. unverändert, ADR + 1
F861 183 1	1 SP	Inh. verändert, ADR + 1
F862 247 0	0 SP	Inh. verändert, ADR + 1
F863 191 300	300 SP	Summer ertönt, da 300 > 255; 30 wird angenommen. Inh. verändert, ADR + 1
F864 160 -	-	Inh. unverändert, ADR - 1
F863 30 -	-	Inh. unverändert, ADR - 1
F862 0 -	-	Inh. unverändert, ADR - 1
F861 1	CR	Inh. unverändert, Fertig
KMD >_	Nächstes Kommando	

— Beispiel für Adresse = F870 und Format = B:

Schirmbild	Eingabe	Wirkung
F870 10100100	SP	Inh. unverändert, ADR + 1
F871 10110111 00001111	00001111 SP	Inh. verändert, ADR + 1
F872 10100000 -	-	Inh. unverändert, ADR - 1
F871 00001111	CR	Inh. unverändert, Fertig
KMD >_	Nächstes Kommando	

Verlassen des Kommandos Memory:

1. Durch Betätigung von CR
   Das Betriebssystem trägt die letzte Änderung in den RAM-Speicher ein und fordert zur Eingabe eines neuen Kommandos auf

2. Durch Betätigung von ECS
   Die Bearbeitung des Memory-Kommandos wird abgebrochen.
   Achtung! Eine gewünschte Änderung des Speicherinhaltes an der zuletzt angezeigten Adresse wird nicht ausgeführt.

Arbeitsblatt | BFZ / MFA 7.1. - 17

MAT 85 | Name:

M-Kommando | Datum:

M3

Geben Sie mit dem M-Kommando ab Adresse F800 folgende Daten im Hex-Format ein:

33	48	20	12	FF	3F	0D	4D	6D	7C

Sehen Sie sich diese Daten in den vier verschiedenen Formaten an und tragen Sie die Ergebnisse in die Tabelle ein.
Oben Sie dabei auch den Gebrauch der Tasten [DEL] , [+] , [-] und [ESC]

Zeile Nr.	Adr.	Inhalt der Speicherzeilen im Format ... H	D	A	B
1	F800				
2	F801				
3	F802				
4	F803				
5	F804				
6	F805				
7	F806				
8	F807				
9	F808				
10	F809				

Besonderheiten im Format A (ASCII-Zeichen):

Zeile	Bemerkungen
3	Die Hex-Zahl 20 entspricht dem ASCII-Zeichen SP (Space, Leerzeichen). Daher ist hier nur der . zu erkennen, der anzeigt, daß die Speicherzeile ein ASCII-Zeichen enthält.
4,5,7	Diese Speicherzeilen enthalten hexadezimale Zahlen, deren Bitkombinationen keine ASCII-Zeichen entsprechen. Im A-Format wird daher der Inhalt solcher Speicherzeilen als Hex-Zahl dargestellt.
8,9	Obwohl in diesen beiden Speicherzeilen Inhalte stehen, die sich durch ein Bit voneinander unterscheiden, wird für beide Inhalte das ASCII-Zeichen M angezeigt. Tatsächlich aber entspricht der Hex-Zahl 4D das M und 6D das m. Das Video-Interface des Computers zeigt jedoch nur Großbuchstaben an.

P1

P-Kommando

Mit dem Print-Kommando werden die Inhalte eines Speicherbereichs im gewünschten Format ausgedruckt. Dazu muß der anzuzeigende Speicherbereich durch Angabe einer Start- und Stop-Adresse definiert werden, die das Betriebsprogramm nach dem Kommandoaufruf vom Bediener erfragt. Die möglichen Formate entsprechen denen des M-Kommandos.

Im Protokoll werden je nach dem gewählten Format bis zu acht Speicherinhalte in einer Zeile ausgedruckt. Jedes Zeilenprotokoll beginnt mit der Adresse des ersten in der Zeile ausgedruckten Speicherinhalts.

Anwendung: Dokumentation von Programmen im Hex-Code,
Text im Speicher suchen.

Aufruf und Handhabung:

```
KMD > PRINT

 START-ADR =X1X1 Y1Y1

 STOP -ADR =X2X2 Y2Y2

 FORMAT    =X Y
```

P [CR] oder P [SP] eintippen
"RINT" wird ergänzt

X1X1 = Vorgabe; Neu: Y1Y1 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X2X2 = Vorgabe; Neu: Y2Y2 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X = Vorgabe; Neu: Y [CR] oder [SP]
Vorgabe: [CR] oder [SP]
Y = A, B, D, H wie beim M-Kommando.

— Beispiel Startadresse = 0080, Stopadresse = 0094, Format = H:

```
KMD > PRINT
 START-ADR =0000 0080
 STOP -ADR =0000 0094
 FORMAT    =H
 0080 43 29 20 43 4F 50 59 52
 0088 49 47 48 54 20 31 39 38
 0090 32 20 42 46 5A
KMD >_
```

Format-Vorgabe wurde akzeptiert.
Speicherzeile mit Adresse 0080 hat den Inhalt 43, Speicherzeile mit Adresse 0081 den Inhalt 29 usw.

— Beispiel für gleiche Start- und Stopadresse, jedoch Format D:

```
KMD > PRINT
 START-ADR =0080
 STOP -ADR =0094
 FORMAT    =H D
 0080  67  41  32  67  79  80  89  82
 0088  73  71  72  84  32  49  57  56
 0090  50  32  66  70  90
KMD >_
```

Die in obigem Beispiel dargestellten Speicherinhalte sind hier in dezimaler Schreibweise ausgedruckt.

Beispiel für gleiche Start- und Stopadresse, jedoch Format = A:

```
KMD > PRINT
 START-ADR =0080
 STOP -ADR =0094
 FORMAT    =D A
 0080 .C .) .  .C .0 .P .Y .R
 0088 .I .G .H .T .  .1 .9 .8
 0090 .2 .  .B .F .Z

KMD > _
```

Die im ersten Beispiel dargestellten Speicherinhalte sind hier im ASCII-Code dargestellt.

- Beispiel für gleiche Start- und Stopadresse, jedoch Format = B:

```
KMD > PRINT
 START-ADR =0080
 STOP -ADR =0094
 FORMAT    =A B
 0080 01000011
 0081 00101001
 0082 00100000
 0083 01000011
 0084 01001111
 0085 01010000
 0086 01011001
 0087 01010010
 0088 01001001
 0089 01000111
 008A 01001000
 008B 01010100
 008C 00100000
 008D 00110001
 008E 00111001
 008F 00111000
 0090 00110010
 0091 00100000
 0092 01000010
 0093 01000110
 0094 01011010

KMD > _
```

Die im ersten Beispiel dargestellten Speicherinhalte sind hier im Binär-Code dargestellt.

Der Bildschirmausdruck wird an dieser Stelle unterbrochen durch Ausgabe der Aufforderung, die SPACE-Taste (_==>SPACE) zu betätigen. Die oberen drei Zeilen werden aus dem Bild "gerollt".
Diese Meldung wird immer dann ausgegeben, wenn mehr als 16 Bildschirmzeilen ausgegeben werden sollen.

Übung: Drucken Sie die Inhalte des Speicherbereichs von 0190 bis 0250 in allen Formaten aus.

G-Kommando G1

Mit dem Go-Kommando wird der Prozessor veranlaßt, Anwender-Programme von einer bestimmten Startadresse an abzuarbeiten.

Aufruf und Handhabung:

```
KMD > GO

 START-ADR =XXXX YYYY

(Kommando-Ausführung)
 *** USER ***
KMD > _
       (oder)
 _
```

G [CR] oder [SP] eintippen
"0" wird ergänzt

XXXX = Vorgabe; Neu: YYYY [CR] oder [SP]
Vorgabe: [CR] oder [SP]

Meldung nach Abarbeitung eines nicht zyklischen Programms, das mit einem Restart-Befehl abgeschlossen wurde.
Meldung bei Abarbeitung eines zyklischen Programms.

Zur Kommando-Ausführung:

- Bei der Ausführung zyklischer Programme (Schleifen ohne Ende) kann eine Rückkehr zum Betriebsprogramm nur durch Betätigen der RESET-Taste erfolgen. Danach meldet sich das Betriebsprogramm mit dem Ausdruck *** RESET *** (Rücksetzen) und erwartet das nächste Kommando.

- Nicht zyklische Programme müssen mit einem Rücksprungbefehl (RST1, Restart, CFH) abgeschlossen sein. Wenn dieser Befehl ausgeführt wurde, meldet sich das Betriebsprogramm mit dem Ausdruck *** USER *** (Benutzer) und erwartet das nächste Kommando.

- Durch Fehlbedienungen des Gerätes während der ersten Experimente mit dem Betriebssystem kann es vorkommen, daß sich ein Programm nicht starten läßt, obwohl es richtig eingegeben wurde. Meist erfolgt nach dem Startversuch die Meldung *** RESET ***. Um das Programm starten zu können, muß der Stack-Pointer (Stapelzeiger), ein spezielles Register in der CPU, mit Hilfe des Register-Kommandos mit dem Wert FC32H geladen werden.
  Hinweise hierzu finden Sie unter der Beschreibung des Register-Kommandos.

MAT 85

Name:

G-Kommando

Datum:

G2

Für die folgenden Experimente benötigen Sie zusätzlich eine Eingabe- und eine Ausgabe-Baugruppe im Baugruppenträger. Stellen Sie vor dem Einschieben die Port-Adressen wie folgt ein:
Eingabe-Baugruppe: Adresse 12H
Ausgabe-Baugruppe: Adresse 13H

1. Laden Sie mit dem Memory-Kommando ab Adresse F800H das folgende zyklische Programm in den Speicher.

Adressen	Daten	Bemerkungen zu den Befehlen
F800 F801	DB 12	Eingabe-Port mit Port-Adr. 12 lesen und gelesenen Wert in den Akkumulator transportieren
F802 F803	D3 13	Akku-Inhalt zum Ausgabe-Port mit Port-Adr. 13 transportieren
F804 F805 F806	C3 00 F8	Sprung zum Anfang dieses Programms (bei Adr. F800)

2. Überprüfen Sie die Programmeingabe mit dem Print-Kommando.

3. Starten Sie das Programm mit dem Go-Kommando.
   Wirkung: Die Signalkombinationen, die Sie am Eingabe-Port einstellen, müssen auch am Ausgabe-Port erscheinen.

4. Beenden Sie den Programmlauf durch Betätigen der RESET-Taste.

5. Ersetzen Sie mit dem Memory-Kommando das Befehlsbyte C3 (Adresse F804) durch das Befehlsbyte CF (Restart-Befehl RST1).

6. Stellen Sie mit den Eingabeschaltern des Eingabe-Ports die Bitkombination 55H ein.

7. Starten Sie das geänderte Programm mit dem Go-Kommando.

G3

Wirkung: Am Ausgabe-Port erscheint die Bitkombination 55 und auf dem Bildschirm die Meldung *** USER ***.
Der Sprungbefehl (C3) zum Anfang des Programms wurde durch den Restart-Befehl (CF) ersetzt. Dadurch ist aus dem zyklischen Programm ein nicht zyklisches Programm mit dem erforderlichen Befehl für die Rückkehr zum Betriebsprogramm geworden.

8. Ersetzen Sie nun mit dem Memory-Kommando das Befehlsbyte CF (Adresse F804) durch das Befehlsbyte FF (Restart-Befehl RST7) und die Einstellung am Eingabe-Port auf die Bitkombination AAH.

9. Starten Sie das geänderte Programm mit dem Go-Kommando.
Wirkung: Auch dieses nicht zyklische Programm wird einmal abgearbeitet (Eingabe-Bitkombination = Ausgabe-Bitkombination) ehe der Rücksprung zum Betriebsprogramm erfolgt. Diesesmal wird jedoch die Meldung *** PROGRAMM-ABORT *** ausgegeben. Eine solche Meldung erfolgt immer dann, wenn der Prozessor im Verlauf seiner Befehlsbearbeitung auf den Datenwert FF trifft. Dies ist z.B. immer der Fall, wenn die Startadresse im G-Kommando in einem Speicherbereich liegt, in dem gar kein RAM-Speicher vorhanden ist.

10. Drucken Sie sich die Inhalte des Betriebsprogramm-RAM's zwischen FC00 und FFFF mit dem Print-Kommando aus.

Sie erkennen das häufige Auftreten des Datums FF. Wenn der Prozessor dieses Byte findet - meist ist das der Fall, wenn ein Anwenderprogramm nicht mehr kontrolliert arbeitet (Fehler im Programm) - wird die weitere Programmbearbeitung abgebrochen.

Mit dem Disassembler-Kommando können Programme, die in Maschinensprache geschrieben sind und sich im Speicher des Systems befinden, in den "Mnemo-Code" oder "Assembler-Code" übersetzt werden.
(Symbolische Namen für Adressen werden immer dann eingesetzt, wenn sie vorher bei der Eingabe des Programms mit dem Assembler-Kommando definiert wurden. Siehe hierzu Beschreibung des Assembler-Kommandos).

Aufruf und Handhabung:

```
KMD > DISASSEMBLER

 START-ADR =X1X1 Y1Y1

 STOP -ADR =X2X2 Y2Y2

(Kommando-Ausführung)
KMD >_
```

D [CR] oder D [SP] eintippen
"ISASSEMBLER" wird ergänzt

X1X1 = Vorgabe; Neu: Y1Y1 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X2X2 = Vorgabe; Neu: Y2Y2 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

Zur Kommando-Ausführung:

- Die zwischen den eingegebenen Start- und Stop-Adressen liegenden Maschinen-Bytes werden disassembliert, d.h., in den zugehörigen Mnemo-Code übersetzt.

- Zur richtigen Übersetzung eines Maschinenprogramms ist es notwendig, daß die Start-Adresse auf ein Befehlsbyte zeigt.

Beispiel für einen Disassembler-Ausdruck:

```
KMD > DISASSEMBLER
 START-ADR =0000 F800
 STOP -ADR =0000 F806
F800 3E 03          MVI  A,03
F802 3D             DCR  A
F803 C2 00F8        JNZ  F800
F806 CF             RST  1
                    END

KMD >_
```

Es werden nur die Speicheradressen angegeben, unter denen das jeweils 1. Byte eines Befehls gespeichert ist. Dieses 1. Byte eines jeden Befehls wird häufig Befehlsbyte genannt.

MAT 85

Name:

D-Kommando

Datum:

D2

1. Laden Sie mit dem M-Kommando ab Adresse F850 das folgende Programm in den Speicher:

Adressen	Daten	Bemerkungen
F850	DB	Befehlsbyte
F851	12	
F852	2F	Befehlsbyte
F853	D3	Befehlsbyte
F854	13	
F855	C3	Befehlsbyte
F856	50	
F857	F8	

2. Disassemblieren Sie das Programm erst ab F850 (richtige Adresse) und dann ab F851 (falsche Adresse). Tragen Sie die verschiedenen Ergebnisse in die vorbereiteten Tabellen ein.

START- ADR= F850
STOP - ADR= F857

Adressen	Daten	Mnemo- Code

START- ADR= F851
STOP - ADR= F857

Adressen	Daten	Mnemo-Code

3. Laden Sie mit dem Memory-Kommando den Wert C3 in die Speicherzeile, die vor Beginn des Programms liegt (F84F). Disassemblieren Sie dann ab F84F und vergleichen Sie dieses Ergebnis mit dem richtigen Programm (ab F850).

4. Fassen Sie die Versuchsergebnisse zusammen!

Mit dem Kommando "Next Instruction" (nächsten Befehl bearbeiten) läßt sich ein Anwenderprogramm schrittweise abarbeiten. Nach jedem ausgeführten Befehl werden die Inhalte der CPU-Register protokolliert. Die Anzahl der abzuarbeitenden Befehle (die Steps oder Schritte) kann vorgewählt werden. Die Programmausführung läuft nicht in Echtzeit ab.
Das N-Kommando kann besonders dazu dienen, die Wirkung einzelner Befehle zu studieren und den Lauf eines zu testenden Programms zu verfolgen.

Aufruf und Handhabung:

```
KMD > NEXT INSTRUCTION

 START-ADR =XXXX YYYY

 STEPS     =XX YY

(Kommando-Ausführung)
KMD >_
```

N [CR] oder N [SP] eintippen
"EXT INSTRUCTION" wird ergänzt

XXXX = Vorgabe; Neu: YYYY [CR] oder [SP]
Vorgabe: [CR] oder [SP]

XX = Vorgabe; Neu: YY [CR] oder [SP]
Vorgabe: [CR] oder [SP]

YY: zwischen 00 und 99 möglich

Zur Kommando-Ausführung:

- Die nach der Startadresse (XXXX oder YYYY) folgenden (XX bzw. YY) Befehle werden ausgeführt. Nach jedem abgearbeiteten Befehl werden die Inhalte der CPU-Register ausgedruckt.

- Wenn sich im Anwenderprogramm ein Halt-Befehl (HLT, 76H) oder ein unbekannter Befehls-Code befindet, wird die weitere Bearbeitung des Programms abgebrochen und folgende Meldung ausgedruckt:

```
*** HALT ODER ILLEGALER OPCODE ***
```

- Bei mehrmaligem Aufruf des N-Kommandos wird das Anwenderprogramm an der jeweils vorher unterbrochenen Stelle fortgesetzt.

- Die Start-Adresse muß auf ein Befehlsbyte des zu untersuchenden Programms zeigen.

Beispiel für die Ausführung des N-Kommandos:
(Das Programm wurde vorher mit dem Memory-Kommando in den Speicher geladen).

```
KMD > NEXT INSTRUCTION
 START-ADR =1E09 F850
 STEPS     =  0 4
 PC   LABEL:  OP   ADR.FELD  A  NZHPC B  C  D  E  H  L  I  SP
 F850         IN   12        F0 00000 00 00 00 00 00 00 80 FC32
 F852         CMA            0F 00000 00 00 00 00 00 00 80 FC32
 F853         OUT  13        0F 00000 00 00 00 00 00 00 80 FC32
 F855         JMP  F850      0F 00000 00 00 00 00 00 00 80 FC32
 F850         IN   12
KMD > _
```

Dieser Befehl wird nicht mehr ausgeführt (5. Schritt).

Die Abkürzungen der Kopfzeile und ihre Bedeutungen:

PC-Programm Counter (Programmzähler, 16 Bit); unter dieser Spalte werden die Adressen der Speicherzeilen angezeigt, die das Befehlsbyte des jeweiligen Befehls enthalten.

LABEL: Symbolische Adresse (siehe Assembler-Kommando).

OP Operations-Code; enthält den mnemonischen Code des Befehlsbytes.

ADR.FELD Adreß-Feld; enthält Adressen bzw. Daten zum Befehl.

A - REGISTER A (Akkumulator, oft kurz Akku, 8 Bit)

N - NEGATIV -FLAG (Negativ-Bit)
Z - ZERO -FLAG (Null-Bit)
H - HALF CARRY-FLAG (Zwischenübertrags-Bit)
P - PARITY -FLAG (Paritäts-Bit)
C - CARRY -FLAG (Übertrags-Bit)

} FLAG-Bits des Prozessor-Status-Registers (Zustandsregister)

B - REGISTER B (8 Bit)
C - REGISTER C (8 Bit)
D - REGISTER D (8 Bit)
E - REGISTER E (8 Bit)
H - REGISTER H (8 Bit)
L - REGISTER L (8 Bit)
I - INTERRUPT CONTROL REGISTER (Interrupt-Masken-Register, 8 Bit)
SP - STACK POINTER (Stapel-Zeiger, 16 Bit)

Gliederung:

Aufruf des Assemblers ____ A1

Aufbau einer Assembler-Befehlszeile ____ A2

Begrenzungszeichen für die Felder der Befehlszeile ____ A3

Fehlermeldungen, Fehlerbeseitigung ____ A4

Programmieren mit Marken (Label) (Einführung) ____ A5

Wie behandelt der Assembler eine Marke? ____ A6
Regeln für den Gebrauch von Marken

Fehlermeldungen beim Umgang mit Marken ____ A7

Anweisungen zum Löschen und Ausdrucken der Marken ____ A8

Assembler-Anweisungen: ORG ____ A9, A10
EQU ____ A11
DB ____ A12
DW ____ A13
END ____ A14

Operations-Symbole + und - ____ A15

Formatierte Programm-Eingabe ____ A16

A-Kommando

A1

Mit dem Assembler-Kommando wird ein Hilfsprogrammm, das man Assembler nennt, aufgerufen. Aufgabe dieses Hilfsprogramms ist die Übersetzung jeder im mnemonischen Code eingegebenen Programmzeile in den zugehörigen Maschinencode. Der Maschinencode wird dann ab einer bestimmten Speicheradresse in den RAM-Speicher geschrieben. Diese Speicheradresse wird nach Aufruf des A-Kommandos durch die Eingabe einer "START-ADRESSE" festgelegt.
Um mit dem "Assemblerprogramm" (kurz Assembler) arbeiten zu können, bedarf es der Einhaltung einiger Regeln, die im folgenden schrittweise erklärt und geübt werden.

Aufruf des Assemblers:

```
KMD > ASSEMBLER

 START-ADR =XXXX YYYY

XXXX od. YYYY  (Programm
                eintippen)
                   !
                   !
                   !
                   !
                  END

*** RESTART ? (Ja/Nein)

KMD > _
```

A [CR] oder A [SP] eintippen
"SSEMBLER" wird ergänzt

XXXX = Vorgabe; Neu: YYYY [CR] oder [SP]
Vorgabe: [CR] oder [SP]

Siehe folgende Beschreibungen

Jede eingetippte Zeile muß mit [CR] abgeschlossen werden!

END [CR]; Jedes Programm muß mit END beendet werden

J [CR] oder N [CR] eintippen.
Sinn und Beschreibung folgen.
Nächstes Kommando wird erwartet.

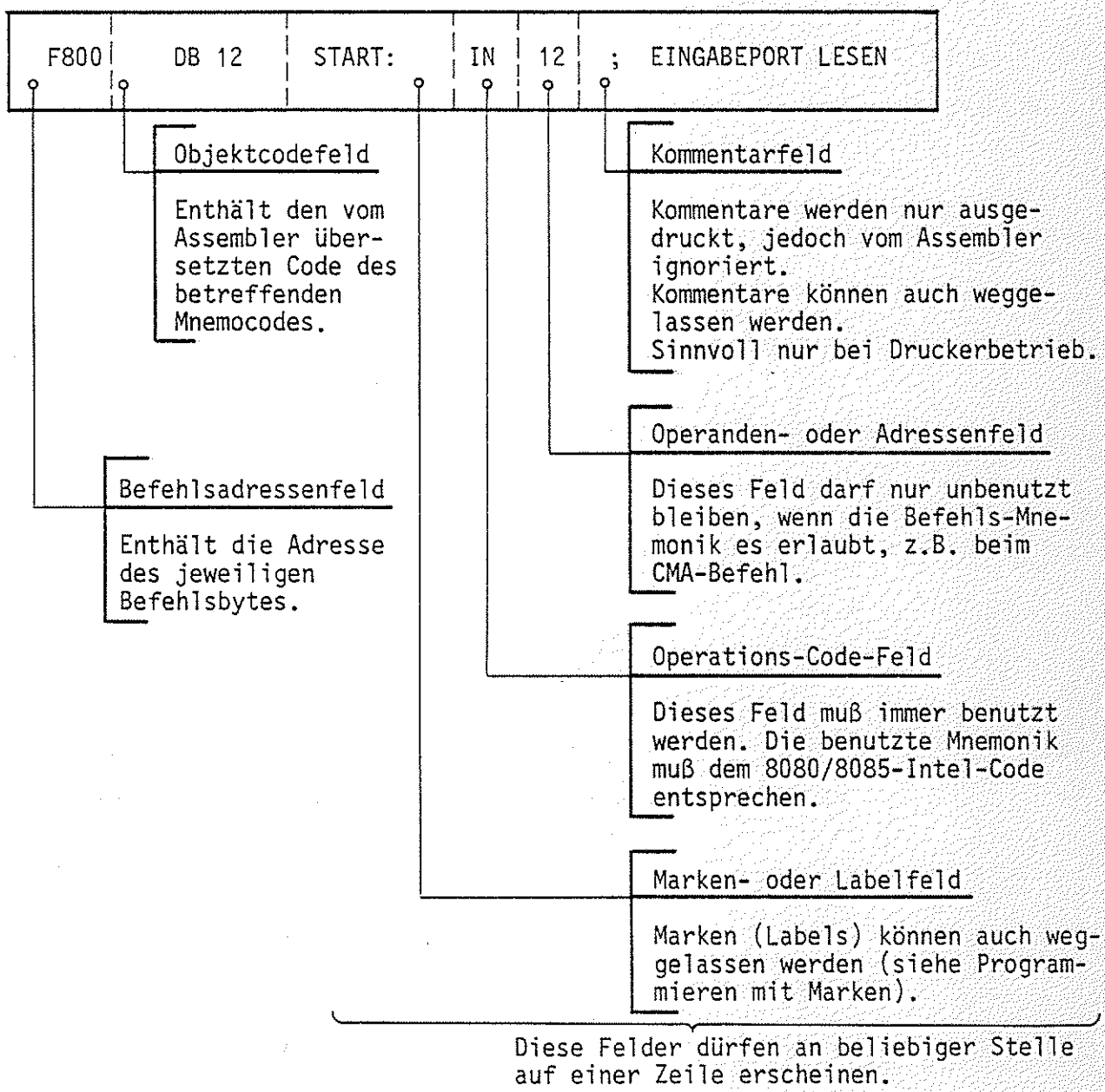
Übung:
Rufen Sie den Assembler auf, geben Sie die RAM-Start-Adresse F800 und den einzigen "Befehl" END ein und üben Sie den Austritt aus dem Assembler mit den verschiedenen "RESTART-Antworten".

Aufbau einer Assembler-Befehlszeile:

Rufen Sie den Assembler auf und geben Sie die Startadresse F800 ein.
Tippen Sie folgende Zeile ein:

START: IN 12; EINGABEPORT LESEN CR

(CR wird im folgenden nicht mehr angegeben).
Die nach der Assemblierung ausgedruckte Zeile (Befehlszeile) kann in die dargestellten Felder zerlegt werden:

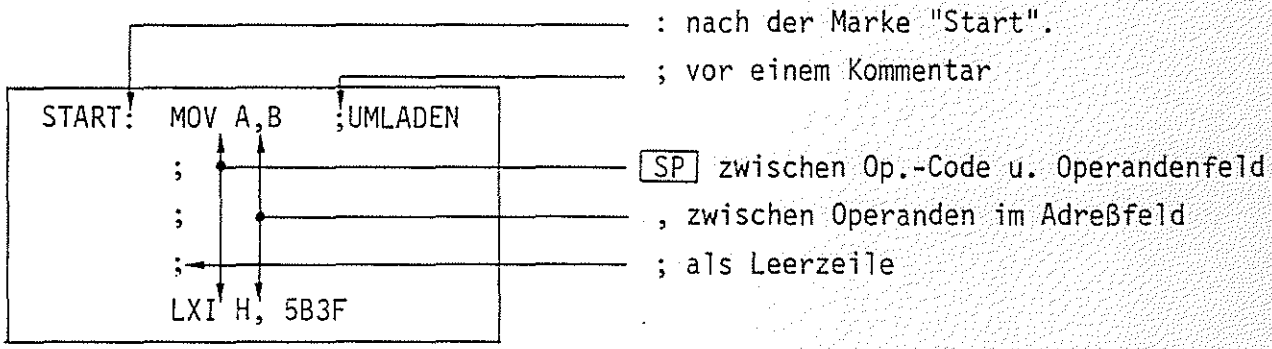


Damit der Assembler die Felder trennen kann, müssen ihm Begrenzungszeichen am Ende oder Beginn eines Feldes mitgeteilt werden.

Begrenzungszeichen für die Felder:

Zwischenraum [SP]	zwischen Operations-Code- und Operandenfeld
,	zwischen Operanden im Adressenfeld
;	1. vor einem Kommentar 2. wenn als einziges Zeichen in der Zeile, zur Erzeugung einer Leerzeile (wird vom Disassembler ignoriert)
:	Unmittelbar nach einer Marke

Beispiele:



Übung:

Tippen Sie das folgende Programm ab Startadresse F900 ein. Füllen Sie in der Tabelle das Befehlsadressenfeld und das Objektcodefeld aus. Kontrollieren Sie Ihre Eingabe mit dem Disassembler. Starten Sie das Programm.

Mit Schalter B0 des E-Ports müssen sich die vier unteren LED's (B4-B7) des A-Ports ein- oder ausschalten lassen.

F900	IN 12 ;EINGABEPORT LESEN
	ANI 01 ; B0 BETÄTIGT ?
	JNZ 0F90E ;NEIN,ALLE LEDS AUS
	MVI A,0F0;JA,LEDS B4-B7 EIN
	OUT 13
	JMP 0F900;ZURÜCK ZUM START
	MVI A,00;ALLE LEDS AUS
	OUT 13
	JMP 0F900
	END ;ENDE

Besonderheit: Wenn im Operanden- oder Adressenfeld eine Zahl mit A-F beginnt, muß dieser Zahl eine 0 vorgestellt werden (Erklärung später).

Fehlermeldungen:

Unbekannte und fehlerhaft eingegebene Befehle und Daten (Op.-Codes und Operanden) werden ignoriert (nicht angenommen) und mit "?" unterhalb der Zeile in der Umgebung des Fehlers kenntlich gemacht. Die Befehlsadresse wird nicht weitergezählt.

Beispiele für typische Fehler:

1.	F907 3E 00 MVI A, 0F0 ;..... ? F907	Nach, darf kein [SP] sein F907 wurde nicht geändert
2.	F909 OUT13 ? F909	[SP] zwischen Op.-Codefeld und Operand fehlt
3.	F912 C3 0000 JMP F900 F915	keine Fehlermeldung, jedoch wurde falsche Adresse assembliert (0000).0 vor F900 vergessen!

Fehlerbeseitigung:

- Fehler 1 und 2 können berichtigt werden, indem einfach die richtige Zeile eingegeben wird, denn die Befehle wurden ignoriert.

- Fehler der Art 3 können zunächst berichtigt werden, indem man nach Eingabe aller Befehle und Abschluß mit END etc. wieder den Assembler aufruft, und die START-ADR. auf die Adresse des zu ändernden Befehls setzt (hier z.B. "START-ADR = XXXX F912" [CR].
  Geben Sie dann den richtigen Befehl ein und verlassen Sie den Assembler mit der Taste [ESC]

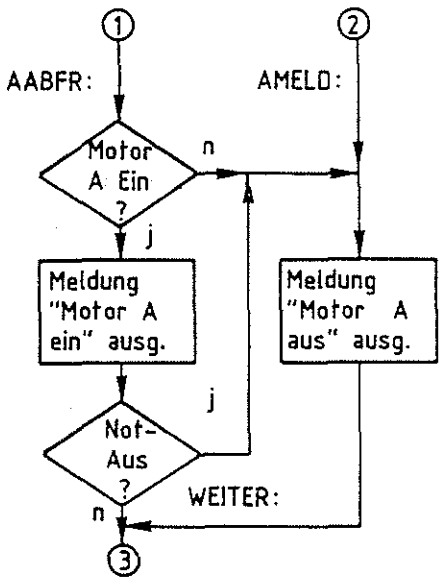
- Wenn Sie einen Fehler vor Abschluß der Zeileneingabe mit [CR] bemerken, können Sie die Fehleingabe mit Taste [DEL] löschen und den richtigen Text eingeben.

Programmieren mit Marken (Label): (Label = Markierung)

Bei der Entwicklung eines Programms für einen Mikrocomputer zur Lösung irgend einer Aufgabe geht man so vor:

- Definition der zu lösenden Aufgabe
- Erstellung eines Flußdiagramms
- Schreiben des Programms
- Testen des Programms, Fehlersuche und deren Beseitigung, dabei evtl. Änderung des Flußdiagramms und des Programms
- Dokumentation

Das folgende Bild zeigt ein Flußdiagramm für ein Programm, das noch zu erstellen ist.



Beim Entwurf des Flußdiagramms weiß man noch nicht, wo das spätere Programm im Speicher liegen wird und wieviele Programmschritte zur Lösung der Blöcke (z.B. Not-Aus?) nötig sein werden. Um trotzdem die Sprungziele kennzeichnen zu können - z.B. wohin, wenn Motor A ausgeschaltet oder wenn Not-Aus betätigt? - bedient man sich der Hilfe von Marken oder Labels.

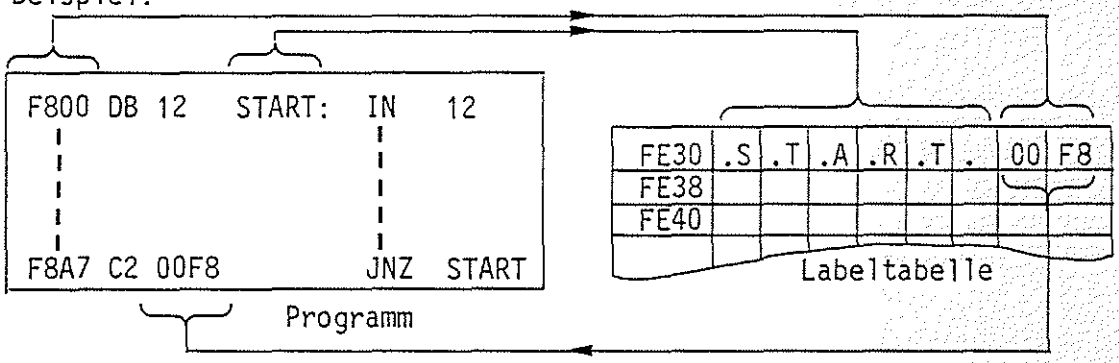
> Marken oder Label sind Namen für Adressen, deren Werte während des Programmierens noch nicht bekannt sind.

- Marken werden am häufigsten in Sprung-Aufruf- und Verzweigungsbefehlen verwendet.
- Marken erleichtern das Auffinden von Programmstellen.
- Während des Programmierens braucht man sich nicht um die Berechnung von Adressen zu kümmern.
- Marken machen Programme verständlicher.

Wie behandelt der Assembler eine Marke?:

Wenn das Markenfeld einer Befehlszeile eine Marke enthält, trägt der Assembler diese Marke und die zugehörige Adresse des Befehlsbytes in eine "Labeltabelle" im Betriebsprogramm-RAM ein.
Man kann danach diese Marke als Adresse (oder als Datum) im Adressenfeld eines anderen Befehls verwenden. Der Assembler ersetzt dann die Marke durch den Adressenwert aus der Labeltabelle, wenn er das Maschinenprogramm erzeugt.

Beispiel:



Das Setzen einer Marke mit nachfolgendem Doppelpunkt in das Marken- oder Labelfeld nennt man auch "Definieren einer Marke oder eines Labels".

Regeln für den Gebrauch von Marken:

- Marken dürfen eine Länge von 1 bis 6 Zeichen haben, das erste Zeichen muß ein Buchstabe sein.

- Damit der Assembler zwischen den Hex-Zahlen A-F und dem 1. Buchstaben einer Marke unterscheiden kann, muß den Hex-Zahlen A-F eine 0 vorangestellt werden.

- Im Programmverlauf erst später definierte Marken dürfen vorher schon im Adreßfeld benutzt werden. Der Assembler trägt die zugewiesenen Adressen nach, sobald sie definiert werden.

- Ein Markenname darf nur einmal definiert werden.

- Mehr als 57 Marken sind nicht erlaubt.

Fehlermeldungen beim Umgang mit Marken:

— Wird eine Marke innerhalb eines Programms mehrfach definiert, so macht der Assembler nach Abschluß der Zeile durch [CR] mit einem "?" auf diesen Fehler aufmerksam. Die gleiche Fehlermeldung tritt auch dann auf, wenn in einem ganz anderen Programm dieser Markenname schon einmal benutzt wurde (häufig hat man mehrere Übungsprogramme in verschiedenen Speicherbereichen gespeichert).

● Abhilfe: Verwenden Sie an der Stelle eine neue Marke!

— Die oben genannte Fehlermeldung tritt auch dann auf, wenn Sie sich beim Eintippen einer Befehlszeile mit Marke im Operationscode- oder im Operandenfeld vertippt haben und im zweiten Anlauf versuchen, diesen Fehler zu beheben. Die Marke ist vom Assembler bereits angenommen und wird bei erneuter Eingabe als "schon definiert" behandelt.

● Abhilfe: Geben Sie nur den Mnemo-Code neu ein!

— Alle verwendeten Marken (auch die aus anderen Programmen) werden immer dann automatisch nach Abschluß der Programmeingabe mit "END" ausgedruckt, wenn das Programm noch undefinierte Marken enthält. Diese werden dann mit * gekennzeichnet. Die rechts daneben angegebene Adresse zeigt auf den Speicherplatz, der das niederwertige Byte der zur Marke gehörenden Adresse enthält.

Beispiele:

```
E000 DB 12          START:   IN 12

E002 C2 0000        JNZ E012

E005 C3 0000        JMP WEITER

E008                END
E012  *E003
START  E000
WEITER*E006
```

Start wurde auf E000 definiert

Weil vor E012 die 0 fehlt, faßt der Assembler die Adresse als Marke auf. Da sie noch nicht definiert ist, wird zunächst 0000 eingesetzt.
WEITER ist ebenfalls noch undefiniert.
Abschluß mit END.
Automatischer Ausdruck der Marken:
"E012" wird als nichtdefinierte Marke ausgedruckt (da mit Buchstaben beginnend), ebenso "WEITER". "START" ist definiert.

Anweisungen an den Assembler zum Löschen und Ausdrucken der Marken:

— Löschen der Marken

Wenn man ohne Rücksicht auf bereits vorher verwendete Marken ein neues Programm eingeben möchte, muß die im RAM liegende Labeltabelle (ab FE30) gelöscht werden. Nach dem Löschen sind alle, auch in früher eingegebenen Programmen, verwendeten Marken verschwunden. Die Programme bleiben trotzdem lauffähig, denn ihr Maschinen-Code befindet sich ja noch im Speicher. Lediglich beim Disassemblieren der Programme fehlen die ursprünglich verwendeten Marken.

Das Löschen der Labeltabelle erfolgt nach dem Eintritt in den Assembler durch Eingeben der Anweisung "LC" (Label Clear).

Beispiel:

```
KMD > ASSEMBLER
 START-ADR =F800 E000
E000             LC
E000       —
```

Aufruf des Assemblers;
Löschen der Labeltabelle;
Programmeingabe wird erwartet;

— Ausdrucken der Marken

Wollen Sie sich zur Orientierung nach dem Eintritt in den Assembler (z.B. um das Programm zu ändern) oder während des Programmierens oder am Ende der Programmeingabe die bisher verwendeten Marken ausdrucken lassen, so müssen Sie die Anweisung "LP" eingeben. Die Befehlsadresse wird dadurch nicht verändert. (LP = Label Print)

Beispiel:

```
   :                :
   :                :
   :                :
   :                :
E100 7D          MOV A,L
E101 C3 0000     JMP WARTE
E104             LP
START  E000
WARTE *E102
Z1     E01B
Z2     E01D

E104           —
```

Ausdruck aller verwendeten Marken, auch der evtl. noch nicht definierten (mit *) und der in anderen Programmen verwendeten.

Weiter mit Programmeingabe.

Assembler-Anweisungen:

Assembler-Anweisungen sind Anweisungen für das Assemblerprogramm, die nicht in Maschinensprache übersetzt werden. Mit ihrer Hilfe läßt sich z.B. ein Maschinenprogramm einem bestimmten Speicherbereich zuweisen, oder ein RAM-Bereich für die Ablage von Datenbytes oder Adressen festlegen.

Um diese Anweisungen - man nennt sie auch Pseudo-Operationen (vorgetäuschte Op.) - zu verwenden, müssen Sie die Mnemonik dieser Anweisungen in das Op.-Codefeld und Adressen oder Daten (falls erforderlich) in das Adressenfeld setzen.

Dieser Assembler gestattet die Verwendung folgender Pseudo-Operationen:

ORG	— ORIGIN (Ursprung)
EQU	— EQUATE (Gleichsetzen)
DB	— DEFINE BYTE, definiere Byte, 8-Bit
DW	— DEFINE WORD, definiere Wort, 16-Bit
LP, LC	— LABEL-PRINT u. CLEAR (bereits erklärt)
END	— Ende einer Programmeingabe

— Die ORG-Anweisung:

Mit dieser Anweisung wird die Befehlsadresse neugesetzt. Der Assembler erhält dadurch Bescheid, ab welcher Adresse er die Maschinensprache der folgenden Befehle in den Speicher schreiben soll. Es können mehrere ORG's an verschiedenen Stellen im Programm verwendet werden. Mit ORG kann man auch zu bereits verlassenen Adressen zurückkehren, um dort z.B. eingegebene Fehler zu berichtigen.

Beispiele:

= Überspringen eines Speicherbereiches

```
E010 3E 04        MVI A,04
E012 2F           CMA
E013              ORG 0FA00
FA00              —
```

— Die Befehlsadresse wird von E013 auf FA00 gesetzt. Die 0 vor der Adresse dient der Unterscheidung der Adresse von einer Marke.

= Nachträgliches Berichtigen einer Befehlszeile

```
F900 79           MOV A,C
F901 00           NOP
F902 00           NOP
F903 00           NOP
F904 A6           ANA M
F905              ORG 0F900
F900 78           MOV A,B
F901              ORG 0F905
F905              —
```

— Hier bemerkt der Programmierer, daß er sich bei Adresse F900 vertippt hatte. Nach Rückkehr zu F900 mit der ORG-Anweisung und Berichtigung des Fehlers springt er mit ORG 0F905 zur ursprünglichen Befehlsadresse vor.

= Label in der ORG-Anweisung

```
E00A 06 FF ZEIT:  MVI B, 0FF
E010 3E FE        MVI A,0FE
E012              ORG ZEIT
E00A              —
```

— Die Adresse ist hier durch die Marke "Zeit" angegeben. Ihr wurde vorher im Programm E00A zugewiesen.

— Die EQU-Anweisung:

Mit dieser Anweisung können bestimmten Adressen oder Daten Namen (Marken) zugeordnet werden. Diese Namen und die ihnen gleichgesetzten (zugewiesenen) Adressen oder Daten werden ebenfalls in die Labeltabelle eingetragen.
Mit der EQU-Anweisung werden jedoch keine Daten in den Programmspeicher geladen.

Setzen Sie EQU-Anweisungen immer an den Anfang einer Programms, es wird dadurch besser lesbar.

Beispiel:

```
KMD > ASSEMBLER
 START-ADR =E010 F800
F800              LC                 — alte Labeltabelle löschen
F800              CR EQU 0D          — Namen CR wird 0D zugewiesen
F800              EPORT EQU 12       — Namen EPORT wird Adr. 12 zugewiesen
F800              APORT EQU 13       — Namen APORT wird Adr. 13 zugewiesen
F800              ;                  — Leerzeile zur Trennung
F800 DB 12        IN EPORT             Der Assembler ersetzt bei der
F802 2F           CMA                  Erzeugung des Maschinen-Codes
F803 D3 13        OUT APORT          —die Namen durch die ihnen zuge-
F805 3E 0D        MVI A,CR             wiesenen Daten oder Adressen.
F807               |
                   |
```

Ein Blick in die Labeltabelle:

```
KMD > PRINT
 START-ADR =FE30
 STOP -ADR =FE48
 FORMAT    =A
 FE30 .A .P .0 .R .T .  13 00
 FE38 .C .R .  .  .  .  0D 00
 FE40 .E .P .0 .R .T .  12 00
 FE48 FF
```

Die EQU-Anweisung kann auch dazu dienen, nicht definierte Marken nachzudefinieren:

— Wenn ein Programm bereits mit "END" abgeschlossen wurde, müssen Sie dazu den Assembler neu aufrufen und die mit "*" gekennzeichneten Marken definieren.

— Wenn Sie sich noch im Programm befinden, können Sie dies bei der gerade aktuellen Befehlsadresse tun, denn sie wird dadurch ja nicht verändert.

— Die DB-Anweisung:

Diese Anweisung setzt Datenbytes oder ASCII-Zeichen in den Programmspeicher und zwar ab der Adresse, bei der die DB-Anweisung erteilt wird.

Beispiele:

= Absetzen von Datenbytes

```
   ¦               ¦
   ¦               ¦
F800 0A3F5BFF    DB 0A,3F,5B,0FF,7C,3A,0E4,55,3E
F804 7C3AE455
F808 3E          —
```

Inhalt des Programmspeichers

- DB und erstes Byte durch Leerzeichen trennen;
- Bytes durch Komma trennen;
- max. 20 Bytes pro DB möglich;
- nach letztem Byte kein Komma;
- vor A-F 0 setzen;

Es empfiehlt sich, jeweils höchstens die Bildschirmzeile zu füllen und dann mit CR abzuschließen. Sollen mehr Bytes abgesetzt werden, erneut DB verwenden.

= Absetzen von ASCII-Zeichen

```
   ¦               ¦
   ¦               ¦
F900 5343484E    DB 'SCHNAPS STINKT'
F904 41505320
F908 5354494E
F90C 4B54        —
```

- Zeichen in ' ' einschließen;
- max. 40 Zeichen pro DB möglich;

= Absetzen von ASCII- und Hex-Zeichen

```
   ¦               ¦
   ¦               ¦
FA00 FA3A4D49    DB 0FA,3A,'MILCH ',0D,'KLARER',00
FA04 4C434820
FA0B 0D4B4C41
FA0C 52455200
FA10             —
```

Übung:

Probieren Sie die gezeigten Beispiele aus!

— Die DW-Anweisung:

Mit der DW-Anweisung kann man 16-Bit-Worte in den Speicher eingeben. Die Anzahl der Worte pro Zeile sollte 9 nicht übersteigen (Bildschirmzeile voll).

Beispiele:

= Absetzen von Adressen

```
FA00 3412FCAA    DW 1234,0AAFC,55AF,0F800
FA04 AF5500F8    —
```

im Speicher sind höher- und niederwertige Bytes vertauscht.

- DW und erste Adresse durch Leerzeichen trennen;
- Adressen durch Komma trennen;
- nach letzter Adresse kein Komma;
- vor A-F 0 setzen;

= Absetzen von Adressen, die vorher durch die EQU-Anweisung definiert wurden

```
E010             AADR EQU 1234
E010             EADR EQU 0F850
E0F0 341250F8    DW AADR,EADR
E0F4             —
```

DB- und DW-Anweisungen setzt man häufig ans Ende eines Programms.

— Die END-Anweisung:

An der END-Anweisung erkennt der Assembler das Ende der Programmeingabe. Nach Abschluß dieser Anweisung mit [CR] fragt der Assembler mit dem Ausdruck

```
*** RESTART ? (JA/NEIN)
```

danach, ob er einen RST-1-Befehl ans Ende des Programms setzen soll (J-[CR]) oder nicht (N-[CR] oder [CR]). Dieser Befehl bewirkt einen Sprung ins Betriebsprogramm.
Mit einem solchen Rücksprung ins Betriebsprogramm soll verhindert werden, daß durch unkontrollierten Lauf des Computers wichtige Speicherinhalte im Betriebsprogramm-RAM überschrieben werden. Abschluß eines "Endlos-Programms" mit RST1 bewirkt den Ausdruck:

```
*** USER ***
```

(Anwender, Benutzer)

und Rückkehr in die "KMD>-Routine".

Beispiel:

```
ASSEMBLER
 START-ADR =E000
E000 DB 12        START:IN 12
E002 E6 04        ANI 04
E004              END
 *** RESTART ? (JA/NEIN) J
```

— Eingabe eines Programms ohne ein Ende durch Rücksprung zum Anfang (Schleife).

— Restart-Frage mit "Ja" beantwortet.

```
KMD > DISASSEMBLER
 START-ADR =E000
 STOP -ADR =E007 E004
E000 DB 12   START:  IN   12
E002 E6 04           ANI  04
E004 CF              RST  1
```

— Der Assembler hat einen RST-1-Befehl ans Programmende gesetzt.

```
KMD > GO
 START-ADR =E0F5 E000

 *** USER ***
```

— Start des Programms bewirkt Rücksprung ins Betriebsprogramm.

Operations-Symbole + und —:

Dieser Assembler gestattet die Verwendung der Operations-Symbole "+" und "-" in Verbindung mit der Verarbeitung von Marken, Daten und Adressen. Addition und Subtraktion erfolgen dabei hexadezimal. Überläufe (etwa FE+3) dürfen nicht auftreten. Durch die Verwendung der Operations-Symbole lassen sich häufig Marken einsparen. Dies kann bei langen Programmen notwendig werden, da die Labeltabelle nur 57 Marken aufnehmen kann.

Beispiele:

— Einsparung einer Marke

```
KMD > ASSEMBLER
 START-ADR =E000 F800
F800                LC
F800 DB 12          START:IN 12
F802 06 07          MVI B,07
F804 4B             MOV C,B

F81B C3 04F8        JMP START+4
F81E
```

— In dieser Zeile wird eine Marke gespart

● Das "+" muß ohne Zwischenraum auf "START" folgen;
(Der Sprung muß auf eine Adresse zeigen, unter der ein Befehlsbyte steht).

— Operations-Symbole in Verbindung mit Assembler-Anweisungen:

```
KMD > ASSEMBLER
 START-ADR = F800
F800                LC
F800                EPORT EQU 12
F800                APORT EQU EPORT+1
F800                ;
F800 DB 12          START:IN EPORT
F802 2F             CMA
F803 D3 13          OUT APORT
F805                ORG START+F5
                                  ?
F805                ORG START+0F5
F8F5 C3 00F8        JMP START
F8F8                END
 *** RESTART ?  (JA/NEIN)
```

— Die APORT-Adresse ergibt sich aus der Addition.

— Die APORT-Adresse ist berechnet.
— Hier fehlt die 0 vor F.

— Richtige Eingabe.
— Neue Befehlsadresse

Formatierte Programm-Eingabe:

Mit der Taste [→] kann der Cursor um je 8 Schreibstellen nach rechts versetzt werden.
Dadurch ist während der Eingabe des Operations-Codes eine übersichtlichere Darstellung aller Zeichen auf dem Bildschirm möglich.
Unabhängig vom Format der Programme im Assemblerbetrieb druckt der Disassembler die Programme jedoch formatiert aus.

Beispiele:

— Unformatiertes Assemblerprogramm

```
KMD > ASSEMBLER
 START-ADR =F800
F800              LC
F800              EPORT EQU 12
F800              APORT EQU 13
F800 DB 12        IN EPORT
F802 D3 13        OUT APORT
F804 C3 00F8      JMP START
F807              END
 *** RESTART ? (JA/NEIN)
```

— Formatiertes Assemblerprogramm

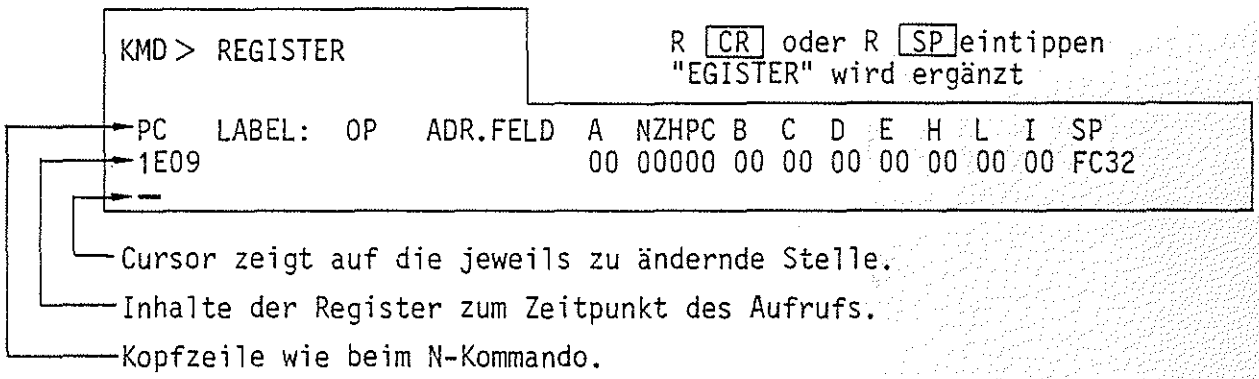
```
KMD > ASSEMBLER
 START-ADR =F800
F800              LC
F800              EPORT EQU 12
F800              APORT EQU 13
F800 DB 12        START:  IN EPORT
F802 D3 13                OUT APORT
F804 C3 00F8              JMP START
F807              END
 *** RESTART ? (JA/NEIN)
```

R1

R-Kommando

Mit dem Register-Kommando können die Inhalte der CPU-Register angezeigt und vor dem Start eines Anwender-Programms vorbelegt werden.

Aufruf und Handhabung:



Zur Kommando-Ausführung:

- Es können nur die Inhalte der Register, auf die der Cursor zeigt, geändert werden.
- Korrekturen eingegebener Werte sind mit der DEL -Taste möglich, sofern noch nicht mit SP abgeschlossen wurde.
- Mit der SP-Taste kann der Cursor von Register zu Register bewegt werden. Mit ihr beendet man auch die Änderung eines Registerinhaltes.
- Es lassen sich nur Hex-Werte in die Register eingeben; nur die Stelle, unter der sich der Cursor befindet, kann geändert werden.
- In die einzelnen Bits des Status-Registers lassen sich nur Binärwerte (0,1) eingeben.
- Mit der CR-Taste beendet man alle Eingaben, das nächste Kommando wird erwartet.

Die Ausgedruckten bzw. eingegebenen Registerinhalte verbleiben zunächst im Schreib-Lese-Speicher. Diese Speicherstellen nennt man auch "Schattenregister". Bevor ein Anwender-Programm gestartet wird, werden die Inhalte der Schattenregister durch das Betriebsprogramm in die CPU-Register geladen.

Beim Experimentieren mit dem R-Kommando sollte man den Inhalt des "Stack-Pointers" (SP) nicht verändern!

● Wenn sich ein Anwender-Programm nicht starten läßt, obwohl es richtig eingegeben wurde (Prüfen z.B. mit dem D-Kommando), hat das Betriebsprogramm durch vorhergegangene Bedienungsfehler den Inhalt von SP geändert (meist um FC80H). Sehen Sie sich mit dem R-Kommando diesen Inhalt an und stellen Sie ihn gegebenenfalls wieder auf FC32.
Danach läßt sich das Anwenderprogramm starten.

MAT 85

Name:

R-Kommando

Datum:

R2

Rufen Sie das R-Kommando auf.
Tragen Sie die angezeigten Registerinhalte in die erste Zeile der folgenden Tabelle ein. Ändern Sie dann die Inhalte der Register in die Werte, die in der zweiten Zeile der Tabelle angegeben sind (keine Angabe ≙ keine Änderung).

	KMD > REGISTER											
	PC	LABEL: OP ADR.FELD	A	NZHPC	B	C	D	E	H	L	I	SP
1.												
2.	F800		55	10101		AA		FF	2A		60	FC30

Nun soll überprüft werden, ob die Register die neuen Werte enthalten.

Rufen Sie hierzu das R-Kommando erneut auf.
Tragen Sie die angezeigten Registerinhalte in die Tabelle ein und vergleichen Sie diese Anzeige mit der 1. Zeile der obigen Tabelle.

KMD > REGISTER											
PC	LABEL: OP ADR.FELD	A	NZHPC	B	C	D	E	H	L	I	SP

B-Kommando **B1**

Mit dem Breakpoint-Kommando (Breakpoint = Haltepunkt) wird das Einsetzen von Haltepunkten in Anwender-Programme freigegeben bzw. gesperrt.
Dieses Kommando ermöglicht es, bestimmte Programmteile (z.B. Zeitschleifen) in Echtzeit durchlaufen zu lassen und ab dem Haltepunkt die Programmausführung mit dem N-Kommando schrittweise zu beochachten. Die Eingabe der Haltepunkt-Adressen erfolgt innerhalb der Ausführung des Go-Kommandos nach der Eingabe der Programm-Startadresse.

Aufruf und Handhabung:

```
KMD > BREAKPOINT

 BREAK-ADR1=X1X1
 BREAK-ADR2=X2X2
 BREAK-ADR3=X3X3
 BREAK-ADR4=X4X4

 EIN/AUS  =X Y
```

B [CR] oder B [SP] eintippen
"REAKPOINT" wird ergänzt

X1X1 = Breakadresse 1 } Änderungen nur unter Go-Kommando möglich
⋮
X4X4 = Breakadresse 4 }

X = Vorgabe; Ein: Y = E [CR] oder E [SP]
Aus: Y = A [CR] oder A [SP]
Unverändert: Y = [CR] oder [SP]

Zur Kommando-Ausführung:

Das Breakpoint-Kommando wird unter dem Go-Kommando ausgeführt, wenn...

- Breakpoints eingeschaltet sind und
- Breakadressen nicht alle 0 sind und
- das Anwenderprogramm eine Adresse erreicht, auf die ein Breakpoint gesetzt ist. Diese Adresse muß auf ein Befehlsbyte zeigen.

Übung:

Rufen Sie das B-Kommando auf und schalten Sie die Breakpoint-Routine ein bzw. aus.

Einsetzen der Break-Adressen:

Die Break-Adressen werden bei eingeschaltetem "Breakpoint" unter dem Go-Kommando wie folgt eingesetzt:

```
KMD > GO

 START-ADR =XXXX YYYY

 BREAK-ADR1=X1X1 Y1Y1
 BREAK-ADR2=X2X2 Y2Y2
 BREAK-ADR3=X3X3 Y3Y3
 BREAK-ADR4=X4X4 Y4Y4
```

G [CR] oder G [SP] eintippen
"0" wird ergänzt

XXXX = Vorgabe; Neu: YYYY [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X1X1 = Vorgabe; Neu: Y1Y1 [SP] *)
Vorgabe: [SP]

X4X4 = Vorgabe; Neu: Y4Y4 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

*) Wenn alle vier Break-Adressen gesetzt werden sollen, bei ADR1 - ADR3 mit [SP] abschließen!
Wenn weniger als vier Break-Adressen gesetzt werden sollen, jeweils mit [CR] abschließen!

Breakpoint-Ausführung:

Wenn die CPU bei der Ausführung des Anwender-Programms eine Break-Adresse erreicht hat, wird die weitere Programmausführung gestoppt und in das Betriebsprogramm zurückgesprungen. Das Betriebsprogramm meldet sich mit folgendem Ausdruck (Beispiel):

```
*** BREAKPOINT ***
PC  LABEL:   OP   ADR.FELD  A  NZHPC B  C  D  E  H  L  I  SP
                            87 00000 00 00 00 00 F8 2A 80 FC34
F802         OUT  02
KMD > _
```

└ Befehl, auf den die Break-Adresse zeigt.
Dieser Befehl ist noch nicht ausgeführt worden!

└ Break-Adresse

Der Rücksprung in das Betriebsprogramm nach dem Erreichen einer Haltepunkt-Adresse erfolgt dadurch, daß das Betriebsprogramm vor der Ausführung des Go-Kommandos einen Rücksprungbefehl (RST 4) in das Anwenderprogramm einbaut. Dazu wird das ursprünglich vorhandene Befehlsbyte aus dem Anwenderprogramm im RAM zwischengespeichert und nach dem Erreichen der Haltepunkt-Adresse wieder eingesetzt. Außerdem werden alle Register-Inhalte der CPU, die beim Erreichen des Haltepunktes vorlagen, in die Schattenregister gerettet.

Bei Neuaufruf des Go-Kommandos wird die jeweils letzte Break-Adresse als Go-Start-Adresse vorgegeben. Jeweils nach Abschluß mit [SP] *) werden dann der Reihe nach wieder alle vorgewählten Break-Adressen angezeigt, ehe das Programm bis zur nächsten Break-Adresse abgearbeitet wird.

*) Um die Übersicht zu behalten, sollte man nach dem Aufruf des Go-Kommandos immer alle Break-Adressen mit [SP] aufrufen. So hat man stets die Anfangsadressen der bereits untersuchten Programmteile und die der noch zu untersuchenden vor Augen.

Fehlermeldungen:

Wenn im Anwenderprogramm ein RST4-Befehl (E7H) gefunden wird, erfolgt der Registerausdruck mit der Überschrift *** BREAKPOINT ERROR ***.

MAT 85 Name:

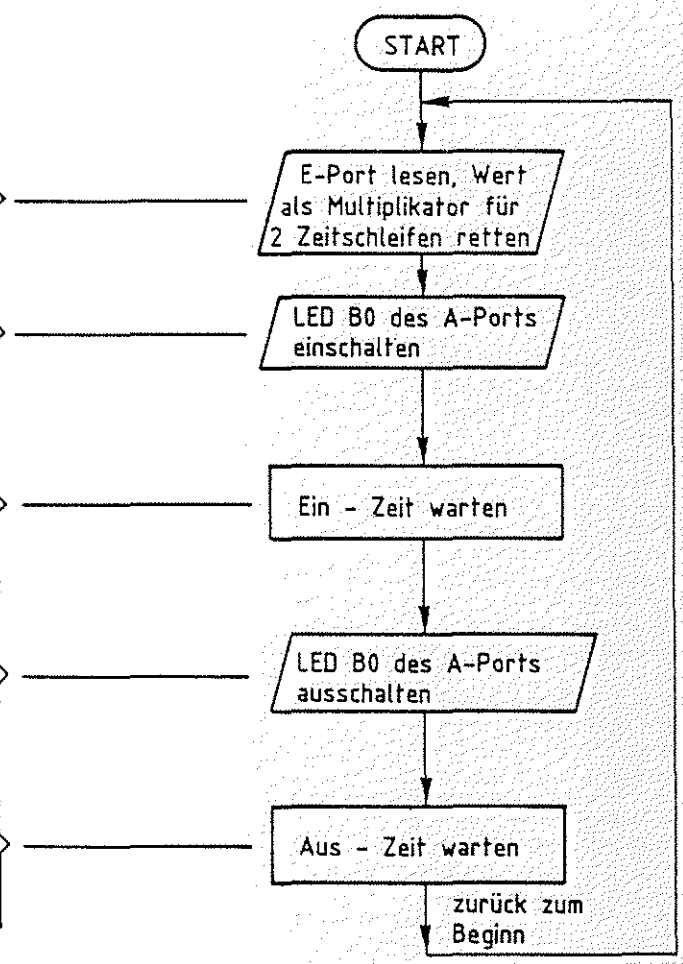
Fehlersuche mit B und N. Datum:

B4

Anhand des folgenden Programms wird der Einsatz des B-Kommandos in Verbindung mit dem N-Kommando demonstriert.
Das Programm hat die Aufgabe, die obere LED B0 des Ausgabe-Ports (Adresse 13H) periodisch blinken zu lassen. Die Periodendauer soll mit der Schalterstellung der Schalter des Eingabe-Ports (Adresse 12H) veränderbar sein (Rechteckgenerator).

1. Geben Sie folgendes (zunächst fehlerhafte) Programm ab Adresse F800 in den Speicher ein.

Adresse	Op- Code	Operand
F800	IN	12
F802	MOV	B,A
F803	MOV	D,A
F804	MVI	A,01
F806	OUT	13
F808	MVI	C,0FF
F80A	DCR	C
F80B	JNZ	0F80A
F80E	DCR	B
F80F	JNZ	0F808
F812	MVI	A,00
F814	OUT	13
F816	MVI	C,0FF
F818	DCR	C
F819	JNZ	0F816
F81C	DCR	D
F81D	JNZ	0F818
F820	JMP	0F800



2. Stellen Sie alle Schalter des E-Ports auf L-Pegel (LED's aus) und starten Sie das Programm bei ausgeschalteten "Breakpoints".

   Wirkung: LED B0 des Ausgabeports leuchtet kurz auf und erlischt wieder. Ein Rücksprung ins Betriebsprogramm erfolgt nicht.

MAT 85 Name:

Fehlersuche mit B und N. Datum:

B5

Das Programm arbeitet in einer Schleife, die es wegen eines logischen Fehlers nicht mehr verlassen kann.

Anweisung	Wirkung/Kommentar
RESET betätigen	Rückkehr ins Betriebsprogramm
BREAKPOINT einschalten	In der Folge wird untersucht, ob die einzelnen Programmteile, die auch durch die Blöcke im Flußdiagramm dargestellt sind, funktionieren.
Mit R alle Register auf 0; Am E-Port 08 einstellen; Go aufrufen; START-ADR ⟶ F800 BREAK-ADR1 ⟶ F808 BREAK-ADR2...4 ⟶ 0000	} Startbedingungen Programmbeginn Beginn der Zeitschleife für die Ein-Zeit. Es wird nur mit einem Haltepunkt gearbeitet (übersichtlich). *** BREAKPOINT *** PC OP ADR.Feld A B D 01 08 08 F808 MVI C,FF LED B0 leuchtet Das am E-Port mit den Schaltern eingestellte Datum 08 wurde in die Register B und D geladen. Der Wert 01 wurde in den Akku geladen und zum A-Port ausgegeben.
Go aufrufen; START-ADR ⟶ F800 BREAK-ADR1 ⟶ F812 BREAD-ADR2...4 ⟶ 0000	Programmbeginn Beginn des Programmteils "LED B0 ausschalten". *** BREAKPOINT *** PC OP ADR.FELD A B C D 01 00 00 08 F812 MVI A,00 LED B0 leuchtet Die Ein-Zeit-Schleife wurde in Echtzeit durchlaufen (N-Kommando wäre hierzu nicht geeignet). Durch die DCR-Befehle in der Schleife enthalten die Register B und C den Wert Null.

MAT 85 Name:

Fehlersuche mit B und N. Datum:

B6

Anweisung	Wirkung/Kommentar
Go aufrufen START-ADR ⟶ F812 BREAK-ADR1 ⟶ F816 BREAK-ADR2...4 ⟶ 0000	ab hier soll weitergeprüft werden. Beginn der Zeitschleife für die Aus-Zeit. *** BREAKPOINT *** PC OP ADR.FELD A B D 00 00 08 F816 MVI C,FF LED B0 dunkel Der Wert 00 wurde in den Akku geladen und zum A-Port ausgegeben.
Go aufrufen START-ADR ⟶ F814 BREAK-ADR1 ⟶ F820 BREAK-ADR2...4 ⟶ 0000	Die Startadresse muß außerhalb der zu prüfenden Zeitschleife liegen. Läge sie innerhalb, würde die Programmabarbeitung nach jedem Dekrementieren angehalten. Ende außerhalb der Schleife. Haltepunkt F820 wird nicht erreicht. Das Programm "hängt fest". Der Fehler liegt in der Aus-Zeit-Schleife.
N aufrufen; START-ADR ⟶ F816 STEPS ⟶ 10	PC OP ADR.FELD C D F816 MVI C,FF FF 08 F818 DCR C FE 08 F819 JNZ F816 FE 08 F816 MVI C,FF FF 08 F818 DCR C FE 08 F819 JNZ F816 FE 08 F816 MVI C,FF FF 08 ⋮ Hier liegt der Fehler! Der Sprung nach dem Dekrementieren von Register C (wenn C≠0) führt wieder nach F816. Dort aber wird Register C wieder mit FF geladen. Diese Schleife wird nicht mehr verlassen.

MAT 85

Name:

Fehlersuche mit B und N.

Datum:

B7

Anweisung	Wirkung/Kommentar
JNZ F816 ändern in JNZ F818 Go aufrufen; START-ADR ⟶ F814 BREAK-ADR1 ⟶ F820 BREAK-ADR2...4 ⟶ 0000	mit dem M- oder A-Kommando Beginn vor die Aus-Zeit-Schleife gelegt. Ende der Zeitschleife *** BREAKPOINT *** PC OP ADR.FELD A B C D 00 00 00 00 F820 JMP F800 Die Schleife ist in Echtzeit durchlaufen worden. Die Inhalte der Register C und D sind auf Null herabgezählt worden.
BREAKPOINT ausschalten Programm mit G starten Verändern Sie die Stellung der Schalter des E-Ports.	Das Programm arbeitet richtig. Die Blinkfrequenz verändert sich.
Bauen Sie eigene Fehler in das Programm ein und versuchen Sie, diese nach der gezeigten Methode zu finden. Setzen Sie später auch mehrere Breakpoints. Setzen Sie auch Breakpoints innerhalb einer Zeitschleife.	

Mit dem Trace-Interval-Kommando wird ...

- die Protokollierung der Registerinhalte für einen gewünschten Programmabschnitt eines Anwenderprogramms ein- oder ausgeschaltet und
- Start- und Stop-Adresse dieses Programmsabschnittes eingegeben.

Aufruf und Handhabung:

KMD > TRACE INTERVAL	T [CR] oder T [SP] eintippen "RACE INTERVAL" wird ergänzt
EIN/AUS =X Y	X = Vorgabe; Ein: Y = E [CR] oder E [SP] Aus: Y = A [CR] oder A [SP] Unverändert: Y = [CR] oder [SP]
START-ADR =X1X1 Y1Y1	X1X1 = Vorgabe; Neu: Y1Y1 [CR] oder [SP] Alt: [CR] oder [SP]
STOP -ADR =X2X2 Y2Y2	X2X2 = Vorgabe; Neu: Y2Y2 [CR] oder [SP] Alt: [CR] oder [SP]
	Y1Y1 und Y2Y2 müssen auf ein Befehlsbyte zeigen

Zur Kommando-Ausführung:

Nach dem Einschalten des "Trace-Interval" und Aufruf des G-Kommandos wird das Anwenderprogramm unter der Kontrolle des Tracers Befehl für Befehl durchlaufen (verlängerte Bearbeitungszeit!).

- Solange sich der Tracer außerhalb des angegebenen Trace-Intervalls aufhält, wird kein Bildschirmausdruck sichtbar.
- Gerät der Tracer in das angegebene Intervall (einschließlich Start- und Stop-Adresse), so werden nach Ausführung jedes Befehls in diesem Bereich alle Registerinhalte ausgedruckt.
- Breakpoints können ohne Einschränkung im gesamten Adreßbereich weiter verwendet werden.
- Im Trace-Betrieb werden der Halt-Befehl (76H) und illegale OP-Codes erkannt und führen zum Programmabbruch (ohne Trace-Betriebsart wird ein Halt-Befehl ohne eine Meldung ausgeführt).

MAT 85

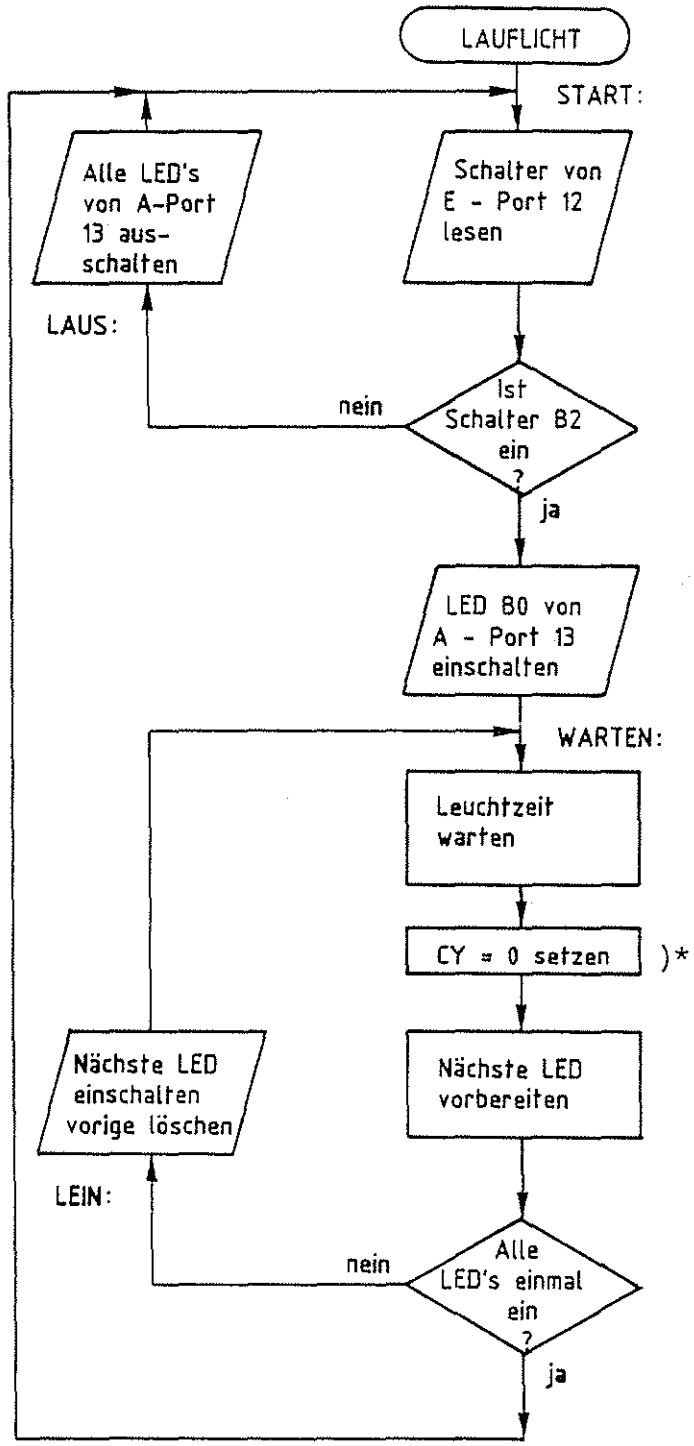
Name:

Übung T-Kommando

Datum:

T2

Laden Sie das folgende "Lauflicht-Programm"
ab Adresse F800 in den Speicher.
Überprüfen Sie Ihre Eingabe mit dem D-Kommando.



Adresse	Label	Op-Code
F800	START:	IN 12
F802		ANI 04
F804		JZ LAUS
F807		NOP
F808		MVI A,01
F80A		OUT 13
F80C	WARTEN:	MVI B,33
F80E	ZA:	MVI C,99
F810	ZI:	DCR C
F811		JNZ ZI
F814		DCR B
F815		JNZ ZA
F818		ANA A
F819		RAL
F81A		CPI 00
F81C		JNZ LEIN
F81F		JMP START
F822	LAUS:	MVI A,00
F824		OUT 13
F826		JMP START
F829	LEIN:	OUT 13
F82B		JMP WARTEN
F82E	END	

MAT 85 Name:

Übung T-Kommando Datum:

T3

)* Der RAL-Befehl schiebt den Akku-Inhalt (hier 01) über das Carry-Bit um eine Bitstelle nach links. Sollte das Carry-Bit zufällig 1 sein, würde diese 1 in Bit 0 des Akkus geschoben, was nicht erwünscht ist. Daher wird hier vor RAL mit ANA A das Carry-Bit auf 0 gesetzt.

Anweisung	Wirkung/Kommentar
Schalter B2 vom E-Port einschalten; Programm mit G bei F800 starten	Die CPU arbeitet das Anwender-Programm in "Echtzeit" ab . Die LED's am A-Port werden in schneller Folge von oben nach unten ein- und ausgeschaltet. Es entsteht der Eindruck eines "Laufenden Lichtes".
RESET betätigen Mit T "Trace Interval" einschalten; START-ADR ⟶ F800 STOP -ADR ⟶ F80A Schalter B2 ⟶ Aus Mit R die Register A,B, C und alle Flags auf 0 setzen; Mit G bei F800 starten;	Die CPU kehrt vom Anwender-Programm zum Betriebsprogramm zurück. Das Betriebsprogramm erwartet ein neues Kommando. Trace Interval liegt damit zwischen F800 und F80A. PC LABEL OP ADR.FELD A ..Z.. B C F800 START: IN 12 00 0 00 00 F802 ANI 04 00 1 00 00 F804 JZ LAUS 00 1 00 00 F822 LAUS: MVI A,00 ├──── Kopfzeile ────┤ Text wie oben, außer Zeile F800 ├──── Kopfzeile ────┤ Text wie vorher ==> SPACE Alle Programmteile, die im vorgegebenen Trace Intervall (Fenster) liegen, werden ausgeführt und ausgedruckt. Weil SB2 auf "AUS" steht, erfolgt ein Sprung zum Programmteil "LAUS". Da dieser Teil außerhalb des vorgegebenen "Fensters" liegt, wird die Zeile F822 nicht vollständig ausgedruckt.

MAT 85

Name:

Übung T-Kommando

Datum:

T4

Anweisung	Wirkung/Kommentar
	Am Ende dieses Programmteils (bei Adr. F826) erfolgt der Rücksprung nach "START". Da der Tracer nun wieder innerhalb des "Fensters" arbeitet, werden die darin liegenden Programmteile erneut ausgedruckt. "==>SPACE" signalisiert, daß die weitere Programmbearbeitung durch den Tracer erst nach Betätigung der SPACE-Taste erfolgt. Bei dieser Meldung kann die Programmbearbeitung aber auch durch Betätigen von [ESC] abgebrochen werden.
Schalter B2 ⟶ Ein [SP] betätigen	PC LABEL OP ADR.Feld A ..Z.. B C F800 START: IN 12 04 1 00 00 F802 ANI 04 04 0 00 00 F804 JZ LAUS 04 0 00 00 F807 NOP 04 0 00 00 F808 MVI A,01 01 0 00 00 F80A OUT 13 01 0 00 00 F80C WARTEN: MVI B,33 – (Die LED's werden der Reihe nach ein- und nach ca. 11s ausgeschaltet. Nach ca. 90s erscheint:) Kopfzeile Protokollanfang wie oben = = > SPACE Weil SB2 eingeschaltet wurde, wird der Sprung zum Programmteil "LAUS" ignoriert (siehe Z-Flag) und bei F807 weitergemacht. Nach Bearbeitung der Zeile F80A verläßt der Tracer das "Fenster" und tritt in die Schleife ein, die mit "WARTEN:" beginnt. Innerhalb dieser Schleife müssen erst alle LED's einmal ein- bzw. ausgeschaltet worden sein, ehe der Tracer wieder in den vorgegebenen "Fensterbereich" eintritt (Sprung bei F81F). Da er die Befehle nicht in Echtzeit bearbeitet, erfolgt eine erneute Protokollierung erst nach ca. 90s. (Diese Zeit hängt vom Inhalt der Register B und C ab und vom Traceprogramm).

MAT 85 Name:

Übung T-Kommando Datum:

T5

Anweisung	Wirkung/Kommentar
Legen Sie folgende Intervall-Adressen fest: START-ADR ⟶ F819 STOP -ADR ⟶ F81F	Das Protokoll zeigt Ihnen, unter welchen Bedingungen die Sprünge nach "LEIN" und "START" erfolgen.
Legen Sie folgende Intervall-Adressen fest: START-ADR ⟶ F80C STOP -ADR ⟶ F818	Das Protokoll zeigt Ihnen, wie die Warteschleife abgearbeitet wird. Wenn die Bearbeitungszeit verkürzt werden soll, müssen die Register B und C mit kleineren Zahlenwerten geladen werden. Hierzu müssen Sie das Programm entsprechend ändern.
Fügen Sie anstelle des NOP-Befehls einen HLT-Befehl ein. Starten Sie das Programm einmal bei ausgeschaltetem und bei eingeschaltetem Tracer.	Tracer aus: Der HLT-Befehl wird ausgeführt, ohne daß eine Meldung erfolgt. (Dabei sendet die CPU die nächst höhere Adresse F808 auf den Adreßbus und schaltet die Daten- und Steuersignale in den hochohmigen Zustand. Rückkehr ins Betriebsprogramm ist nur durch RESET möglich. Tracer ein: Es erfolgt Programmabbruch und die Meldung *** HALT ODER ILLEGALER OPCODE ***
Schalten Sie nach den Experimenten den Tracer aus.	

I-Kommando

Mit dem In-Kommando (nicht zu verwechseln mit dem 8085-Befehl IN) können Daten von Eingabe-Baugruppen gelesen werden.

Aufruf und Handhabung:

```
KMD > IN

 PORT-NR=X1 Y1



 DATEN  =X2 Y2
```

I [CR] oder I [SP] eintippen
"N" wird ergänzt

X1 = Vorgabe; Vorgabe - 1: [-] eintippen
Vorgabe + 1: [+] eintippen
Neu: Y1 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X2 = Gelesener Wert (Hex)
Y2 = [CR] : Fertig, nächstes Kommando
Y2 = [SP] : Nochmal lesen
Y2 = [+/-] : Neue Port-Adresse

Übung:

Lesen Sie die Daten Ihres Eingabe-Ports.
Ändern Sie die Bitkombination mit Hilfe der Schalter und lesen Sie erneut.
Versuchen Sie auch Daten von Ports zu lesen, die gar nicht existieren.

O-Kommando

O

Mit dem Out-Kommando (nicht zu verwechseln mit dem 8085-Befehl OUT) lassen sich Daten zu Ausgabe-Baugruppen übertragen.

Aufruf und Handhabung:

```
KMD > OUT

 PORT-NR=X1 Y1



 DATEN  =X2 Y2
```

O [CR] oder O [SP] eintippen
"UT" wird ergänzt

X1 = Vorgabe; Vorgabe - 1: [-] eintippen
Vorgabe + 1: [+] eintippen
Neu: Y1 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X2 = Vorgabe; Neu: Y2 [CR] (Fertig)
Y2 [SP] (Nochmal)
Vorgabe: [SP]

Y2 = [CR] : Fertig (auch [ESC])
Y2 = [+/-] : Neue Port-Adresse
Y2 = [SP] : Nochmal schreiben

Übung:
Schreiben Sie der Reihe nach die Datenbytes
01 - 10 - 00 - FF - 55 - AA in Ihr Ausgabe-Port.
Versuchen Sie auch Daten in ein Port zu schreiben, das gar nicht existiert.

S-Kommando

S

Mit dem Save-Kommando können Programme und Daten über einen Kassetten-Recorder auf Magnetband gespeichert werden. Die Verwendung des S-Kommandos erfordert zusätzlich die Baugruppe "Kassetten-Interface BFZ/MFA 4.4".

Aufruf und Handhabung:

```
KMD> SAVE

 START-ADR =X1X1 Y1Y1

 STOP -ADR =X2X2 Y2Y2

 BAND EINSCHALTEN, DANN SPACE
 (Kommando-Ausführung)
KMD>_
```

S [CR] oder S [SP] eintippen
"AVE" wird ergänzt

X1X1 = Vorgabe; Neu: Y1Y1 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X2X2 = Vorgabe; Neu: Y2Y2 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

Reihenfolge beachten!
Vorspann am Anfang
der Bänder kann nicht
beschrieben werden!

Zur Kommando-Ausführung:

- Die Startadresse (X1X1 oder Y1Y1) wird für den späteren Ladevorgang mit auf das Band übertragen.
- Zur Erkennung von Lesefehlern werden mit den Daten, die als ASCII-Zeichen übertragen werden, auch Prüfsummenbytes übertragen.
- Soll die Kommando-Ausführung abgebrochen werden, ist die RESET-Taste auf der CPU-Baugruppe zu betätigen.

L-Kommando

L

Mit dem Load-Kommando werden Daten vom Kassetten-Recorder in den Speicher zurückgelesen. Sollen die Daten nicht in den beim Save-Kommando angegebenen Speicherbereich übertragen werden, so kann eine neue Startadresse angegeben werden.

Aufruf und Handhabung:

```
KMD> LOAD TAPE

 START-ADR =YYYY




 SPACE, DANN BAND EINSCHALTEN
 (Kommando-Ausführung)
KMD>_
```

L [CR] oder L [SP] eintippen
"OAD TAPE wird ergänzt

YYYY = 0: Startadresse, die auf der Kassette gespeichert ist, wird genommen.
Neu: YYYY [CR] oder YYYY [SP] eintippen.

Reihenfolge beachten!

Zur Kommando-Ausführung:

- Die Daten vom Band werden eingelesen, eine Kontrollsummenbildung findet dabei statt.
- Ausdruck bei fehlerfreiem Empfang: [READY]
- Ausdruck bei fehlerbehaftetem Empfang: [CHECKSUM ERROR]
- Ausdruck, wenn ein empfangenes Zeichen nicht den Hex-Zeichen 0 bis F im ASCII-Code entspricht (z.B. 0≙30, F≙46): [*** NICHT HEX = XX], wobei die empfangene Bitkombination XX (z.B. 0A≙LF) ausgegeben wird.
- Abbruch des L-Kommandos ist nur mit RESET möglich.

Anhang

1. Anschluß einer Datensichtstation

1.1. Bedingungen für eine fehlerfreie Datenübertragung

Um eine fehlerfreie Datenübertragung zwischen dem MFA-Mikrocomputer und der Datensichtstation zu gewährleisten, sind folgende Punkte zu beachten:

1. Arbeiten beide Geräte mit den gleichen Strom- oder Spannungspegeln?
2. Übertragen beide Geräte die gleiche Anzahl Daten-Bits?
3. Ist in beiden Geräten die Paritätsprüfung ein- oder ausgeschaltet?
4. Wird auf gerade oder ungerade Parität geprüft?
5. Stimmt die Anzahl der Stop-Bits in beiden Geräten überein?
6. Mit welcher Baudrate sendet die Datensichtstation?

Informationen hierzu für den MFA-Mikrocomputer:

Zu 1.: Es sind möglich eine 20-mA-Stromschnittstelle und eine V-24-Spannungsschnittstelle. Beide müssen zusätzlich verdrahtet werden (siehe Anschlußpläne auf den folgenden Seiten).
Pegel der 20-mA-Stromschnittstelle:
log. 1 = unterbrochener Stromkreis
log. 0 = Strom von 20 mA
Pegel der V-24-Spannungsschnittstelle:
log. 1 = -12 V
log. 0 = +12 V

Zu 2.-5.: Der MFA-Mikrocomputer sendet 1 Start-, 7 Daten-, 1 Paritäts- und 2 Stop-Bits aus. Die gleiche Bitfolge kann er auch empfangen. Die Paritätsbits werden nicht überprüft.

Zu 6.: Nach Einschalten des MFA-Mikrocomputers muß die Space-Taste der Datensichtstation betätigt werden. Aus dem empfangenen Datenwort bestimmt das Betriebsprogramm dann die Übertragungsgeschwindigkeit der Datensichtstation und paßt an diese die eigene Baudrate an.

1.2. Anschlußplan

Die meisten Datensichtgeräte verwenden einen genormten Buchsenstecker mit 25 Anschlüssen (ITT-Cannon DB-25S oder Harting 09 67 025 2704). Die zugehörigen Stiftstecker haben die Bezeichnung DB-25P (Cannon) und 09 67 025 2604 (Harting). Die Steckerbelegung ist genormt, die wichtigsten Anschlüsse zeigt Bild 7.

Pin-Nr.	Funktion	Bemerkungen
1	Gehäusemasse	—
2	Transmit Data (TxD)	Sendeleitung
3	Receive Data (RxD)	Empfangsleitung
4	Request to Send (RTS)	Sendeaufforderung
5	Clear to Send (CTS)	Sender betriebsbereit log. 1 = Sender freigeben
6	Data Set Ready (DSR)	Empfänger bereit?
7	Signalmasse	—
8	Data Carrier detect (DCD)	Signal vorhanden
20	Data Terminal Ready (DTR)	Empfänger bereit

Bild 7: Anschlußbelegung V-24-Norm

In Bild 8 ist gezeigt, wie nun die Verbindungen zwischen einer Datensichtstation und dem MFA-Mikrocomputer herzustellen sind.

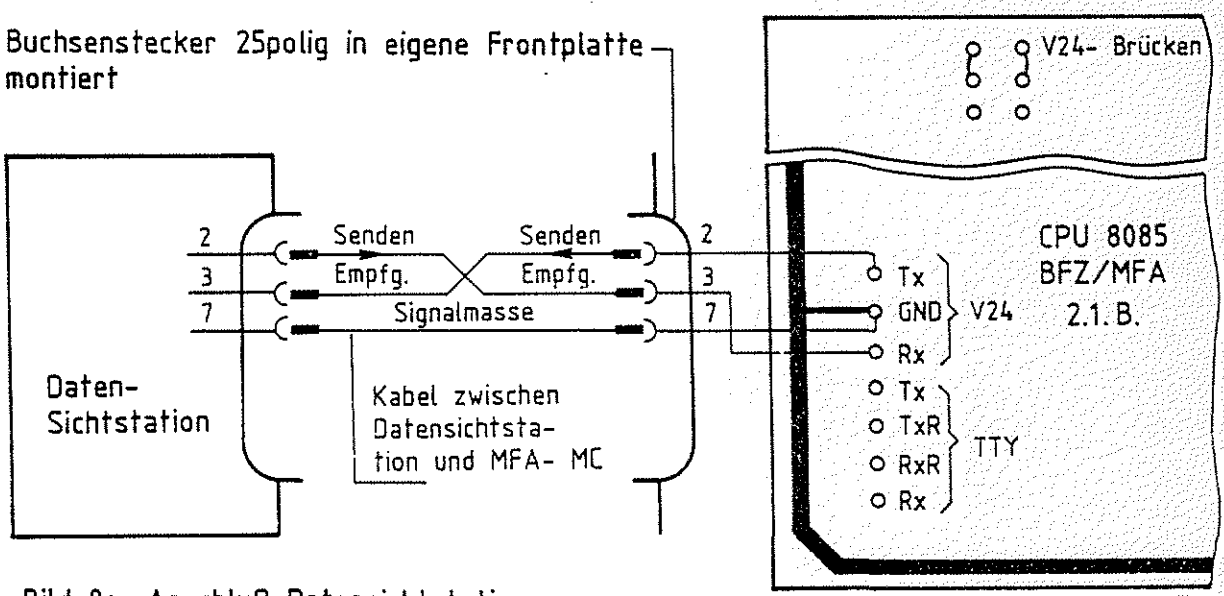


Bild 8: Anschluß Datensichtstation

2. Druckermodus, TTY-Betrieb

Das Betriebsprogramm unterscheidet je nach gemessener Übertragungsgeschwindigkeit zwischen einem Bildschirm- und einem Drucker-Modus. Der Drucker-Modus stellt sich immer dann ein, wenn die gemessene Übertragungsgeschwindigkeit gleich oder kleiner 300 Baud ist.

Im Gegensatz zum Bildschirm-Modus können bei Betrieb mit einer TTY falsch gedruckte Zeichen nicht gelöscht werden. Bei Betätigung der Taste [DEL] nach einem falsch eingegebenen Zeichen wird ein "/" (Slash-Schrägstrich) ausgedruckt und das falsch an den Mikrocomputer gesendete Zeichen gelöscht. Außerdem erfolgt bei längeren Protokollen kein Zwischenstop.

Bild 9 zeigt, wie ein Fernschreiber an den MFA-Mikrocomputer anzuschließen ist. Auch hier wird die Baudrate wieder vom Betriebsprogramm gemessen, wenn nach dem Einschalten der Geräte die [SP] -Taste betätigt wird.

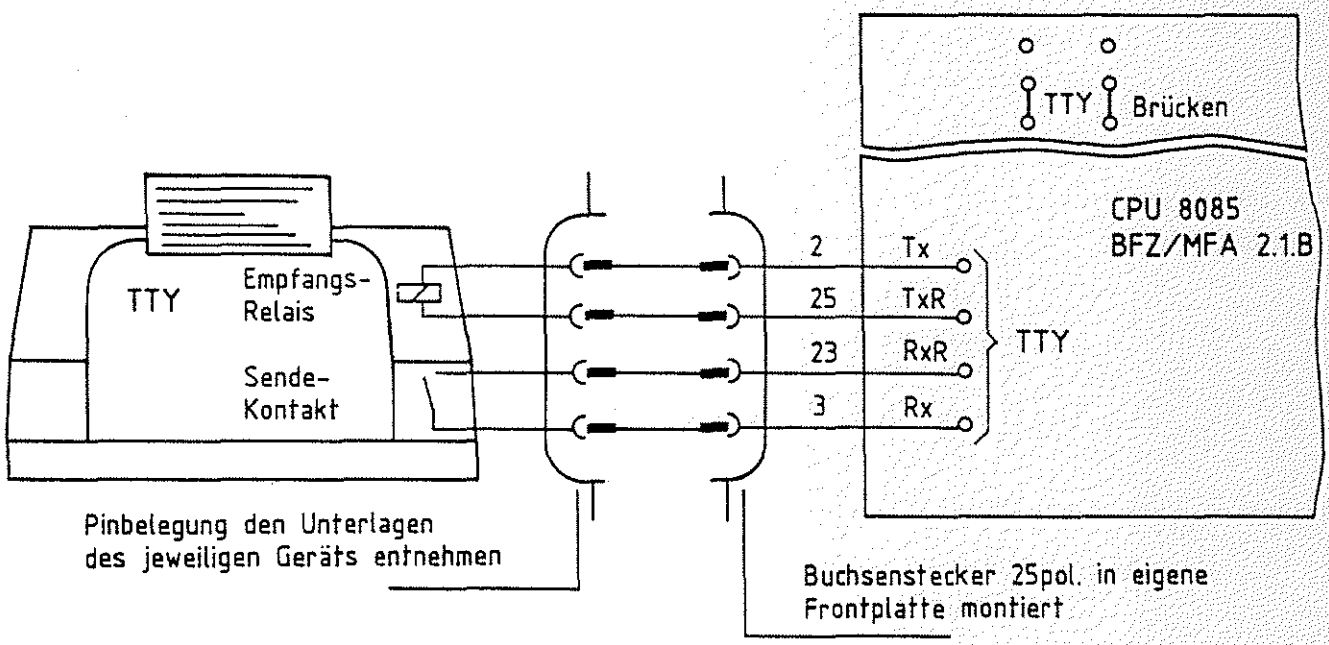


Bild 9: Anschluß TTY

3. Anschluß eines Matrix-Druckers

Matrix-Drucker arbeiten häufig mit paralleler Datenübertragung. Zum Anschluß solcher Drucker gibt es keine Norm. Der Druckerhersteller Centronics verwendete in seinen Druckern erstmals eine Steckerbelegung, die heute allgemeiner Standard ist und als "Centronics-Schnittstelle" bezeichnet wird. Für die parallele Daten-

übertragung zwischen MFA-Mikrocomputer und Drucker wird eine zusätzliche Baugruppe, die "Programmierbare parallele Schnittstelle BFZ/MFA 4.3." (Baugruppe mit Anschlußleitung) erforderlich. An diese Baugruppe wird der Drucker mit einem 25poligen Cannon-Stecker angeschlossen.
Das Betriebsprogramm enthält alle notwendigen Programmschritte, die zum Betrieb des Druckers erforderlich sind.

3.1. Betrieb des Matrix-Druckers

— Einschalten des Druckers
Drucker-Betriebsspannung einschalten!

```
KMD > GO

 START-ADR =XXXX 1E00
*** PRINTER ON ***
```

G [CR] oder G [SP] eintippen
"0" wird ergänzt
XXXX = Vorgabe; Neu: 1E00 [CR] oder [SP]
Meldung auf dem Bildschirm, daß der Drucker eingeschaltet ist. Der Drucker erzeugt zwei Zeilenvorschübe und bringt den Druckkopf in die Ausgangsposition.

— Eingabe eines kleinen Programms mit dem Assembler und Ausdruck.

```
KMD > ASSEMBLER
 START-ADR =0000 F800
F800 DB 12       START:  IN 12
F802 2F                  CMA
F803 D3 13               OUT 13
F805 C3 00F8             JMP START
F808             END
 *** RESTART ? (JA/NEIN) N
```

— Ausschalten des Druckers

```
KMD > GO

 START-ADR =1E09 1E03

*** PRINTER OFF ***
```

G [CR] oder G [SP] eintippen
"0" wird ergänzt
1E09 = Vorgabe; Neu: 1E03 [CR] oder [SP]

Meldung auf dem Bildschirm, daß der Drucker abgeschaltet ist. Auf dem Druckerpapier erscheint diese Meldung nicht!

— Fehlermeldung

```
*** PRINTER NOT READY ***
```

Diese Meldung erscheint auf dem Bildschirm, wenn bei Aufruf des Druckers mit 1E00 die Betriebsspannung nicht eingeschaltet wurde, oder wenn bei eingeschalteter Betriebsspannung kein Papier eingespannt ist.

4. ASCII-Code-Tabelle

Der ASCII-Code (American Standard Code for Information Interchange) ist ein in Amerika entwickelter Code für Fernschreiber, der heute allgemein für Datenübertragungseinrichtungen verwendet wird. Er setzt sich aus den Zeichengruppen Buchstaben, Ziffern, Sonderzeichen und Steuerzeichen zusammen.

Bild 10 zeigt eine Zuordnung der einzelnen Zeichen zu ihren hexadezimalen und binären Signaldarstellungen.

B6 B4 \ B3 B0	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	8 1000	9 1001	A 1010	B 1011	C 1100	D 1101	E 1110	F 1111
0 000	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1 001	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	VS
2 010	SPC	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3 011	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4 100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5 101	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6 110	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7 111	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Bild 10: ASCII - Code - Tabelle

Das Code-Wort in hexadezimaler und binärer Darstellung ergibt sich aus Zeilen- und Spaltenwerten wie in folgenden Beispielen gezeigt:

Zeichen	Code in Hex.-Darstellung	Code in Binärdarst.
A	41 (Zeile 4, Spalte 1)	100 0001
e	65 (Zeile 6, Spalte 5)	110 0101
}	7D (Zeile 7, Spalte D)	111 1101
?	3F (Zeile 3, Spalte F)	011 1111
5	35 (Zeile 3, Spalte 5)	011 0101

Die Bedeutung der Steuerzeichen (00H-20H und 7FH) zeigt die folgende Tabelle in Bild 11.

Code Hex	Code ASCII	Bedeutung	Steuerung des Cursors im Video-Interface BFZ/MFA 8.2.
00	NUL	Null, Nichts	
01	SOH	Kopfzellenbeginn	
02	STX	Textanfangszeichen	
03	ETX	Textendezeichen	
04	EOT	Ende der Übertragung	
05	ENQ	Aufforderung zur Datenübertragung	
06	ACK	Positive Rückmeldung	
07	BEL	Klingelzeichen	
08	BS	Rückwärtsschritt, (←)	um 1 Stelle nach links
09	HT	Horizontal Tabulator, (→)	um 1 Stelle nach rechts
0A	LF	Zeilenvorschub, (↓)	um 1 Stelle nach unten
0B	VT	Vertikal Tabulator, (↑)	um 1 Stelle nach oben
0C	FF	Seitenvorschub	Bildschirm löschen und zurück nach links oben (Zeitdauer dazu ca. 150ms)
0D	CR	Wagenrücklauf	zurück zum Zeilenanfang und Löschen des Zeilenendes.
0E	SO	Dauerumschaltzeichen	
0F	SI	Rückschaltungszeichen	
10	DLE	Datenübertragungsumschaltung	
11	DC1	Gerätesteuerzeichen 1	
12	DC2	Gerätesteuerzeichen 2	
13	DC3	Gerätesteuerzeichen 3	
14	DC4	Gerätesteuerzeichen 4	
15	NAK	Negative Rückmeldung	
16	SYN	Synchronisierung	
17	ETB	Ende des Datenübertragungsblocks	
18	CAN	Ungültig	
19	EM	Ende der Aufzeichnung	
1A	SUB	Substitution	
1B	ESC	Umschaltung	um 1 Zeile nach unten ohne Löschen der letzten Zeile
1C	FS	Hauptgruppentrennzeichen	zurück nach links oben
1D	GS	Gruppentrennzeichen	zurück zum Zeilenanfang
1E	RS	Untergruppentrennzeichen	
1F	US	Teilgruppentrennzeichen	
20	SP	Leerzeichen	
7F	DEL	Löschen des vorhergehenden Zeichens	

Bild 11: Bedeutung der Steuerzeichen

5. Die Tastatur Cherry G80-0177

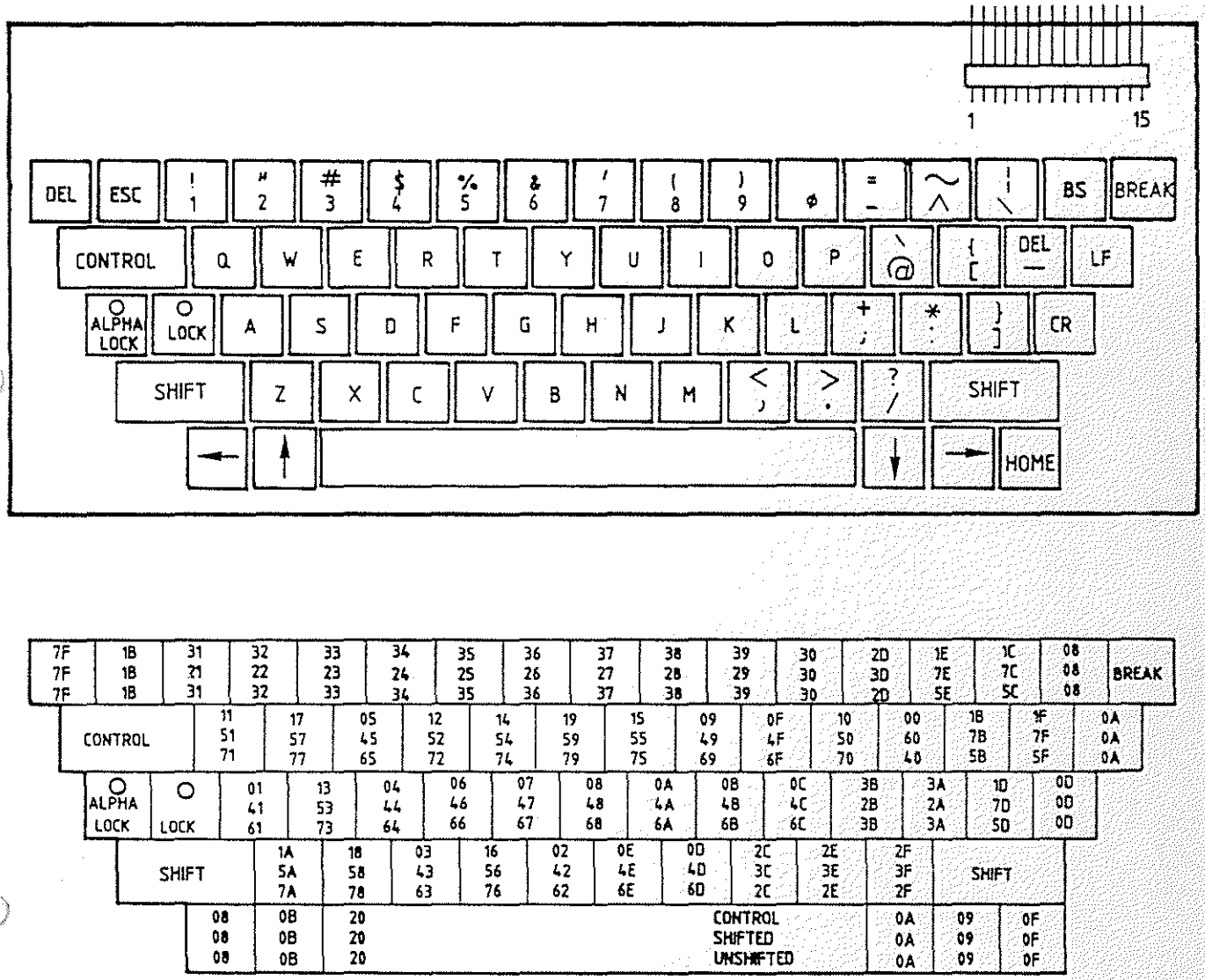


Bild 12: Beschriftung der Tasten und hexadezimale Verschlüsselung der Tastenfunktion

6. Häufig verwendete Symbole für Flußdiagramme (DIN 66001)

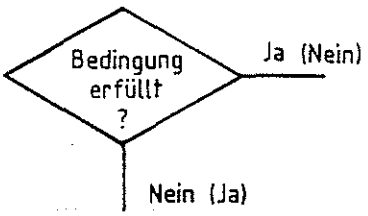
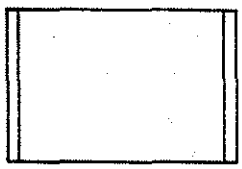
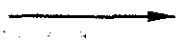
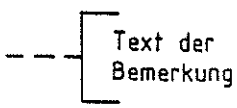
Symbole	Bedeutung
	Grenzstelle (Beginn, Ende ...)
	allgemeine Operation
	Verzweigung
	Ein-/Ausgabe- Operation
	Unterprogramm
	Flußlinie (Richtung der Abarbeitung)
	Zusammenführung
	Bemerkung
	Nahtstelle

Bild 13: Symbole für Flußdiagramme

7. Unterprogramme des Betriebsprogramms

Die in folgender Tabelle aufgeführten Unterprogramme aus dem Betriebsprogramm können Sie in eigenen Programmen verwenden. Wenn Sie die in der Tabelle angegebenen Namen der Unterprogramme in Ihren Programmen mitbenutzen wollen, müssen Sie diese Namen mit Hilfe der EQU-Anweisung vorher den zugehörigen Adressen zuweisen (siehe Beispiele).

Unterpr. Name	Eingangs-Adresse	Veränd. Register	Funktion des Unterprogramms
KMD	0040		Rücksprung in die Kommandoroutine des Monitorprogramms und Ausdruck von KMD>.
RCHAR	0043	A	Liest ein Zeichen von der Tastatur ein. Der ASCII-Code des Zeichens steht im Akku. Bei [ESC] Rückkehr in die Kommandoroutine und Klingeln.
WCHARI	0055		Gibt 1 Zeichen, das nach dem CALL-Befehl im Speicher steht, auf Bildschirm und Drucker (wenn Ein) aus. Das auszugebende Zeichen muß mit der DB-Anweisung in den Speicher geschrieben werden. Beispiel: CALL 0055 DB 'A' ; A wird ausgegeben
WAHEX	0058		Gibt den Akku-Inhalt (8Bit) als zwei Hexadezimalziffern auf dem Bildschirm/Drucker aus.
WHLHEX	005B	H,L	Gibt den HL-Registerinhalt (16Bit) als vier Hexadezimalziffern auf dem Bildschirm/Drucker aus.
WABIN	005E	A	Gibt den Akku-Inhalt (8Bit) als Binärzahl am Bildschirm/Drucker aus. Beispiel: MVI A,23 ; 23 Hexadezimal CALL 005E ; wird als 00100011 ausgegeben
WADEZ	0061	A	Gibt den Akku-Inhalt (8Bit) als Dezimalzahl auf dem Bildschirm/Drucker aus. MVI A,23 ;23 Hexadezimal wird CALL 0061 ;als 35 ausgegeben

Fortsetzung Unterprogramme des Betriebsprogramms

Unterpr. Name	Eingangs-Adresse	Veränd. Register	Funktion des Unterprogramms
WAFOR	0064	A,C	Gibt den Akku-Inhalt (8Bit) in einem der zu wählenden Formate ASCII-Binär-Dezimal-Hex auf dem Bildschirm/Drucker aus. Das Format wird durch den Inhalt des Registers C wie folgt gewählt: 0 → ASCII-Zeichen 1 → Binärzahl 2 → Dezimalzahl 3 → Hexadezimalzahl Beispiel: MVI A,23 ; 23 Hexadezimal wird MVI C,1 ; als 00100011 ausge- CALL 0064 ; geben
WBLANK	0067		Gibt ein Leerzeichen (Blank) auf dem Bildschirm/Drucker aus.
WBUFI	006D		Gibt den hinter dem CALL-Befehl stehenden Text auf dem Bildschirm aus. Der Text muß mit der DB-Anweisung in den Speicher (Textpuffer) geladen werden. Am Ende des Textes muß als Enderkennung eine 0 stehen. Beispiel: CALL 006D DB ' DIES IST EINE UEBUNG', 0 Gibt den Text DIES IST EINE UEBUNG aus.
WCRLFI	0073		Gibt einen Wagenrücklauf (CR), eine Neue Zeile (LF, Line Feed) und Text in diese neue Zeile aus. Der Text muß wie bei "WBUFI" vorher eingegeben werden.
HADR	08DF		Liest eine 16-Bit-Adresse (4 Hex-Stellen) von der Tastatur ein und speichert sie im Doppelregister H ab. Dabei gelangt der höherwertige Teil der Adresse ins H-Register und der niederwertige Teil ins L-Register. Die Eingabe der Adresse muß mit [CR] oder [SP] abgeschlossen werden.
BSTIME	0895	A,D,E	Zeitverzögerung von 0,24 s. Damit die Inhalte der Register A,D und E vor Aufruf des Unterprogramms gerettet und nachher

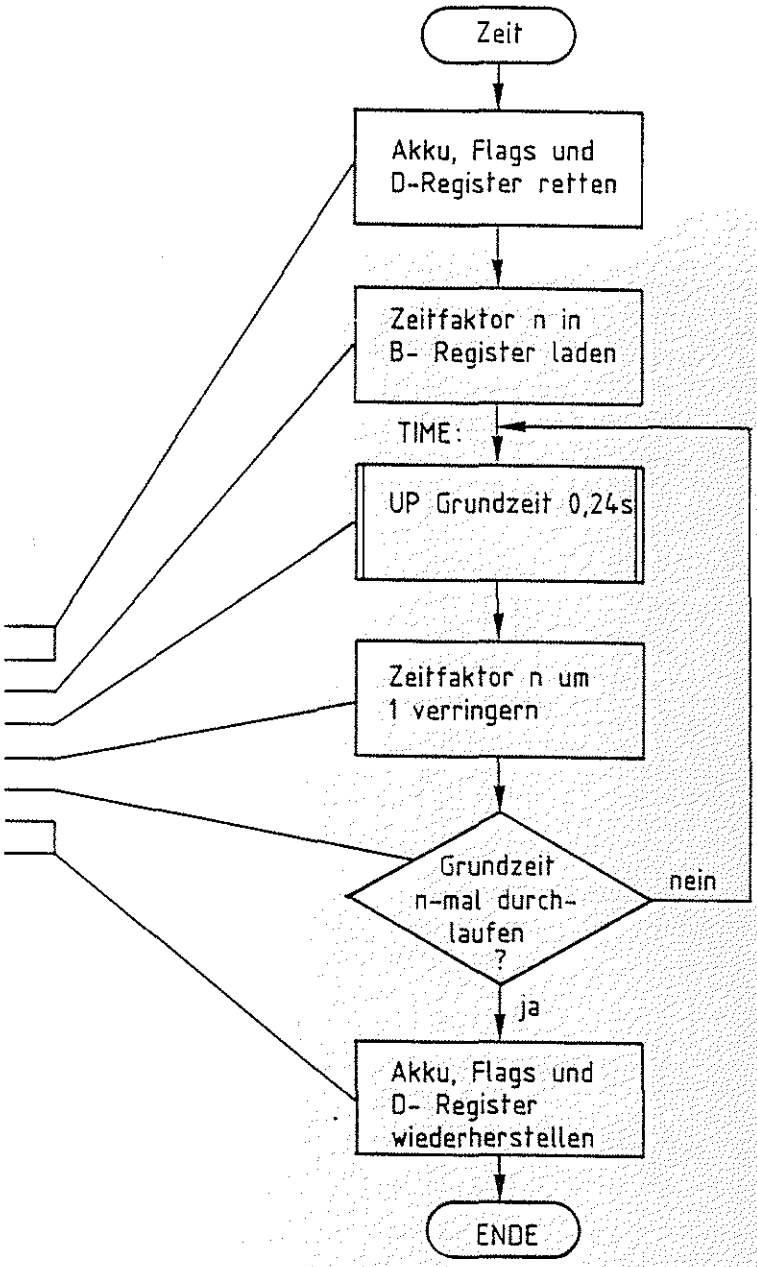
Fortsetzung Unterprogramme des Betriebsprogramms

Unterpr. Name	Eingangs-Adresse	Veränd. Register	Funktion des Unterprogramms
BSTIME			wiederhergestellt werden, muß die folgende Befehlsfolge eingehalten werden: PUSH PSW ;Registerinhalte A,D,E PUSH D ;retten CALL 0895 ;Zeitverzögerung POP D ;Registerinhalte wiederher- POP PSW ;stellen
CMP2	0EA8	A	Vergleicht die Inhalte der Register DE mit denen der Register HL. Wenn (HL)>(DE) ist, wird das Carry-Flag auf 1 gesetzt, sonst auf 0. Die zu vergleichenden Inhalte müssen vor Aufruf des Unterprogramms in die Doppelregister D und H geladen werden. Beispiel: LXI D, Zahl 1 LXI H, Zahl 2 CALL CMP2
SUB2	1039	A,HL, DE	Subtrahiert die 16-Bit-Zahl im Doppelregister D von der 16-Bit-Zahl im Doppelregister H. Das Ergebnis steht dann im Doppelregister H. [(HL) = (HL) - (DE)]
WBUF	0BA1		Gibt Text aus einem Textpuffer aus, dessen Anfangsadresse durch den Inhalt des HL-Registers adressiert ist. Der Text wird mit der DB-Anweisung ab dieser Adresse geladen, das Textende muß mit 0 gekennzeichnet sein. Nach der Ausgabe des Textes zeigt das HL-Register auf die Adresse nach dem Endezeichen.

8. Beispiele für den Gebrauch von Unterprogrammen aus dem Betriebsprogramm

8.1. Verzögerungszeit n x 0,24 s
(Hier wurde n = 10 (0AH)
gewählt)

```
KMD > ASSEMBLER
 START-ADR =F800
F800                LC
F800                BSTIME EQU 0895
F800 F5                 PUSH PSW
F801 D5                 PUSH D
F802 06 0A              MVI  B,0A
F804 CD 9508 TIME:      CALL BSTIME
F807 05                 DCR  B
F808 C2 04F8            JNZ  TIME
F80B D1                 POP  D
F80C F1                 POP  PSW
```



Das Retten der Register ist nur nötig, wenn die Inhalte dieser Register vor Aufruf des Unterprogramms "BSTIME" Werte enthalten, die nach Abarbeitung des Unterprogramms erst im weiteren Programmverlauf benötigt werden.

8.2. Bildschirm löschen und Textausgabe

```
KMD > ASSEMBLER
 START-ADR =F807 F800
F800              ; BILDSCHIRM LOESCHEN UND
F800              ; CURSOR NACH LINKS OBEN
F800              ; AUSGABE EINES TEXTES
F800              LC
F800              FF EQU 0C  ; FF = SEITENVORSCHUB
F800              WCHARI EQU 55
F800              WCRLFI EQU 73
F800              BSTIME EQU 0895
F800              ;
F800 CD 5500              CALL WCHARI
F803 0C                   DB FF
F804 CD 9508              CALL BSTIME
F807 CD 7300              CALL WCRLFI
F80A 20444945             DB ' DIE WARTEZEIT "BSTIME" IST',0A,0D
F80E 20574152
F812 54455A45
F816 49542022
F81A 42535449
F81E 4D452220
F822 4953540A
F826 0D
F827 20455246             DB ' ERFORDERLICH ,WEIL DAS VIDEO-',0A,0D
F82B 4F524445
F82F 524C4943
F833 48202C57
F837 45494C20
F83B 44415320
F83F 56494445
F843 4F2D0A0D
F847 20494E54             DB ' INTERFACE CA. 150 MS ZUR LOE-',0A,0D
F84B 45524641
F84F 43452043
F853 412E2031
F857 3530204D
F85B 53205A55
F85F 52204C4F
F863 452D0A0D
F867 20534348             DB ' SCHUNG DES BILDSCHIRMS BRAUCHT',00
F86B 554E4720
F86F 44455320
F873 42494C44
F877 53434849
F87B 524D5320
F87F 42524155
F883 43485400
F887 CF                   RST 1
F888              END
```

Wagenrücklauf

Zeilenvorschub

Textende

Der ausgegebene Text:

```
DIE WARTEZEIT "BSTIME" IST
ERFORDERLICH ,WEIL DAS VIDEO-
INTERFACE CA. 150 MS ZUR LOE-
SCHUNG DES BILDSCHIRMS BRAUCHT
```

Wenn man Wagenrücklauf und Zeilenvorschub am Ende der jeweiligen DB-Anweisungen wegläßt, wird jeweils die ganze Bildschirmzeile vollgeschrieben.

8.3. Steuerung des Cursors per Programm

Zur Steuerung des Cursors auf dem Bildschirm muß man dem Video-Interface bestimmte Steuerzeichen senden, die dort als solche erkannt werden und unmittelbar zur Bewegung des Cursors in horizontaler oder vertikaler Richtung führen (siehe Bilder 11 u. 12).
Im folgenden Programmbeispiel soll dies demonstriert werden.
Aufgabe: In die Mitte des leeren Bildschirms soll ein Rechteck mit einer Seitenlänge von 30 Zeichen (-) und einer Höhe von 6 x Zeilenabstand (I) dargestellt werden.
In die Mitte des Rechtecks soll ein Text geschrieben werden; danach soll in die oberste Zeile des Bildschirms die Meldung zur Rückkehr ins Betriebsprogramm ausgegeben werden. Eine solche Rückkehr soll nur mit [ESC] möglich sein.

Bild 14 zeigt das zu programmierende Rechteck und seine Lage innerhalb des Bildschirms.

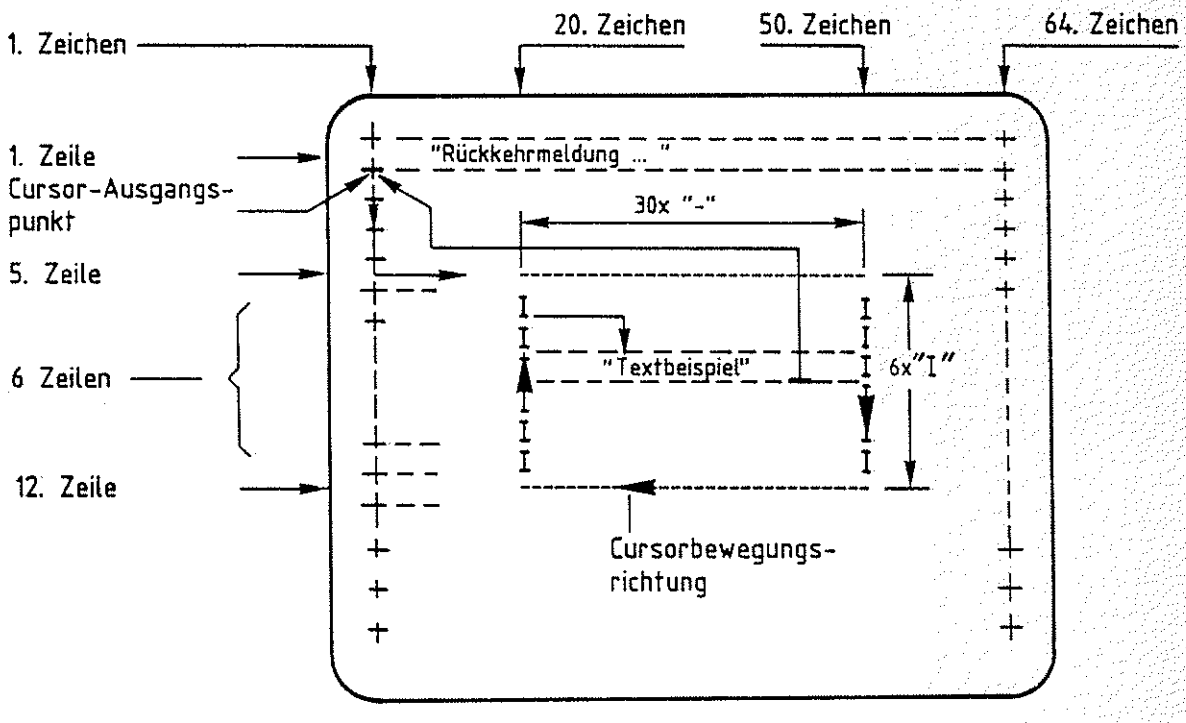


Bild 14: Einteilung des Bildschirms für das Programmierbeispiel

Bild 15 zeigt das Flußdiagramm zu dieser Aufgabe. Beachten Sie die Kommentare zu einzelnen Schritten und die Lösungen dazu im Programm-Listing.

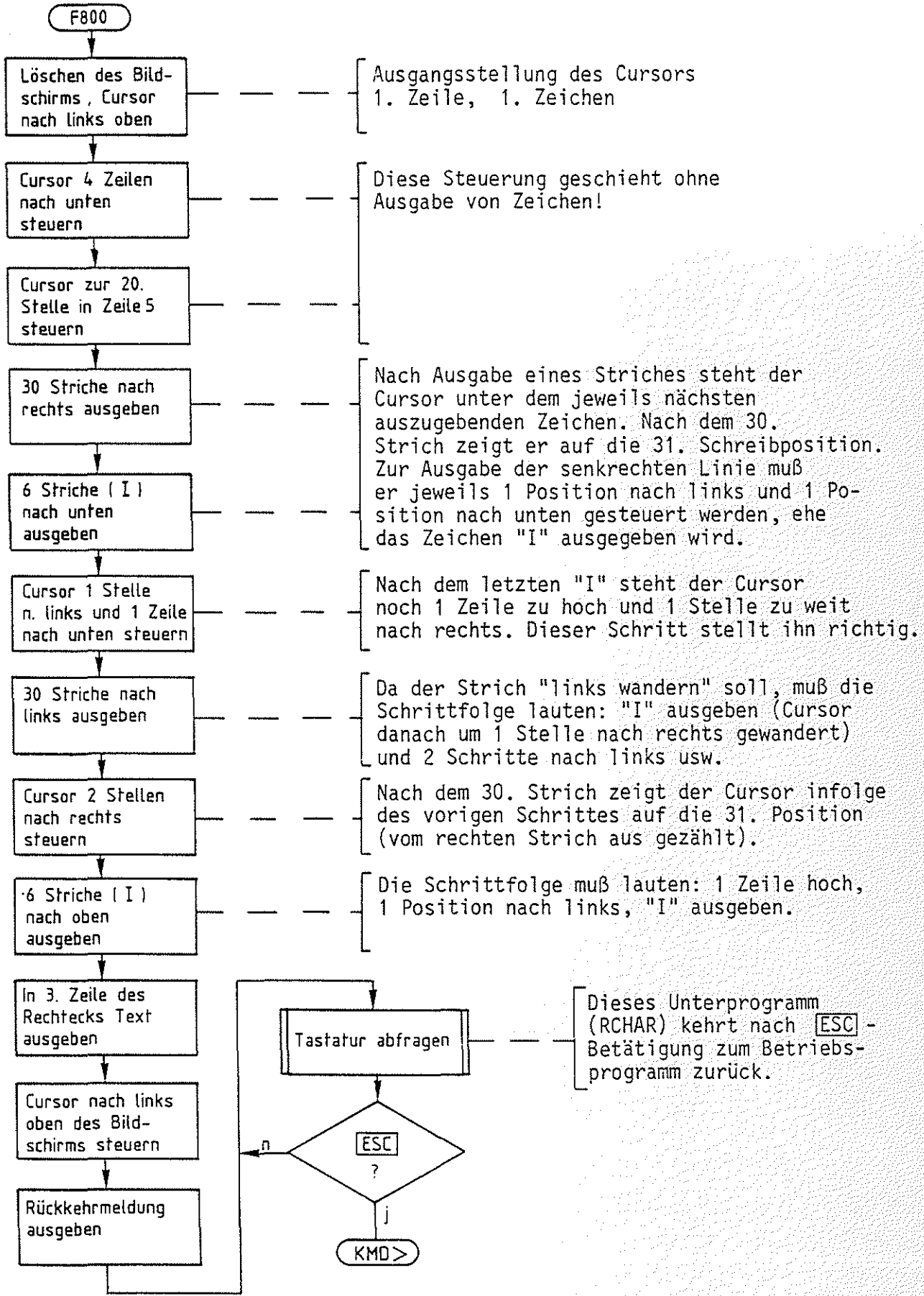


Bild 15: Flußdiagramm "Cursorsteuerung per Programm"

Anhang

Programm "Cursorsteuerung per Programm".

```
KMD > ASSEMBLER
 START-ADR =F800
F800                ; CURSOR - STEUERUNG
F800                ;
F800                LC
F800                WBUFI EQU 6D
F800                WCHARI EQU 55
F800                BSTIME EQU 0895
F800                RCHAR EQU 43
F800                ;
F800                ;
F800 CD 5500        CALL WCHARI
F803 0C             DB 0C
F804 CD 9508        CALL BSTIME
F807 0E 04          MVI  C,04
F809 CD 5500 LF:    CALL WCHARI
F80C 0A             DB 0A
F80D 0D             DCR  C
F80E C2 09F8        JNZ  LF
F811 0E 13          MVI  C,13
F813 CD 5500 HT:    CALL WCHARI
F816 20             DB 20
F817 0D             DCR  C
F818 C2 13F8        JNZ  HT
F81B 0E 1E          MVI  C,1E
F81D CD 5500 HZ:    CALL WCHARI
F820 2D             DB 2D
F821 0D             DCR  C
F822 C2 1DF8        JNZ  HZ
F825 0E 06          MVI  C,06
F827 CD 6D00 VU:    CALL WBUFI
F82A 080A4900       DB 08,0A,'I',00
F82E 0D             DCR  C
F82F C2 27F8        JNZ  VU
F832 CD 6D00        CALL WBUFI
F835 080A00         DB 08,0A,00
F838 00             NOP
F839 0E 1E          MVI  C,1E
F83B CD 6D00 HL:    CALL WBUFI
F83E 2D080800       DB '-',08,08,00
F842 0D             DCR  C
F843 C2 3BF8        JNZ  HL
F846 CD 6D00        CALL WBUFI
F849 090900         DB 09,09,00
```

Definition der verwendeten Unterprogramme

Zeilenabstand

Bildschirm löschen und Cursor nach links oben

4 Zeilen nach unten

zur 20. Schreibposition nach rechts

30 Striche (Minuszeichen) nach rechts ausgeben

6 Zeichen "I" nach unten ausgeben

1 Pos. nach links und 1 Zeile nach unten steuern.

NOP kann entfallen; 30 Striche nach links ausgeben

Cursor 2 Stellen nach rechts

Fortsetzung des Programms "Cursorsteuerung per Programm".

```
F84C 0E 06          MVI  C,06
F84E CD 6D00 VO:    CALL WBUFI
F851 0B084900       DB 0B,08,'I',00
F855 0D             DCR  C
F856 C2 4EF8        JNZ  VO
```

6 Striche nach oben ausgeben

```
F859 CD 6D00        CALL WBUFI
F85C 0A0A2020   DB 0A,0A,20,20,20,2A,20,41,4C
F860 202A2041
F864 4C
F865 4C455320   DB 4C,45,53,20,50,41,4C,45,54
F869 50414C45
F86D 54
F86E 5449203F   DB 54,49,20,3F,20,2A,00
F872 202A00
```

Text innerhalb des Rechtecks

```
F875 CD 5500    CALL WCHARI
F878 1C         DB 1C
```

Cursor nach oben links auf Bildschirm

```
F879 CD 6D00    CALL WBUFI
F87C 4D495420   DB 'MIT ECS KOENNEN SIE NACH KMD > ZURUECK',00
F880 45534320
F884 4B4F454E
F888 4E454E20
F88C 53494520
F890 4E414348
F894 204B4D44
F898 203E205A
F89C 55525545
F8A0 434B00
```

Text

```
F8A3 CD 4300    ENDE:CALL RCHAR
F8A6 C3 A3F8    JMP ENDE
F8A9            END
 *** RESTART ? (Ja/NEIN)
```

Ende mit ESC

Anhang, 8085-Befehlsliste

Befehls-gruppe (Bedeutung der Spalten)	Mnemonik	Maschinen Code	Taktzyklen	Bytes	Masch. Zyklus	Funktion des Befehls	Veränderte Flags N	Z	P	C	H
Sprünge	JMP m	C 3	10	3	3	(PC) ← m	X	X	X	X	X
	PCHL	E 9	6	1	1	(PC) ← (H) (L)	X	X	X	X	X
	JC m	D A	10/7	3	3/2	(C)= 1	X	X	X	X	X
	JNC m	D 2	10/7	3	3/2	(C)= 0 Wenn Bedingung erfüllt	X	X	X	X	X
	JZ m	C A	10/7	3	3/2	(Z)= 1 (PC) ← m	X	X	X	X	X
	JNZ m	C 2	10/7	3	3/2	(Z)= 0	X	X	X	X	X
	JP m	F 2	10/7	3	3/2	(N)= 0	X	X	X	X	X
	JM m	F A	10/7	3	3/2	(N)= 1 Wenn Bedingung nicht erfüllt	X	X	X	X	X
	JPE m	E A	10/7	3	3/2	(P)= 1 (PC) ← (PC)+ 3	X	X	X	X	X
	JPO m	E 2	10/7	3	3/2	(P)= 0	X	X	X	X	X
Unterprogramm-aufrufe	CALL m	C D	18	3	5	((SP)- 1) ((SP)- 2) ← (PC)+ 3, (PC) ← m (SP) ← (SP)- 2	X	X	X	X	X
	RST n		12	1	3	((SP)- 1) ((SP)- 2) ← (PC)+ 1, (PC) ← n x 8 (SP) ← (SP)- 2 wobei 0 ≦ n ≦ 7	X	X	X	X	X
	CC m	D C	18/9	3	5/2	(C)= 1	X	X	X	X	X
	CNC m	D 4	18/9	3	5/2	(C)= 0 Wenn Bedingung erfüllt	X	X	X	X	X
	CZ m	C C	18/9	3	5/2	(Z)= 1 ((SP)- 1) ((SP)- 2) ← (PC)+ 3	X	X	X	X	X
	CNZ m	C 4	18/9	3	5/2	(Z)= 0 (PC) ← m, (SP) ← (SP)- 2	X	X	X	X	X
	CP m	F 4	18/9	3	5/2	(N)= 0	X	X	X	X	X
	CM m	F C	18/9	3	5/2	(N)= 1 Wenn Bedingung nicht erfüllt	X	X	X	X	X
	CPE m	E C	18/9	3	5/2	(P)= 1 (PC) ← (PC)+ 3	X	X	X	X	X
	CPO m	E 4	18/9	3	5/2	(P)= 0	X	X	X	X	X
Unterprogramm-Rücksprünge	RET	C 9	10	1	3	(PC) ← ((SP)+ 1) ((SP)), (SP) ← (SP) + 2	X	X	X	X	X
	RC	D 8	12/6	1	3/1	(C)= 1	X	X	X	X	X
	RNC	D 0	12/6	1	3/1	(C)= 0 Wenn Bedingung erfüllt	X	X	X	X	X
	RZ	C 8	12/6	1	3/1	(Z)= 1 (PC) ← ((SP)+ 1) ((SP)), (SP) ← (SP)+ 2	X	X	X	X	X
	RNZ	C 0	12/6	1	3/1	(Z)= 0	X	X	X	X	X
	RP	F 0	12/6	1	3/1	(N)= 0	X	X	X	X	X
	RM	F 8	12/6	1	3/1	(N)= 1 Wenn Bedingung nicht erfüllt	X	X	X	X	X
	RPE	E 8	12/6	1	3/1	(P)= 1 (PC) ← (PC)+ 1	X	X	X	X	X
	RPO	E 0	12/6	1	3/1	(P)= 0	X	X	X	X	X
Ein-Ausgabe	IN n	D B	10	2	3	(A) ← (Eing. Puffer) ← (Eingang Daten von Gerät n)	X	X	X	X	X
	OUT n	D 3	10	2	3	(Ausgabe Gerät n) ← (A)	X	X	X	X	X
Unterbrech. Steuerung	EI	F B	4	1	1	(INTE) ← 1	X	X	X	X	X
	DI	F 3	4	1	1	(INTE) ← 0	X	X	X	X	X
Stapel-steuerung	PUSH PSW	F 5	12	1	3	((SP)- 1) ← (A), ((SP)- 2) ← (F), (SP) ← (SP)- 2	X	X	X	X	X
	PUSH B	C 5	12	1	3	((SP)- 1) ← (B), ((SP)- 2) ← (C), (SP) ← (SP)- 2	X	X	X	X	X
	PUSH D	D 5	12	1	3	((SP)- 1) ← (D), ((SP)- 2) ← (E), (SP) ← (SP)- 2	X	X	X	X	X
	PUSH H	E 5	12	1	3	((SP)- 1) ← (H), ((SP)- 2) ← (L), (SP) ← (SP)- 2	X	X	X	X	X
	POP PSW	F 1	10	1	3	(F) ← ((SP)), (A) ← ((SP)+ 1), (SP) ← (SP)+ 2	▮	▮	▮	▮	▮
	POP B	C 1	10	1	3	(C) ← ((SP)), (B) ← ((SP) + 1), (SP) ← (SP)+ 2	X	X	X	X	X
	POP D	D 1	10	1	3	(E) ← ((SP)), (D) ← ((SP)+ 1), (SP) ← (SP)+ 2	X	X	X	X	X
	POP H	E 1	10	1	3	(L) ← ((SP)), (H) ← ((SP)+ 1), (SP) ← (SP)+ 2	X	X	X	X	X
Sonstige	HLT	7 6	5	1	1	(PC) ← (PC)+ 1	X	X	X	X	X
	NOP	0 0	4	1	1	(PC) ← (PC)+ 1	X	X	X	X	X
8085 Befehle	RIM	2 0	4	1	1	Liest Unterbrechungsmaske und seriellen Eingang in Akku	X	X	X	X	X
	SIM	3 0	4	1	1	Setzt Unterbrechungsmaske und seriellen Ausgang	X	X	X	X	X

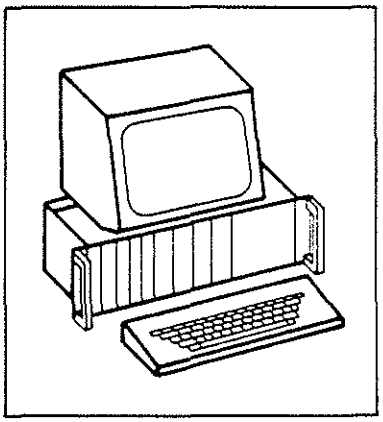
Symbol	Erklärung
r	Register (A, B, C, D, E, H, L, M)
m	Zwei- Byte Daten
n	Ein- Byte Daten
⟨B2⟩, ⟨B3⟩	2. bzw. 3. Byte des Befehls
F	8- Bit- Daten-Wort aus N, Z, X, H, X, P, X, C
PC	Programm- Zähler
SP	Stapel- Zeiger
←	Datum in gezeichneter Richtung transportiert

Symbol	Erklärung
( )	Inhalt eines Registers oder Speichers
∨, ⊻	Inklusiv ODER, Exklusiv ODER
∧	Log. UND
—	1er Komplement
X	Flaginhalt bleibt unverändert
▮	Flaginhalt wird gesetzt oder rückgesetzt
M	Inhalt der durch H und L adressierten Speicherzeile

Anhang, 8085-Befehlsliste

Befehlsgruppe (Bedeutung der Spalten)	Mnemonik	Maschinen Code	Taktzyklen	Bytes	Masch. Zyklus	Funktion des Befehls	Veränderte Flags N	Z	P	C	H
Daten-Transport	MOV r1, r2		4	1	1	(r1) ← (r2)	X	X	X	X	X
	MOV M, r		7	1	2	(M) ← (r)	X	X	X	X	X
	MOV r, M		7	1	2	(r) ← (M)	X	X	X	X	X
	MVI r, n		7	2	2	(r) ← n	X	X	X	X	X
	MVI M, n	3 6	10	2	3	(M) ← n	X	X	X	X	X
	LXI B, m	0 1	10	3	3	(C) ← <B2> (B) ← <B3> m= B3 B2	X	X	X	X	X
	LXI D, m	1 1	10	3	3	(E) ← <B2> (D) ← <B3> m= B3 B2	X	X	X	X	X
	LXI H, m	2 1	10	3	3	(L) ← <B2> (H) ← <B3> m= B3 B2	X	X	X	X	X
	LXI SP, m	3 1	10	3	3	(SP) ← m	X	X	X	X	X
	SPHL	F 9	6	1	1	(SP) ← (H) (L)	X	X	X	X	X
	STAX B	0 2	7	1	2	((B) (C)) ← (A)	X	X	X	X	X
	STAX D	1 2	7	1	2	((D) (E)) ← (A)	X	X	X	X	X
	LDAX B	0 A	7	1	2	(A) ← ((B) (C))	X	X	X	X	X
	LDAX D	1 A	7	1	2	(A) ← ((D) (E))	X	X	X	X	X
	STA m	3 2	13	3	4	(m) ← (A)	X	X	X	X	X
	LDA m	3 A	13	3	4	(A) ← (m)	X	X	X	X	X
	SHLD m	2 2	16	3	5	(m) ← (L) (m+1) ← (H)	X	X	X	X	X
	LHLD m	2 A	16	3	5	(L) ← (m) (H) ← (m+1)	X	X	X	X	X
	XCHG	E B	4	1	1	(H) (L) ↔ (D) (E)	X	X	X	X	X
	XTHL	E 3	16	1	5	(H) (L) ↔ ((SP)+1) ((SP))	X	X	X	X	X
Arithmetik, Logik Vergleich	ADD r		4	1	1	(A) ← (A)+(r)	▮	▮	▮	▮	▮
	ADD M	8 6	7	1	2	(A) ← (A)+(M)	▮	▮	▮	▮	▮
	ADI n	C 6	7	2	2	(A) ← (A)+n	▮	▮	▮	▮	▮
	ADC r		4	1	1	(A) ← (A)+(r)+(C)	▮	▮	▮	▮	▮
	ADC M	8 E	7	1	2	(A) ← (A)+(M)+(C) C= Carry	▮	▮	▮	▮	▮
	ACI n	C E	7	2	2	(A) ← (A)+n+(C)	▮	▮	▮	▮	▮
	DAD B	0 9	10	1	3	(H) (L) ← (H) (L)+(B) (C)	X	X	X	▮	X
	DAD D	1 9	10	1	3	(H) (L) ← (H) (L)+(D) (E)	X	X	X	▮	X
	DAD H	2 9	10	1	3	(H) (L) ← (H) (L)+(H) (L)	X	X	X	▮	X
	DAD SP	3 9	10	1	3	(H) (L) ← (H) (L)+(SP)	X	X	X	▮	X
	SUB r		4	1	1	(A) ← (A)-(r)	▮	▮	▮	▮	▮
	SUB M	9 6	7	1	2	(A) ← (A)-(M)	▮	▮	▮	▮	▮
	SUI n	D 6	7	2	2	(A) ← (A)-n	▮	▮	▮	▮	▮
	SBB r		4	1	1	(A) ← (A)-(r)-(C)	▮	▮	▮	▮	▮
	SBB M	9 E	7	1	2	(A) ← (A)-(M)-(C) C= Carry	▮	▮	▮	▮	▮
	SBI n	D E	7	2	2	(A) ← (A)-n-(C)	▮	▮	▮	▮	▮
	ANA r		4	1	1	(A) ← (A)∧(r)	▮	▮	▮	0	1
	ANA M	A 6	7	1	2	(A) ← (A)∧(M)	▮	▮	▮	0	1
	ANI n	E 6	7	2	2	(A) ← (A)∧n	▮	▮	▮	0	1
	XRA r		4	1	1	(A) ← (A)⊻(r)	▮	▮	▮	0	0
	XRA M	A E	7	1	2	(A) ← (A)⊻(M)	▮	▮	▮	0	0
	XRI n	E E	7	2	2	(A) ← (A)⊻n	▮	▮	▮	0	0
	ORA r		4	1	1	(A) ← (A)∨(r)	▮	▮	▮	0	0
	ORA M	B 6	7	1	2	(A) ← (A)∨(M)	▮	▮	▮	0	0
	ORI n	F 6	7	2	2	(A) ← (A)∨n	▮	▮	▮	0	0
	CMP r		4	1	1	(A) – (r)	▮	▮	▮	▮	▮
	CMP M	B E	7	1	2	(A) – (M)	▮	▮	▮	▮	▮
	CPI n	F E	7	2	2	(A) – n	▮	▮	▮	▮	▮
Register Inkrement Dekrement	INR r		4	1	1	(r) ← (r)+1	▮	▮	▮	X	▮
	INR M	3 4	10	1	3	(M) ← (M)+1	▮	▮	▮	X	▮
	DCR r		4	1	1	(r) ← (r)-1	▮	▮	▮	X	▮
	DCR M	3 5	10	1	3	(M) ← (M)-1	▮	▮	▮	X	▮
	INX B	0 3	6	1	1	(B) (C) ← (B) (C)+1	X	X	X	X	X
	INX D	1 3	6	1	1	(D) (E) ← (D) (E)+1	X	X	X	X	X
	INX H	2 3	6	1	1	(H) (L) ← (H) (L)+1	X	X	X	X	X
	INX SP	3 3	6	1	1	(SP) ← (SP)+1	X	X	X	X	X
	DCX B	0 B	6	1	1	(B) (C) ← (B) (C)-1	X	X	X	X	X
	DCX D	1 B	6	1	1	(D) (E) ← (D) (E)-1	X	X	X	X	X
	DCX H	2 B	6	1	1	(H) (L) ← (H) (L)-1	X	X	X	X	X
	DCX SP	3 B	6	1	1	(SP) ← (SP)-1	X	X	X	X	X
Akku rotieren	RLC	0 7	4	1	1	Links schieben C ← A7 A6 ... A1 A0 ← A7	X	X	X	▮	X
	RRC	0 F	4	1	1	Rechts schieben C → A7 A6 ... A1 A0 → A0	X	X	X	▮	X
	RAL	1 7	4	1	1	Links rotieren C ← A7 A6 ... A1 A0 ← C	X	X	X	▮	X
	RAR	1 F	4	1	1	Rechts rotieren C → A7 A6 ... A1 A0 → C	X	X	X	▮	X
Akku beeinfl.	CMA	2 F	4	1	1	(A) ← (Ā)	X	X	X	X	X
	DAA	2 7	4	1	1	(A) in BCD wandeln	▮	▮	▮	▮	▮
Carry setzen	STC	3 7	4	1	1	(C) ← 1	X	X	X	1	X
	CMC	3 F	4	1	1	(C) ← (C̄)	X	X	X	▮	X

FACHPRAKTISCHE ÜBUNG MIKROCOMPUTER-TECHNIK



Softwarepaket SP 1

BFZ/MFA 7.2.

BfZ MFA

Diese Übung ist Bestandteil eines Mediensystems, das im Rahmen eines vom Bundesminister für Bildung und Wissenschaft, vom Bundesminister für Forschung und Technologie sowie der Bundesanstalt für Arbeit geförderten Modellversuches zum Einsatz der "Mikrocomputer-Technik in der Facharbeiterausbildung" vom BFZ-Essen e.V. entwickelt wurde.

Inhaltsverzeichnis

		Seite
1.	Einleitung	1
2.	Aufbau des Systems	3
3.	Schaltungsergänzungen	3
4.	Beschreibung der Programme	4
4.1.	Die Monitorerweiterung MAT85+	4
4.1.1.	Einleitung	4
4.1.2.	Kommando-Eingabe	5
4.1.3.	Reset-Betätigung	6
4.1.4.	Bildschirm-Modus, Drucker-Modus	6
4.1.5.	Bediener-Führung	9
4.1.6.	Kommando-Kurzbeschreibung	8
4.1.7.	Hinweise zur Beschreibung der Kommandos	9
4.1.8.	Kommando-Beschreibung	12
4.1.8.1.	Das HELP-Kommando	12
4.1.8.2.	Das BASIC-Kommando	13
4.1.8.3.	Das COPY-Kommando	14
4.1.8.4.	Das FIND-Kommando	18
4.1.8.5.	Das INSERT-Kommando	23
4.1.8.6.	Das PROMMER-Kommando	28
4.1.8.7.	Das RAM-TEST-Kommando	29
4.1.8.8.	Das SPS-Kommando	32
4.1.8.9.	Das VERIFY-Kommando	33
4.1.8.10.	Das WRITE CONSTANT-Kommando	36
4.2.	Der EPROM-Programmer	41
4.2.1.	Kommando-Eingabe	44
4.2.2.	Bildschirm-Modus, Drucker-Modus	44
4.2.3.	Bediener-Führung	44
4.2.3.1.	RAM-START/STOP-Adresse, EPROM-START-Adresse	45
4.2.4.	Kommando-Kurzbeschreibung	46
4.2.5.	Beschreibung der Kommandos	47
4.2.5.1.	Das HELP-Kommando	47
4.2.5.2.	Das COMPARE-Kommando	48
4.2.5.3.	Das PROGRAM-Kommando	51
4.2.5.4.	Das QUIT-Kommando	54
4.2.5.5.	Das READ-Kommando	55
4.2.5.6.	Das TEST-Kommando	57

Inhaltsverzeichnis

		Seite
4.3.	Das SPS-Programm	58
4.3.1.	Die SPS-Operanden	58
4.3.1.1.	Eingänge (E)	59
4.3.1.2.	Ausgänge (A)	59
4.3.1.3.	Merker (M)	60
4.3.1.4.	Hardware-Timer (T)	61
4.3.1.5.	Kennzahlen der Software-Timer (Z) und der Zähler (C)	62
4.3.1.6.	Software-Timer (Z)	62
4.3.1.7.	Zähler (C)	66
4.3.1.8.	Beispiele für die Kennzahlen	66
4.3.2.	Die Operationen	68
4.3.2.1.	Ein SPS-Ausdruck mit Bedingungs- und Zuweisungs-Teil	69
4.3.2.2.	Die GLEICH-Anweisung (=)	70
4.3.2.3.	Die UND-Verknüpfung (*)	70
4.3.2.4.	Die ODER-Verknüpfung (+)	70
4.3.2.5.	Die SETZ-Anweisung (=S)	71
4.3.2.6.	Die RÜCKSETZ-Anweisung (=R)	71
4.3.2.7.	Die LADE-Anweisung (=L)	72
4.3.2.8.	Die Negation (/)	72
4.3.2.9.	Das Syntax-Diagramm	73
4.3.3.	Grundzustand der Operanden	76
4.3.4.	Programm-Beispiele	76
4.3.4.1.	Blinklicht	76
4.3.4.2.	Zähler	77
4.3.5.	Aufruf des SPS-Programms	78
4.3.6.	Kommando-Eingabe	79
4.3.7.	Bildschirm-Modus, Drucker-Modus	79
4.3.8.	Bediener-Führung	79
4.3.9.	Beschreibung der Kommandos	80
4.3.9.1.	Das HELP-Kommando	80
4.3.9.2.	Das EDIT-Kommando	81
4.3.9.3.	Das GO-Kommando	90
4.3.9.4.	Das LIST-Kommando	92
4.3.9.5.	Das NEW-Kommando	93
4.3.9.6.	Das QUIT-Kommando	94
4.3.9.7.	Das READ-Kommando	95
4.3.9.8.	Das STEP-Kommando	96
4.3.9.9.	Das TRACE-Kommando	99
4.3.9.10.	Das WRITE-Kommando	102

Inhaltsverzeichnis

		Seite
4.4.	Das BFZ-Steuer-BASIC	103
4.4.1.	Zahlenbereich, Zahlendarstellung	103
4.4.2.	Zulässige Variablen-Namen	106
4.4.3.	Die Eingabe von Befehlen	107
4.4.4.	Aufruf des BFZ-Steuer-BASIC	108
4.4.5.	Der Befehlssatz des BFZ-Steuer-BASIC	110
4.4.5.1.	Der ABS-Befehl	111
4.4.5.2.	Der AND-Befehl	111
4.4.5.3.	Der CLS-Befehl	112
4.4.5.4.	Der DATA-Befehl	112
4.4.5.5.	Der DEC-Befehl	113
4.4.5.6.	Der END-Befehl	113
4.4.5.7.	Der FOR-Befehl	114
4.4.5.8.	Der FREE-Befehl	115
4.4.5.9.	Der GOSUB-Befehl	115
4.4.5.10.	Der GOTO-Befehl	116
4.4.5.11.	Der IF-Befehl	116
4.4.5.12.	Der INP-Befehl	117
4.4.5.13.	Der INPUT-Befehl	117
4.4.5.14.	Der LET-Befehl	119
4.4.5.15.	Der LIST-Befehl	119
4.4.5.16.	Der LOAD-Befehl	120
4.4.5.17.	Der LPOFF-Befehl	120
4.4.5.18.	Der LPON-Befehl	120
4.4.5.19.	Der NEW-Befehl	121
4.4.5.20.	Der NEXT-Befehl	121
4.4.5.21.	Der NOT-Befehl	121
4.4.5.22.	Der OUT-Befehl	122
4.4.5.23.	Der OR-Befehl	122
4.4.5.24.	Der PEEK-Befehl	123
4.4.5.25.	Der POKE-Befehl	123
4.4.5.26.	Der PRINT-Befehl	124
4.4.5.27.	Das QUIT-Kommando	126
4.4.5.28.	Das READ-Kommando	127
4.4.5.29.	Das REM-Kommando	128
4.4.5.30.	Das RESTORE-Kommando	128
4.4.5.31.	Das RETURN-Kommando	129
4.4.5.32.	Das RND-Kommando	129
4.4.5.33.	Das RUN-Kommando	129
4.4.5.34.	Das SAVE-Kommando	130
4.4.5.35.	Das STEP-Kommando	130
4.4.5.36.	Das STOFF-Kommando	130
4.4.5.37.	Das STON-Kommando	131
4.4.5.38.	Die STOP-Anweisung	132
4.4.5.39.	Der THEN-Befehl	132
4.4.5.40.	Der TO-Befehl	132
4.4.5.41.	Der TROFF-Befehl	133
4.4.5.42.	Der TRON-Befehl	133
4.4.5.43.	Der USR-Befehl	134
4.4.5.44.	Der WAIT-Befehl	135
4.4.6.	Die vier Grundrechenarten	139
4.4.7.	Das $-Symbol	140

Inhaltsverzeichnis

		Seite
5.	Anhang	141
5.1.	ROM-Bestückung	141
5.2.	Schaltungserweiterungen	143
5.2.1.	Programmabbruch durch Tastaturbetätigung	143
5.2.2.	Zählimpuls für SPS-Software-Timer	144
5.3.	Wichtige Unterprogramme	147
5.4.	Systemadressen	152
5.4.1.	SPS-Systemadressen	152
5.4.2.	BASIC-Systemadressen	152

System-Informationen

1. Einleitung

Das Software-Paket "SP 1" enthält die Monitorerweiterung MAT 85+, die das Betriebsprogramm MAT 85 um 10 Kommandos ergänzt. Außerdem enthält das Softwarepaket ein Programm für einen EPROM-Programmierer, ein Programm "SPS" (Speicherprogrammierbare Steuerung) und einen BASIC-Interpreter.

Das Softwarepaket ist in vier 2-KByte-EPROMs vom Typ 2716 gespeichert und belegt den Adreßraum ab Adresse 2000 bis 3FFF. Zum Betrieb des Softwarepaketes "SP1" ist das Betriebsprogramm MAT 85 erforderlich.

Informationen darüber, wie die EPROMs bei Verwendung der 8-K-Speicherkarte BFZ/MFA 3.1. bzw. der 16-K-Speicherkarte BFZ/MFA 3.2. in die Sockel eingesteckt werden müssen, finden Sie im Anhang.

Abhängig von der Verwendung der oben genannten Programmteile werden bestimmte RAM-Mindestbestückungen vorausgesetzt. Um die Monitorerweiterung MAT 85+ zu betreiben, muß mindestens der Adreßbereich F800 bis FFFF mit RAM bestückt sein. Für den EPROM-Programmierer gilt die gleiche Mindestbestückung. Will man das SPS-Programm verwenden, so muß zusätzlich der Adreßbereich E000 bis E7FF mit RAM bestückt sein. Für die Arbeit mit dem BASIC-Interpreter muß außerdem der Adreßbereich 6000 bis 67FF mit RAM bestückt sein. Die Speicherbelegung ist in Bild 1 dargestellt.

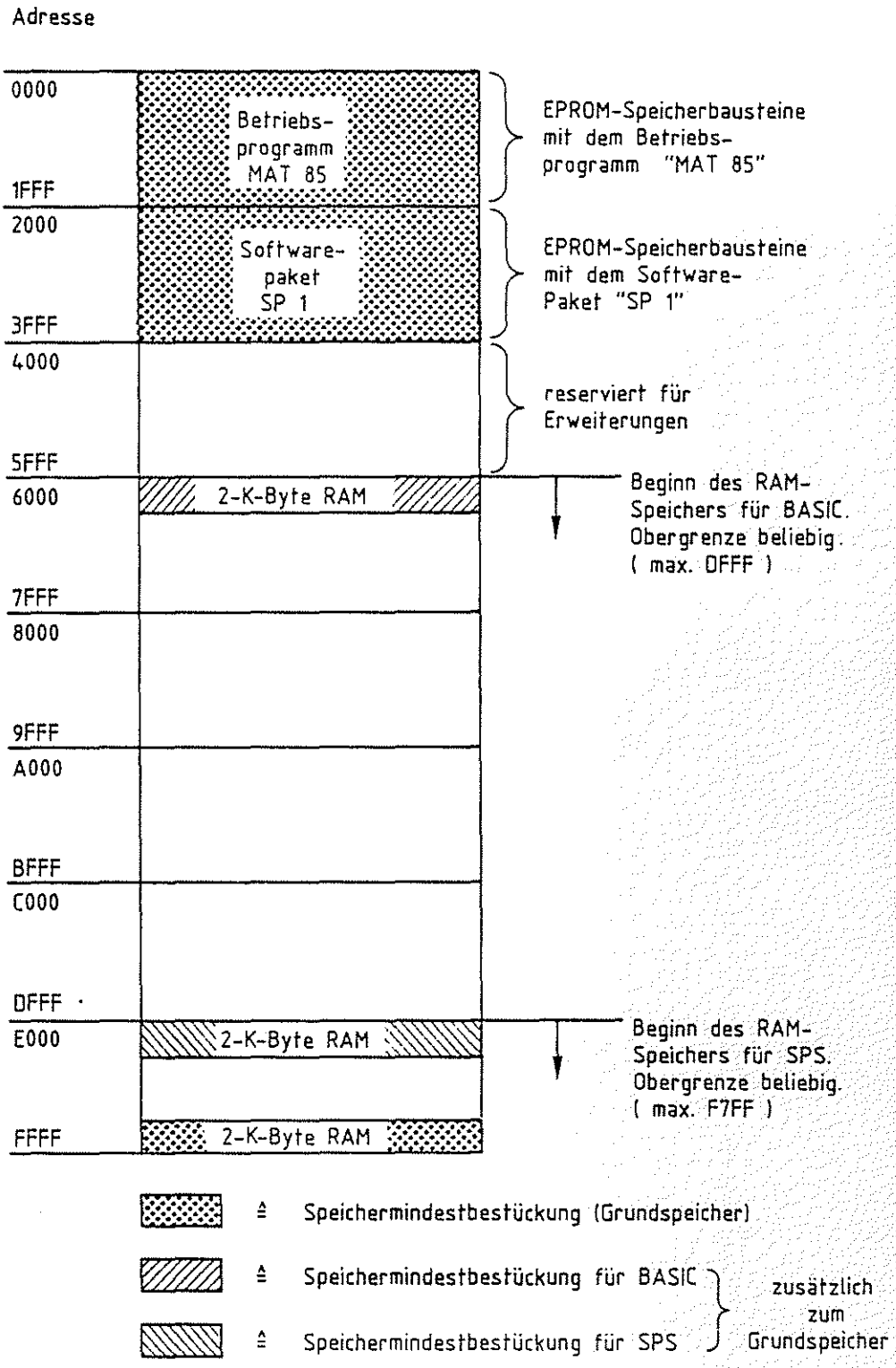


Bild 1: Speicherbelegung

2. Aufbau des Systems

Für den Aufbau des Systems benötigen Sie die folgenden Baugruppen:

1. Baugruppenträger mit Busverdrahtung BFZ/MFA 0.1.
2. Busabschluß BFZ/MFA 0.2.
3. Trafo-Einschub BFZ/MFA 1.1.
4. Spannungsregelung BFZ/MFA 1.2.
5. Prozessor 8085 BFZ/MFA 2.1.
6. 8-K-RAM/EPROM BFZ/MFA 3.1. bestückt mit MAT 85
7. 8-K-RAM/EPROM BFZ/MFA 3.1. bestückt mit SP1
8. 8-K-RAM/EPROM BFZ/MFA 3.1. bestückt mit mindestens 2-K-RAM
9. Video-Interface BFZ/MFA 8.2.
10. ASCII-Tastatur BFZ/MFA 8.1.
11. Monitor mit Cinch-Anschluß

Hinweis: Die Positionen 6 und 7 können durch eine 16-K-RAM/EPROM-Karte BFZ/MFA 3.2. mit entsprechender Bestückung ersetzt werden.

Je nach Anwendung sind noch folgende Ergänzungen notwendig:

Für den Einsatz des EPROM-Programmierers:
- EPROM-Programmierer BFZ/MFA 4.3.a

Für den Einsatz der SPS:
- ein zusätzlicher 2-K-RAM-Baustein (BFZ/MFA 3.1. aus obiger Auflistung mit mindestens 4-K-RAM entsprechend Bild 1 bestückt)
- je nach Bedarf:
  - 8-Bit-Parallel-Eingabe BFZ/MFA 4.2.
  - 8-Bit-Parallel-Ausgabe BFZ/MFA 4.1.
  - Zeitwerk (4fach) BFZ/MFA 4.3.c
  - Kassetten-Interface BFZ/MFA 4.4.a

Für den Einsatz des Steuer-Basics:
- 8-K-RAM/EPROM BFZ/MFA 3.1. bestückt mit mindestens 2-K-RAM
- je nach Bedarf:
  - Kassetten-Interface BFZ/MFA 4.4.a

3. Schaltungs-Ergänzungen

Um den SPS- und den BASIC-Interpreter voll nutzen zu können, sind zwei Schaltungs-Ergänzungen notwendig. Diese können dem Anhang entnommen werden.

4. Beschreibung der Programme

4.1. Die Monitorerweiterung MAT 85+

4.1.1. Einleitung

Die Erweiterung MAT 85+ ergänzt das Programm MAT 85 um die 10 Kommandos

- BASIC
- COPY
- FIND
- HELP (für MAT 85+)
- INSERT
- PROMMER
- RAM-TEST
- SPS
- VERIFY
- WRITE CONSTANT

Um diese Kommandos ausführen zu können, muß erst die Erweiterung MAT 85+ aufgerufen werden:

Nach dem Einschalten des Mikrocomputers und dem Betätigen der Leertaste (Space) meldet sich das Betriebsprogramm MAT 85. Nachdem es eine Liste aller zur Verfügung stehenden Kommandos ausgegeben hat, erscheint die "Bereit"-Meldung (Prompt) von MAT 85:

```
KMD > _
```

Durch Betätigung der Leertaste kann nun die Monitor-Erweiterung MAT 85+ aufgerufen werden. Die Erweiterung meldet sich mit dem Prompt:

```
KMD+> _
```

Betätigt man die Leertaste erneut, so wird wieder MAT 85 aufgerufen:

```
KMD > _
```

Die beiden "Bereit"-Meldungen unterscheiden sich durch das "+"-Zeichen. Wenn der Mikrocomputer bereit ist, die Kommandos des Betriebsprogramms MAT 85 auszuführen, wird "KMD > " ausgegeben. Wenn er bereit ist, die Kommandos der Erweiterung MAT 85+ auszuführen, wird "KMD+>" ausgegeben. Durch Betätigen der Leertaste kann man zwischen MAT 85 und MAT 85+ wechseln.

4.1.2. Kommando-Eingabe

Die Bereitschaft zur Annahme eines Kommandos zeigt die Monitorerweiterung durch den Ausdruck "KMD+>" an. Jedes der oben aufgelisteten Kommandos kann durch die Eingabe seines ersten Buchstabens und durch anschließendes Betätigen der Taste "CR" (Carriage Return, Wagenrücklauf) aufgerufen werden. Daraufhin druckt das Programm den vollständigen Kommandonamen und fordert eventuell zusätzliche Informationen an. Falsch eingegebene Zeichen können durch die Betätigung der Taste "DEL" (Delete, Löschen) gelöscht werden. Soll das Kommando abgebrochen werden, so muß die Taste "ESC" (Escape, Flucht) betätigt werden. Man wechselt dann automatisch von der Monitorerweiterung MAT 85+ zum Betriebsprogramm MAT 85. Dieses quittiert die Eingabe von "ESC" durch ein akustisches Signal und fordert durch das Ausdrucken von "KMD > " ein neues Kommando an. Will man wieder die Erweiterung MAT 85+ aufrufen, muß die Leertaste erneut betätigt werden.

4.1.3. Reset-Betätigung

Bei der Betätigung des RESET-Tasters bricht der Mikrocomputer augenblicklich alle Aktionen ab und meldet sich durch die Ausgabe von

```
*** RESET ***

KMD >
```

Das Prompt zeigt, daß nun das Betriebsprogramm MAT 85 aktiv ist. Es erwartet die Eingabe eines neuen Kommandos. Will man die Erweiterung MAT 85+ aufrufen, muß die Leertaste erneut betätigt werden.

4.1.4. Bildschirm-Modus, Drucker-Modus

Die Erweiterung unterscheidet zwischen Bildschirm-Modus und Drucker-Modus.

Im Bildschirm-Modus wird der Ausdruck bei längeren Protokollen nach jeder Bildschirmseite gestoppt und der Text "==> SPACE" ausgegeben. Der Bediener erhält damit die Möglichkeit, die Protokollierung auch bei hohen Übertragungsgeschwindigkeiten zu verfolgen. Der Ausdruck wird fortgesetzt, wenn die SPACE-Taste (Leertaste) kurz betätigt wird. Durch das Betätigen der Taste "ESC" wird das Kommando abgebrochen und von der Erweiterung MAT 85+ zum Betriebsprogramm MAT 85 gewechselt.

Wenn sich das Programm im Drucker-Modus befindet, erfolgt ein kontinuierlicher Ausdruck aller Ausgaben auf dem Bildschirm und auf dem Drucker.

Immer dann, wenn das Programm sein Prompt "KMD+>" ausgibt und auf eine Kommando-Eingabe wartet, kann durch gleichzeitiges Betätigen der Taste "CONTROL" und der Taste "P" zwischen dem Bildschirm-Modus und dem Drucker-Modus gewechselt werden.

Der Bildschirm-Modus ist vorbelegt, d.h.: wird das System eingeschaltet, arbeitet es solange in diesem Modus, bis vom Bediener der Drucker-Modus gewählt wird. Nur wenn statt einer Datensichtstation beispielsweise ein Fernschreiber angeschlossen ist, wird von Anfang an im Drucker-Modus gearbeitet.

4.1.5. Bediener-Führung

Unabhängig vom Bildschirm- bzw. Drucker-Modus wird der System-Bediener vom Programm geführt, indem es eventuell zusätzliche Informationen für die Kommando-Ausführung (z.B. Adressen) anfordert. Dabei erfolgt sofort eine Kontrolle, ob die Eingabedaten dem notwendigen Format entsprechen. Ist dies nicht der Fall, wird der Bediener durch ein akustisches Signal auf seinen Fehler aufmerksam gemacht. Die falsche Eingabe wird ignoriert.

4.1.6. Kommando-Kurzbeschreibung

BASIC — Mit dem Kommando BASIC kann der BASIC-Interpreter aufgerufen werden.

COPY — Mit dem Kommando COPY ist es möglich, den Inhalt eines Speicherbereichs in einen anderen Speicherbereich zu kopieren. Quell- und Ziel-Speicherbereich dürfen sich überschneiden.

FIND — Das FIND-Kommando ermöglicht das Suchen eines Zeichens oder einer Zeichenfolge im Speicher. Die Zeichen können in den Formaten ASCII, binär, dezimal und hexadezimal eingegeben werden. Die Zeichenfolgen, die mit diesem Kommando gesucht werden können, dürfen bis zu 16 Zeichen lang sein.

HELP — Das HELP-Kommando listet alle Kommandos der Monitorerweiterung MAT 85+ auf.

INSERT — Mit dem INSERT-Kommando kann in Programmen, die mit dem ASSEMBLER-Kommando von MAT 85 erstellt wurden, nachträglich Platz geschaffen werden. Mit dem ASSEMBLER-Kommando kann dieser Platz dann mit zusätzlichen Befehlen gefüllt werden.

PROMMER — Durch das Kommando PROMMER wird das Programm für den EPROM-Programmierer aufgerufen.

RAM-TEST — Mit dem RAM-TEST-Kommando kann ein RAM-Speicherbereich getestet werden. Der Inhalt des Speichers wird dabei nicht verändert.

SPS — Das SPS-Kommando ruft das SPS-Programm auf.

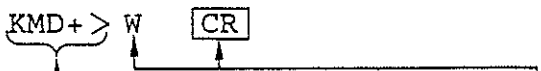
VERIFY — Mit dem VERIFY-Kommando können die Inhalte zweier Speicherbereiche miteinander verglichen werden. Alle Unterschiede werden angezeigt.

WRITE CONSTANT — Mit diesem Kommando kann ein RAM-Speicherbereich mit einer Konstanten gefüllt werden. Diese kann in den Formaten ASCII, binär, dezimal und hexadezimal eingegeben werden.

4.1.7. Hinweise zur Beschreibung der Kommandos

Im folgenden werden Aufruf und Verwendung der einzelnen Kommandos ausführlich beschrieben. Anhand von Bildschirmausdrucken und Kommentaren kann die Anwendung eines jeden Kommandos nachvollzogen werden.
Um Tastatureingaben, Bildschirmausdrucke und Kommentare übersichtlich und allgemeingültig zu gestalten, werden einige Abkürzungen und Darstellungsweisen verwendet, die am Beispiel des WRITE CONSTANT-Kommandos erklärt werden sollen:

Aufruf des WRITE CONSTANT-Kommandos (wenn KMD+> angezeigt wird):

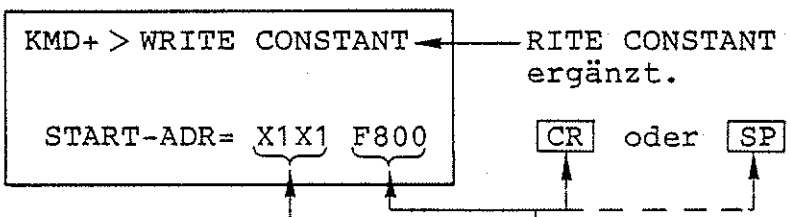


Bereitschafts-meldung

Buchstabe W gefolgt von der Wagenrücklauftaste eintippen.

Eingerahmte Zeichen stellen TASTEN mit einer Steuerfunktion dar! [CR] steht für die Taste Carriage-Return, [SP] steht für die Space-Taste (Leertaste).

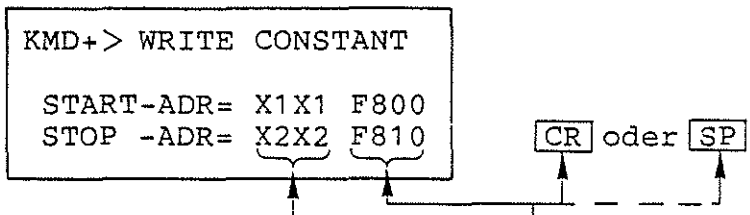
Wirkung:



Hier steht die Vorgabe-Adresse des Computers (Vorschlag). Wird ihr Wert akzeptiert, müssen Sie die [CR] - oder [SP] - Taste betätigen.

Wenn nicht, müssen Sie die hexadezimale Adresse derjenigen Speicherzeile eintippen, die als erste bearbeitet werden soll. Hier wird diese Adresse zu F800 gewählt, indem hintereinander die Zeichen F800 eingetippt werden, gefolgt von der Betätigung der [CR] - oder [SP] - Taste. Allgemeingültig wird diese Adresse "Neu-Adresse" genannt und durch "Y1Y1" gekennzeichnet.

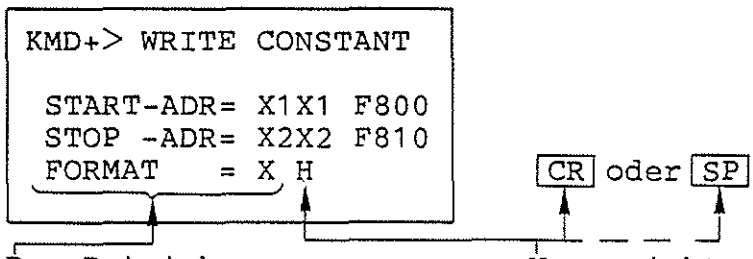
Wirkung:



Hier steht die Vorgabe-Adresse des Computers (Vorschlag). Wird ihr Wert akzeptiert, müssen Sie die CR - oder SP - Taste betätigen.

Wenn nicht, müssen Sie die hexadezimale Adresse derjenigen Speicherzeile eintippen, die als letzte bearbeitet werden soll. Hier wird diese Adresse zu F810 gewählt, indem hintereinander die Zeichen F810 eingetippt werden, gefolgt von der Betätigung der CR - oder SP - Taste. Allgemeingültig wird diese Adresse "Neu-Adresse" genannt und durch "Y2Y2" gekennzeichnet.

Wirkung:



Das Betriebsprogramm schlägt vor, die Daten in dem angegebenen Format darzustellen. Wird die Vorgabe akzeptiert, müssen Sie CR oder SP betätigen.

Wenn nicht, müssen Sie einen der Buchstaben A, B, D oder H gefolgt von CR oder SP eintippen. Hier wird H für hexadezimale Darstellung der Daten eingegeben. Allgemeingültig wird das Format "Neu-Format" genannt und durch "Y" gekennzeichnet. Die Bedeutung der verschiedenen Formate wird bei der Beschreibung des WRITE-CONSTANT-Kommandos erläutert.

Wirkung:

```
KMD+ > WRITE CONSTANT

 START-ADR= X1X1 F800
 STOP -ADR= X2X2 F810
 FORMAT   = X H
 DATA     = _
```

Das Programm erwartet weitere Eingaben, die in der Beschreibung des WRITE-CONSTANT-Kommandos erläutert werden.

Im folgenden Bild sind die oben beschriebenen Schritte in gekürzter Form dargestellt. Diese Art der Darstellung wird bei der Beschreibung der Kommandos verwendet.

Schirmbild (oft Ausschnitt)	Eingabe, Kommentar
KMD+ > WRITE CONSTANT	W [CR] eintippen, "RITE CONSTANT" wird ergänzt.
START-ADR= X1X1 Y1Y1	X1X1 = Vorgabe; Neu: Y1Y1 [CR] oder [SP] Vorgabe: [CR] oder [SP]
STOP -ADR= X2X2 Y2Y2	X2X2 = Vorgabe; Neu: Y2Y2 [CR] oder [SP] Vorgabe: [CR] oder [SP]
FORMAT = X Y	X = Vorgabe; Neu: Y [CR] oder [SP] Vorgabe: [CR] oder [SP]
	Y = A: ASCII (druckb. Zeichen) B: Binär (0,1) D: Dezimal (0 ... 9) H: Hexadezimal (0 ... F)

4.1.8. Kommando-Beschreibung

4.1.8.1 Das HELP-Kommando

Mit dem HELP-Kommando lassen sich die Namen der zulässigen Kommandos der Monitorerweiterung MAT 85+ in alphabetischer Reihenfolge ausdrucken.

Aufruf und Handhabung:

KMD+> HELP	H [CR] eintippen "ELP" wird ergänzt
(Kommando-Ausführung)	
KMD+>_	nächstes Kommando

Zur Kommando-Ausführung:

- Nach dem Ausdrucken aller Kommandonamen erfolgt ein Rücksprung in die Kommando-Routine, KMD+>_ wird ausgegeben.
- Zum Aufruf eines der Kommandos muß nur der 1. Buchstabe, gefolgt von der Taste [CR], eingegeben werden.
- Eingaben, die vor Betätigung von [CR] erfolgen, können mit der Taste [DEL] gelöscht werden.

4.1.8.2. Das BASIC-Kommando

Mit dem BASIC-Kommando kann der BASIC-Interpreter aufgerufen werden. Zum Betrieb des Interpreters wird zusätzlich zur Speichermindestbestückung für MAT 85+ ab der Adresse 6000 RAM-Speicher benötigt (siehe auch Bild 1). Die Obergrenze dieses RAM-Speichers ist beliebig. Weitere Hinweise zum Betrieb des BASIC-Interpreters entnehmen Sie bitte dem Kapitel 4.4.

Aufruf und Handhabung:

KMD+> BASIC	B [CR] eintippen, "ASIC" wird ergänzt
BFZ-STEUER-BASIC V2.4	Basic meldet sich
READY >_	und ist bereit, Befehle entgegenzunehmen
(Eingabe von Befehlen)	
>QUIT	QUIT [CR] eintippen
KMD+> _	Rückkehr nach MAT 85+

Fehlermeldungen:

Ist ab Adresse 6000 kein RAM-Speicher vorhanden, so wird die Meldung

```
*** KEIN RAM ***
```

ausgegeben. MAT 85+ meldet sich mit seinem Prompt und ist bereit, weitere Kommandos entgegen zu nehmen.

4.1.8.3. Das COPY-Kommando

Mit dem Kommando COPY ist es möglich, den Inhalt eines Speicherbereiches in einen anderen Speicherbereich zu kopieren. Dabei dürfen sich Quell- und Ziel-Speicherbereich überschneiden.
Der Quellspeicherbereich beginnt bei "START-ADR" und endet bei "STOP-ADR". Der Ziel-Speicherbereich beginnt bei "ZIEL-BER".

Aufruf und Handhabung:

KMD+> COPY	C [CR] eintippen, "OPY" wird ergänzt
START-ADR = X1X1 Y1Y1	X1X1 = Vorgabe; Neu: Y1Y1 [CR] oder [SP] Vorgabe: [CR] oder [SP]
STOP -ADR = X2X2 Y2Y2	X2X2 = Vorgabe; Neu: Y2Y2 [CR] oder [SP] Vorgabe: [CR] oder [SP]
ZIEL -BER = X3X3 Y3Y3	X3X3 = Vorgabe; Neu: Y3Y3 [CR] oder [SP] Vorgabe: [CR] oder [SP]

Schirmbild	Eingabe,Kommentar
KMD+> COPY START-ADR =E000 0000 STOP -ADR =E7FF 0100 ZIEL -BER =0000 F800	C [CR] 0000 [CR] oder [SP] 0100 [CR] oder [SP] F800 [CR] oder [SP] Es wird der Quellspeicherinhalt von 0000 (START) bis 0100 (STOP) in den Zielspeicher ab F800 (ZIEL) kopiert

Fehlermeldungen:

Die Fehlermeldung

```
*** KEIN RAM ***
```

wird ausgegeben, wenn der Ziel-Speicherbereich nicht vollständig mit RAM-Speicherbausteinen bestückt ist. Wenn die START-Adresse größer als die STOP-Adresse ist, wird die Fehlermeldung

```
*** START-ADR > STOP-ADR ***
```

ausgegeben. In diesem Fall kann die Eingabe der START-Adresse sofort wiederholt werden.

Hinweis: Kopieren Sie nicht in den Speicherbereich FC00 bis FFFF, da hier wichtige Daten gespeichert sind, die nicht verändert werden dürfen!

Softwarepaket SP 1

Name:

MAT 85+ / COPY-Kommando

Datum:

Drucken Sie den Inhalt des Speicherbereichs von 0000 bis 0006 mit Hilfe des PRINT-Kommandos von MAT 85 in hexadezimaler Form auf dem Bildschirm aus und tragen Sie die Werte in die Tabelle 1 ein.

C3 MAT 85+

Drucken Sie anschließend den Inhalt des Speicherbereichs F800 bis F806 aus und tragen Sie die Werte in Tabelle 2 ein.

Kopieren Sie nun mit dem COPY-Kommando den Inhalt des Speicherbereichs 0000 bis 0006 in den Speicherbereich ab F800.

Drucken Sie den Inhalt des Speicherbereichs F800 bis F806 erneut aus, tragen Sie die Werte in Tabelle 3 ein und vergleichen Sie die ausgedruckten Werte mit den Tabellen 1 und 2.

C4 MAT 85+

Tabelle 1
Inhalt des Speicherbereichs 0000 bis 0006

Adresse	0000	0001	0002	0003	0004	0005	0006
Inhalt							

Tabelle 2
Inhalt des Speicherbereichs F800 bis F806
vor der Ausführung des COPY-Kommandos

Adresse	F800	F801	F802	F803	F804	F805	F806
Inhalt							

Tabelle 3
Inhalt des Speicherbereichs F800 bis F806
nach der Ausführung des COPY-Kommandos

Adresse	F800	F801	F802	F803	F804	F805	F806
Inhalt							

Versuchen Sie, den Inhalt des Speicherbereichs FC00 bis FFFF in den Speicherbereich 0000 bis 03FF zu kopieren. Warum kommt es zu einer Fehlermeldung ?

Antwort:

4.1.8.4. Das FIND-Kommando

Das FIND-Kommando ermöglicht das Suchen eines Zeichens oder einer Zeichenfolge im Speicher. Die Zeichen können in den Formaten ASCII, Binär, Dezimal und Hexadezimal eingegeben werden. Da alle Eingabe-Zeichen in Großbuchstaben umgewandelt werden, können im ASCII-Format keine Kleinbuchstaben gesucht werden. Die Zeichenfolgen, die mit dem FIND-Kommando gesucht werden können, dürfen bis zu 16 Zeichen lang sein. Der Speicherbereich, in dem das Zeichen bzw. die Zeichenfolge gesucht wird, reicht von START- bis STOP-Adresse einschließlich. Die gesuchten Zeichenfolgen müssen vollständig innerhalb dieser Grenzen liegen. Wird ein Zeichen gefunden, so gibt MAT 85+ eine entsprechende Meldung aus. Wird die gesuchte Zeichenfolge nicht gefunden, so entfällt die Meldung "GEFUNDEN BEI ADRESSE" und MAT 85+ gibt seine "Bereit"-Meldung "KMD+>" aus.
Die eingegebenen Suchzeichen werden im RAM ab Adresse FD15 zwischengespeichert. Wenn der Zwischenspeicher im Suchbereich liegt, erfolgt die Meldung:

"GEFUNDEN BEI ADRESSE: FD15"

Aufruf und Handhabung:

```
KMD+> FIND

START-ADR = X1X1 Y1Y1

STOP -ADR = X2X2 Y2Y2

FORMAT    = X Y
```

F [CR] eintippen,
"IND" wird ergänzt

X1X1 = Vorgabe;
Neu: Y1Y1 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X2X2 = Vorgabe;
Neu: Y2Y2 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X = Vorgabe;
Neu: Y [CR] oder [SP]
Vorgabe: [CR] oder [SP]

Y = A: ASCII (druckb. Zeichen)
= B: Binär (0,1)
= D: Dezimal (0...9)
= H: Hexadezimal (0...F)

Beispiel für START-Adresse = 0000
STOP -Adresse = 1FFF
FORMAT = H (hexadezimal)

Schirmbild	Eingabe,Kommentar
KMD+> FIND START-ADR =FF04 0000 STOP -ADR =FF00 1FFF FORMAT =H ZEICHENFOLGE: 42 46 5A GEFUNDEN BEI ADRESSE: 0092 GEFUNDEN BEI ADRESSE: 0190	F [CR] 0000 [CR] oder [SP] 1FFF [CR] oder [SP] [CR] oder [SP] (Vorgabe) 42 [SP] 46 [SP] 5A [CR] Sollen weitere Zeichen eingegeben werden, so muß [SP] gedrückt werden. Die letzte Eingabe muß mit [CR] beendet werden. Bei diesen Adressen wurden die Zeichenfolgen gefunden.

Beispiel für START-Adresse = 0000
STOP -Adresse = 1FFF
FORMAT = D (dezimal)

Schirmbild	Eingabe,Kommentar
KMD+> FIND START-ADR =0000 STOP -ADR =1FFF FORMAT =H D ZEICHENFOLGE: 66 70 90 GEFUNDEN BEI ADRESSE: 0092 GEFUNDEN BEI ADRESSE: 0190	F [CR] [CR] oder [SP] (Vorgabe) [CR] oder [SP] (Vorgabe) D [CR] oder [SP] 66 [SP] 70 [SP] 90 [CR] Sollen weitere Zeichen eingegeben werden, so muß [SP] gedrückt werden. Die letzte Eingabe muß mit [CR] beendet werden. Bei diesen Adressen wurden die Zeichenfolgen gefunden

Beispiel für START-Adresse = 0000
STOP -Adresse = 1FFF
FORMAT = B (binär)

Schirmbild	Eingabe,Kommentar
KMD+> FIND	F [CR]
START-ADR =0000	[CR] oder [SP] (Vorgabe)
STOP -ADR =1FFF	[CR] oder [SP] (Vorgabe)
FORMAT =D B	B [CR] oder [SP]
ZEICHENFOLGE: 01000010	01000010 [SP]
01000110	01000110 [SP]
01011010	01011010 [CR]
	Sollen weitere Zeichen eingegeben werden, so muß [SP] gedrückt werden. Die letzte Eingabe muß mit [CR] beendet werden.
GEFUNDEN BEI ADRESSE: 0092	Bei diesen Adressen wurden die
GEFUNDEN BEI ADRESSE: 0190	Zeichenfolgen gefunden

Beispiel für START-Adresse = 0000
STOP -Adresse = 1FFF
FORMAT = A (ASCII)

Schirmbild	Eingabe,Kommentar
KMD+> FIND	F [CR]
START-ADR =0000	[CR] oder [SP] (Vorgabe)
STOP -ADR =1FFF	[CR] oder [SP] (Vorgabe)
FORMAT =B A	A [CR] oder [SP]
ZEICHENFOLGE: B	B [SP]
F	F [SP]
Z	Z [CR]
	Sollen weitere Zeichen eingegeben werden, so muß [SP] gedrückt werden. Die letzte Eingabe muß mit [CR] beendet werden.
GEFUNDEN BEI ADRESSE: 0092	Bei diesen Adressen wurden die
GEFUNDEN BEI ADRESSE: 0190	Zeichenfolgen gefunden

Fehlermeldungen:

Eine Fehlermeldung wird ausgegeben, wenn versucht wird, mehr als 16 Zeichen einzugeben. In diesem Fall wird die Meldung

```
*** BUFFER VOLL ***
```

ausgegeben und der Suchvorgang gestartet. Eine Fehlermeldung wird ebenfalls ausgegeben, wenn die Start-Adresse größer als die Stop-Adresse ist. In diesem Fall hat der Bediener die Möglichkeit, die Adressen neu einzugeben.

Softwarepaket SP 1

Name:

MAT 85+ / FIND-Kommando

Datum:

Suchen Sie die unten angegebenen Zeichenfolgen im Speicherbereich 2000 bis 3FFF und tragen Sie die Adressen, an denen die Zeichenfolgen gefunden wurden, in die Tabellen ein.

F5 MAT 85+

a) Zeichenfolge: 23 02 56 ,Format: hexadezimal

gefunden bei:

b) Zeichenfolge: F I N ,Format: ASCII

gefunden bei:

c) Zeichenfolge: 00101000 01000011 00101001 ,Format: binär

gefunden bei:

d) Zeichenfolge: 72 69 76 ,Format: dezimal

gefunden bei:

4.1.8.5 Das INSERT-Kommando

Mit dem INSERT-Kommando können nachträglich NOP-Befehle (NOP = no operation, Leerschritt) in Programme eingefügt werden, die mit dem ASSEMBLER des Betriebsprogramms MAT 85 erstellt wurden. Hierdurch wird in diesen Programmen Platz für das Einsetzen zusätzlicher Befehle geschaffen. Label- und Sprungadressen werden automatisch korrigiert. Die eingefügten NOP-Befehle können anschließend mit dem ASSEBLER durch die gewünschten Befehle ersetzt werden.

Eingegeben werden muß:

- Start- und Stop-Adresse des zu ändernden Programms.
  Bei der Stop-Adresse ist das letzte Byte des Programms anzugeben.
- die Adresse, an der das Befehls-Byte des ersten einzufügenden Befehls stehen soll. Die einzufügenden Befehle müssen lückenlos aufeinander folgen.
- die Länge der einzufügenden Befehle in Bytes (dezimal). Für jedes Byte wird ein NOP-Befehl eingefügt.

Sollen an verschiedenen Stellen eines Programms Befehle eingefügt werden, so muß das INSERT-Kommando mehrfach aufgerufen werden.

Aufruf und Handhabung:

```
KMD+> INSERT

PROGRAMM:
START-ADR = X1X1 Y1Y1

STOP -ADR = X2X2 Y2Y2

INSRT-ADR = X3X3 Y3Y3

BYTE -ANZ = Y
```

I [CR] eintippen,
"NSERT" wird ergänzt

X1X1 = Vorgabe;
Neu: Y1Y1 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X2X2 = Vorgabe;
Neu: Y2Y2 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

Adresse, bei der der erste Befehl eingefügt werden soll:
X3X3 = Vorgabe;
Neu: Y3Y3 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

Anzahl der Bytes, die eingefügt werden sollen (dezimal):
Eingabe: Y [CR] oder [SP]
Y = 0 ... 255

Beispiel:

Es soll das Programm

```
START: IN  01
       OUT 02
       JMP START
```

eingegeben werden:

```
KMD > ASSEMBLER




 START-ADR = 0000 F800
F800 DB 01      START: IN  01
F802 C3 00F8           JMP START
F805            END
 *** RESTART ? (JA/NEIN) N
```

A [CR] oder [SP]
(Aufruf nur von MAT 85 aus.
ASSEMBLER-Beschreibung
siehe FPÜ 7.1.)

F800 [CR] oder [SP]
START: IN 01 [CR]
JMP START [CR]
END [CR]
N [CR]

Der Programmierer merkt, daß er die Anweisung "OUT 02" vergessen hat. Er kann zunächst mit dem INSERT-Kommando nachträglich Platz für die beiden vergessenen Bytes schaffen.

```
KMD > _


KMD+> INSERT
 PROGRAMM:
 START-ADR =AAAA F800
 STOP -ADR =2000 F804
 INSRT-ADR =F800 F802
 BYTE -ANZ =2
```

[SP] (Auruf von MAT 85+)

I [CR]
Programm-Adressen:
F800 [CR] oder [SP]
F804 [CR] oder [SP]
F802 [CR] oder [SP]
2 [CR] oder [SP]

Hierauf verschiebt MAT 85+ den Programmteil zwischen der INSRT- und der STOP-Adresse (hier C3 00 F8) um die angegebene BYTE-Anzahl (hier 2) und setzt in die freigewordenen Speicherstellen (hier F802, F803) jeweils einen NOP-Befehl ein.

Das Ergebnis kann mit dem DISASSEMBLER kontrolliert werden:

```
KMD+> _

KMD > DISASSEMBLER



 START-ADR =F800
 STOP -ADR =F806
F800 DB 01     START: IN    01
F802 00               NOP
F803 00               NOP
F804 C3 00F8          JMP START
                      END
```

[SP] (Aufruf von MAT 85)

D [CR] oder [SP]
(Aufruf nur von MAT 85 aus.
DISASSEMBLER-Beschreibung
siehe FPÜ 7.1.)

[CR] oder [SP] (Vorgabe)
[CR] oder [SP] (Vorgabe)

NOPs wurden durch INSERT-
Kommando eingefügt.

Nun sollen die NOPs mit dem vergessenen Befehl "OUT 02" überschrieben werden:

```
KMD > ASSEMBLER


 START-ADR =F802
F802 D3 02            OUT  02
F804          END
 *** RESTART ? (JA/NEIN) N
```

A [CR] oder [SP]

Einfügen des Befehls:
[CR] oder [SP] (Vorgabe)
OUT 02 [CR]
END [CR]
N [CR]

ACHTUNG: Hier muß die Restart-Abfrage mit N für NEIN beantwortet werden !

Kontrolle mit dem Disassembler:

```
KMD > DISASSEMBLER
 START-ADR =F800
 STOP -ADR =F806
F800 DB 01     START: IN  01
F802 D3 02            OUT 02
F804 C3 00F8          JMP START
                      END
```

D [CR] oder [SP]
[CR] oder [SP] (Vorgabe)
[CR] oder [SP] (Vorgabe)

OUT-Befehl ist eingefügt

Der Aufruf des DISASSEMBLERS diente hier nur Demonstrationszwecken. Er ist zur Durchführung des INSERT-Kommandos nicht erforderlich.

Fehlermeldungen:

Die Fehlermeldung

```
*** KEIN RAM ***
```

wird ausgegeben, wenn der um die BYTE-ANZ zu verschiebende Programmteil (zwischen INSRT-ADR und STOP-ADR einschließlich) nicht mehr in den zur Verfügung stehenden RAM-Speicher paßt.

Die Fehlermeldung

```
*** INSERT-ADR. LIEGT NICHT IM PGM-BEREICH ***
```

wird ausgegeben, wenn eine INSRT-ADR eingegeben wurde, die außerhalb des Programmbereichs (zwischen START- und STOP-ADR) liegt. Der Bediener wird anschließend aufgefordert, neue Adressen einzugeben.

Die Fehlermeldung

```
*** START-ADR > STOP-ADR ***
```

wird ausgegeben, wenn die Start-Adresse oberhalb der Stop-Adresse liegt. Auch in diesem Fall können die Adressen sofort neu eingegeben werden.

Softwarepaket SP 1

Name:

MAT 85+ / INSERT-Kommando

Datum:

Vollziehen Sie das obige Beispiel nach.

I5 MAT 85+

Fügen Sie zwischen IN- und OUT-Befehl den 1-Byte-Befehl CMA (Complement Accu) ein.

Versuchen Sie anschließend den 2-Byte-Befehl ANI 07 nach dem CMA-Befehl einzufügen. Beantworten Sie dabei die Frage

*** RESTART ? (JA/NEIN)

des Assemblers falsch, indem Sie "J" für JA eingeben. Welches Programm zeigt der DISASSEMBLER an?

Programm: _____

_____

_____

_____

_____

Durch das Beantworten der Frage mit J wurde der Befehl "RST 1" an den eingefügten Befehl angehängt. Der OUT-Befehl wurde dadurch überschrieben.

Damit an den eingefügten Befehl kein RST-1-Befehl angehängt wird, muß die RESTART-Frage stets mit

N (NEIN)

beantwortet werden

4.1.8.6. Das PROMMER-Kommando

Mit dem PROMMER-Kommando kann das Programm für den EPROM-Programmierer aufgerufen werden. Weitere Hinweise zu diesem Programm entnehmen Sie bitte dem Abschnitt 4.2.

Aufruf und Handhabung:

KMD+ > PROMMER	P [CR] eintippen, "ROMMER" wird ergänzt
BFZ-EPROM-PROGRAMMER V2.1	Das Programm meldet sich,
COMPARE HELP PROGRAM QUIT READ TEST	gibt eine Liste der Kommandos aus
P>_	und ist bereit Kommandos entgegenzunehmen.
P>QUIT	Verlassen des Programms: Q [CR] eintippen, "UIT" wird ergänzt
KMD+ >_	Rückkehr nach MAT 85+

4.1.8.7. Das RAM-TEST-Kommando

Mit dem RAM-TEST-Kommando kann ein RAM-Speicherbereich (von START- bis STOP-Adresse einschließlich) getestet werden. Bei Fehlern werden die Adresse, der Soll-Wert und der Ist-Wert ausgegeben. Soll-Wert ist der Wert, der in der Speicherzelle stehen sollte. Ist-Wert ist der Wert, der in der Speicherzelle steht. Wird kein Fehler gefunden, so meldet sich MAT 85+ sofort mit seinem Prompt "KMD+>". Nach dem Test ist der Inhalt des getesteten Speicherbereichs unverändert.

Aufruf und Handhabung:

```
KMD+>RAM-TEST

 START-ADR = X1X1 Y1Y1


 STOP -ADR = X2X2 Y2Y2
```

R [CR] eintippen,
"AM-TEST" wird ergänzt

X1X1 = Vorgabe;
Neu: Y1Y1 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X2X2 = Vorgabe;
Neu: Y2Y2 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

Beispiel 1:

Schirmbild	Eingabe,Kommentar
KMD+>RAM-TEST START-ADR =2000 F800 STOP -ADR =FFFF F8FF KMD+>_	R [CR] F800 [CR] oder [SP] F8FF [CR] oder [SP] Teste Speicher F800 bis F8FF. Kein Fehler festgestellt

Beispiel 2:

Schirmbild	Eingabe, Kommentar
KMD+> RAM-TEST START-ADR =F800 0000 STOP -ADR =F8FF 0001	R [CR] 0000 [CR] oder [SP] 0001 [CR] oder [SP] Versuch, den ROM-Speicher zu testen.
ADR: 0000 SOLL: 55 IST: C3 ADR: 0001 SOLL: 55 IST: 49 ADR: 0000 SOLL: AA IST: C3 ADR: 0001 SOLL: AA IST: 49	1. Fehler 2. Fehler erneute Fehlermeldungen bei den ersten Adressen, nun mit einem anderen Sollwert. (Jede Speicherzelle wird mit zwei Werten, 55 und AA, getestet.)
KMD+> _	

Fehlermeldungen:

Die Fehlermeldung

```
*** START-ADR > STOP-ADR ***
```

wird ausgegeben, wenn die Start-Adresse größer als die Stop-Adresse ist. Die Eingabe der Adressen kann dann sofort wiederholt werden.

Softwarepaket SP 1

Name:

MAT 85+ / RAM-TEST-Kommando

Datum:

Testen Sie den Speicher ab Adresse F900 bis F9FF. R3 MAT 85+

Testen Sie den Speicher ab Adresse 2000 bis 2001. Warum meldet der RAM-TEST hier Fehler ?

Antwort:

4.1.8.8. Das SPS-Kommando

Mit dem SPS-Kommando kann das SPS-Programm (Speicherprogrammierbare Steuerung) aufgerufen werden. Zum Betrieb dieses Programms ist es erforderlich, daß (zusätzlich zur Mindestbestückung für MAT 85+) der Speicherbereich E000 bis E7FF mit RAM-Speicherbausteinen bestückt ist (siehe auch Bild 1).
Weitere Hinweise zum SPS-Programm entnehmen Sie bitte dem Abschnitt 4.3.

Aufruf und Handhabung:

KMD+>SPS	S [CR] eintippen, "PS" wird ergänzt
BFZ-SPS-PROGRAMM V2.1	Das Programm meldet sich,
EDIT GO HELP LIST NEW READ STEP TRACE WRITE QUIT	listet seine Kommandos auf
SPS>_	und ist bereit, Befehle entgegenzunehmen.
SPS>QUIT	Verlassen des Programms: Q [CR] eintippen, "UIT" wird ergänzt
KMD+>_	Rückkehr nach MAT 85+

Fehlermeldungen:

Ist ab Adresse E000 kein RAM-Speicher vorhanden, wird die Fehlermeldung

```
*** KEIN RAM ***
```

ausgegeben. MAT 85+ meldet sich dann mit seinem Prompt und ist bereit, weitere Kommandos entgegenzunehmen.

4.1.8.9. Das VERIFY-Kommando

Mit dem VERIFY-Kommando kann der Inhalt eines Speicherbereichs (von START- bis STOP-Adresse einschließlich) mit dem Inhalt eines anderen Speicherbereichs (ab VERGL-BER) verglichen werden. Treten bei dem Vergleich Unterschiede auf, so werden die jeweiligen Adressen der beiden Speicherbereiche zusammen mit den Speicherinhalten angezeigt. Sind die beiden Speicherinhalte völlig identisch, so meldet sich MAT 85+ sofort mit seinem Prompt "KMD+>".

Aufruf und Handhabung:

```
KMD+> VERIFY

 START-ADR = X1X1 Y1Y1

 STOP- ADR = X2X2 Y2Y2

 VERGL-BER = X3X3 Y3Y3
```

V [CR] eintippen,
"ERIFY" wird ergänzt

X1X1 = Vorgabe;
Neu: Y1Y1 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X2X2 = Vorgabe;
Neu: Y2Y2 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X3X3 = Vorgabe;
Neu: Y3Y3 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

Beispiel:

Schirmbild	Eingabe,Kommentar
KMD+> VERIFY	V [CR]
START-ADR =0000	[CR] oder [SP] (Vorgabe)
STOP -ADR =07FF 0002	0002 [CR] oder [SP]
VERGL-BER =0000 F800	F800 [CR] oder [SP]
0000 --> C3 F800 --> 6D	(keine Übereinstimmung)
0001 --> 49 F801 --> FF	(keine Übereinstimmung)
0002 --> 01 F802 --> EF	(keine Übereinstimmung)
KMD+> _	Auf Ihrem Bildschirm können auch andere Speicher-Inhalte erscheinen.

Nun wird der Inhalt des Speicher-Bereichs von 0000 bis 0002 nach F800 bis F802 kopiert (siehe auch COPY-Kommando):

KMD+> COPY	C [CR]
START-ADR =0000	[CR] oder [SP] (Vorgabe)
STOP -ADR =0002	[CR] oder [SP] (Vorgabe)
ZIEL -BER =F800	[CR] oder [SP] (Vorgabe)

Ein erneuter Vergleich der beiden Speicherbereiche wird durchgeführt:

KMD+> VERIFY	V [CR]
START-ADR =0000	[CR] oder [SP] (Vorgabe)
STOP -ADR =0002	[CR] oder [SP] (Vorgabe)
VERGL-BER =F800	[CR] oder [SP] (Vorgabe)
KMD+> _	Stimmen die Inhalte der beiden Speicherbereiche überein, so meldet sich MAT 85+ sofort mit seinem Prompt "KMD+>"

Fehlermeldungen:

Die Fehlermeldung

```
*** START-ADR > STOP-ADR ***
```

wird ausgegeben, wenn die Start-Adresse größer als die Stop-Adresse ist. In diesem Fall können die Adressen erneut eingegeben werden.

Softwarepaket SP 1

Name:

MAT 85+ / VERIFY-Kommando

Datum:

V3 MAT 85+

Vollziehen Sie das obige Beispiel mit anderen Adressen nach.

Kopieren sie dabei nicht in den Speicherbereich FC00 bis FFFF, da dort wichtige Daten gespeichert sind, die nicht verändert werden dürfen.

4.1.8.10. Das WRITE CONSTANT-Kommando

Mit dem WRITE CONSTANT-Kommando ist es möglich, einen Speicherbereich (von START- bis STOP-Adresse einschließlich) mit einem konstanten Wert zu füllen. Der Wert, mit dem der Speicher gefüllt werden soll, kann in den Formaten ASCII, Binär, Dezimal oder Hexadezimal eingegeben werden.

Aufruf und Handhabung:

```
KMD+> WRITE CONSTANT

START-ADR = X1X1 Y1Y1

STOP -ADR = X2X2 Y2Y2

FORMAT    = X Y

DATA      = _
```

W [CR] eintippen,
"RITE CONSTANT" wird ergänzt

X1X1 = Vorgabe;
Neu: Y1Y1 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X2X2 = Vorgabe;
Neu: Y2Y2 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X = Vorgabe;
Neu: Y [CR] oder [SP]
Vorgabe: [CR] oder [SP]

Y = A: ASCII (druckb. Zeichen)
= B: Binär (0,1)
= D: Dezimal (0...9)
= H: Hexadezimal (0...F)

Eingabe der Konstanten, mit der der Speicher gefüllt werden soll.

Hinweis: Füllen Sie den Speicherbereich FC00 bis FFFF nicht mit einer Konstanten, da hier Daten gespeichert sind, die nicht zerstört werden dürfen!

Beispiel für START-Adresse = F800
STOP -Adresse = F80F
FORMAT = H (hexadezimal)

Schirmbild	Eingabe,Kommentar
KMD+> WRITE CONSTANT START-ADR =FF04 F800 STOP -ADR =FF00 F80F FORMAT =H DATA =57	W [CR] F800 [CR] oder [SP] F80F [CR] oder [SP] [CR] oder [SP] (Vorgabe) 57 [CR] oder [SP] Der Speicherbereich F800 bis F80F wird mit dem hexadezimalen Wert 57 gefüllt

Beispiel für START-Adresse = F800
STOP -Adresse = F80F
FORMAT = D (dezimal)

Schirmbild	Eingabe,Kommentar
KMD+> WRITE CONSTANT START-ADR =F800 STOP -ADR =F80F FORMAT =H D DATA =88	W [CR] [CR] oder [SP] (Vorgabe) [CR] oder [SP] (Vorgabe) D [CR] oder [SP] 88 [CR] oder [SP] Der Speicherbereich F800 bis F80F wird mit dem dezimalen Wert 88 gefüllt.

Beispiel für START-Adresse = F800
STOP -Adresse = F80F
FORMAT = B (binär)

Schirmbild	Eingabe,Kommentar
KMD+> WRITE CONSTANT START-ADR =F800 STOP -ADR =F80F FORMAT =D B DATA =10001001	W CR CR oder SP (Vorgabe) CR oder SP (Vorgabe) B CR oder SP 10001001 CR oder SP Der Speicherbereich F800 bis F80F wird mit dem binären Wert 10001001 gefüllt

Beispiel für START-Adresse = F800
STOP -Adresse = F80F
FORMAT = A (ASCII)

Schirmbild	Eingabe,Kommentar
KMD+> WRITE CONSTANT START-ADR =F800 STOP -ADR =F80F FORMAT =B A DATA =Z	W CR CR oder SP (Vorgabe) CR oder SP (Vorgabe) A CR oder SP Z CR oder SP Der Speicherbereich F800 bis F80F wird mit dem ASCII-Zeichen "Z" gefüllt

Fehlermeldungen:

Die Fehlermeldung

*** KEIN RAM ***

wird ausgegeben, wenn der zu füllende Speicherbereich nicht vollständig mit RAM-Speicherbausteinen bestückt ist.

Die Fehlermeldung

*** START-ADR > STOP-ADR ***

wird ausgegeben, wenn die START-Adresse größer als die STOP-Adresse ist. In diesem Fall kann die Eingabe der Adressen sofort wiederholt werden.

Softwarepaket SP 1

Name:

MAT 85+ / WRITE-CONSTANT-Kommando

Datum:

W5 MAT 85+

Füllen Sie den Speicherbereich von F900 bis F9FF mit

FORMAT	DATA
hexadezimal	41
dezimal	66
binär	01000011
ASCII	D

und kontrollieren Sie das Ergebnis jeweils mit dem PRINT-Kommando des Betriebsprogramms MAT 85.

4.2. Der EPROM-Programmer

Das Softwarepaket SP 1 enthält das Programm "EPROM-PROGRAMMER" für einen EPROM-Programmierer mit dem EPROMs vom Typ 2716 programmiert werden können. Um dieses Programm einsetzen zu können, wird die Baugruppe EPROM-Programmierer BFZ/MFA 4.3.a benötigt. Die Adresse dieser Baugruppe muß auf DX (hexadezimal) eingestellt werden (siehe Bild 2).

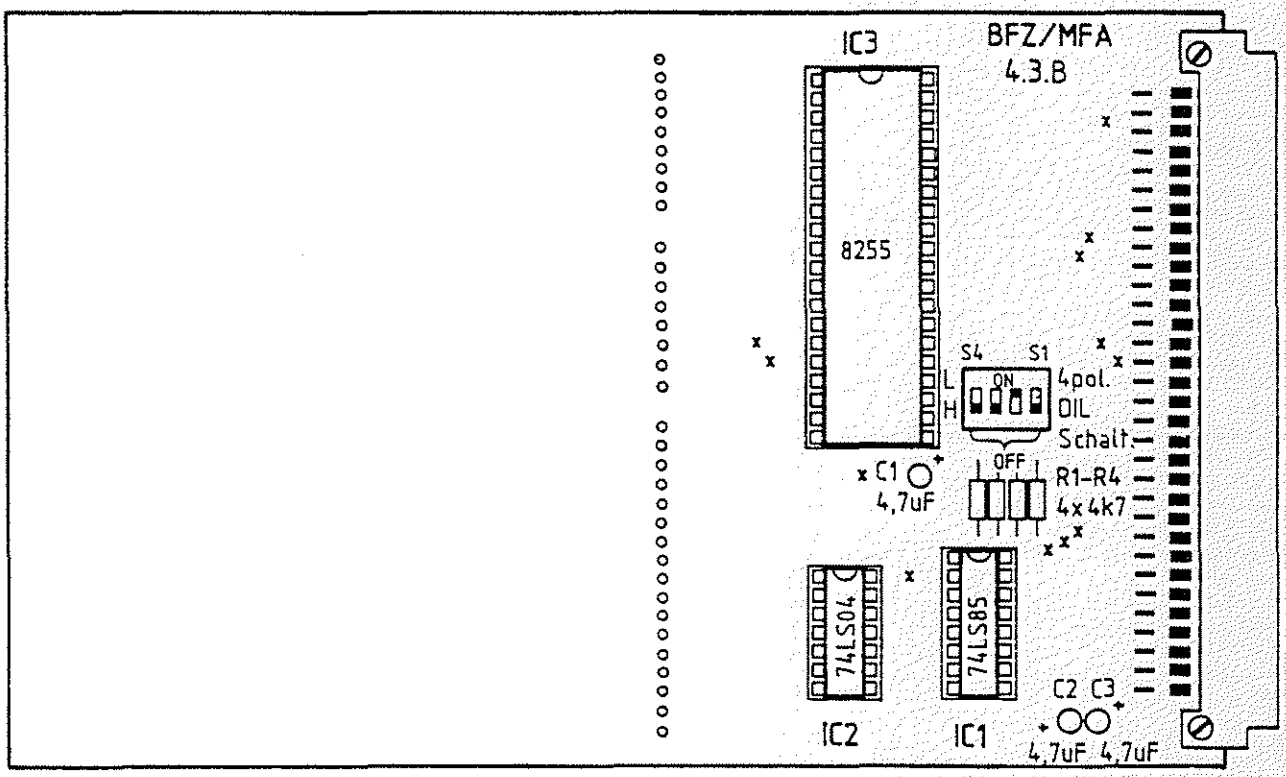


Bild 2: Adreßeinstellung der Baugruppe EPROM-Programmierer

EPROM-PROGRAMMER

Die Programmierspannung von 27 V wird von einer externen Spannungsquelle geliefert. Der Pluspol dieser Spannungsquelle wird mit der linken Buchse auf der Frontplatte des EPROM-Programmierers verbunden, der Minuspol mit der rechten. Das EPROM muß so in den Sockel auf der Frontplatte eingesetzt werden, daß Pin 1 links oben ist (Kerbe im EPROM-Gehäuse nach oben). Weitere Hinweise über die Hardware entnehmen Sie bitte der FPÜ 4.3.a.

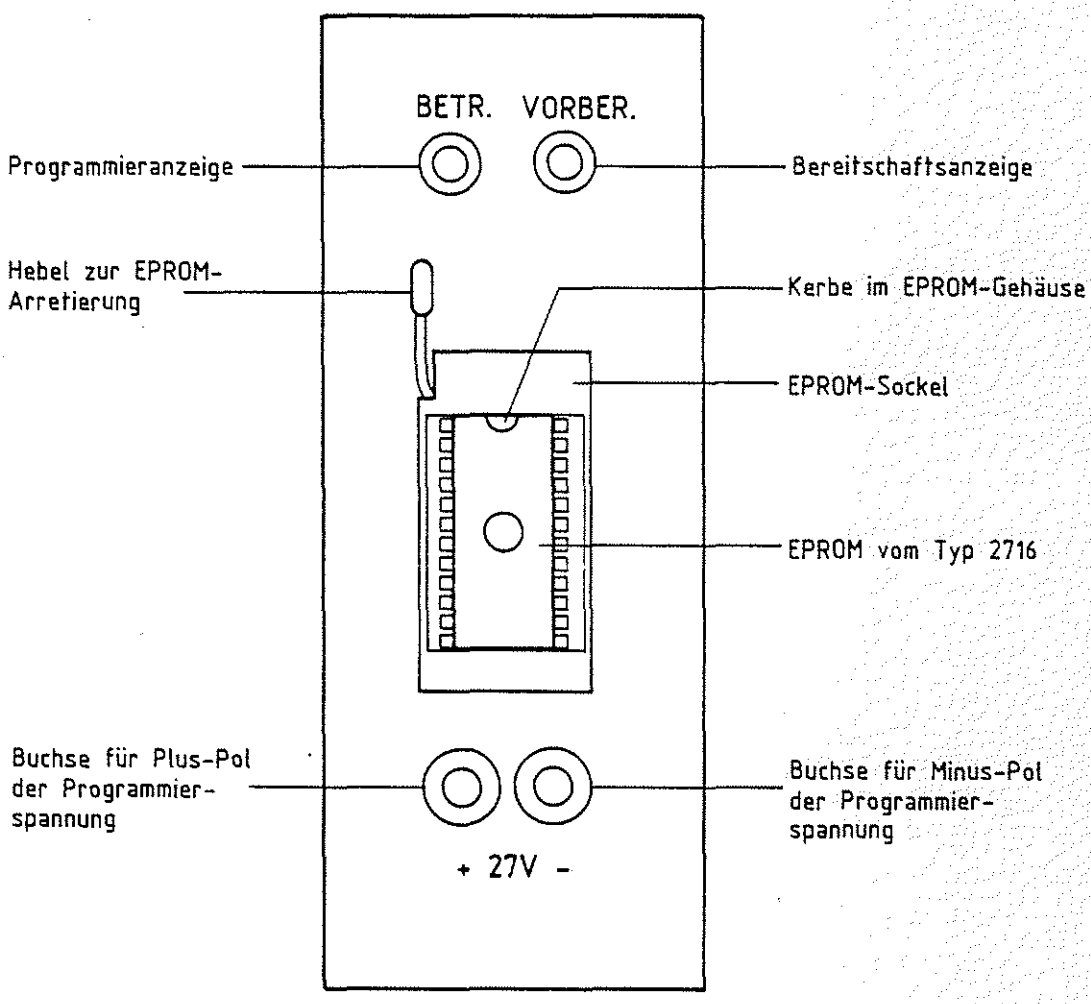


Bild 3: Frontplatte des EPROM-Programmierers

EPROM-PROGRAMMER / Aufruf

Um das Programm "EPROM-Programmer" starten zu können, muß erst die Monitor-Erweiterung MAT 85+ aufgerufen werden:

Nach dem Einschalten des Mikrocomputers und dem Betätigen der Leertaste (Space) meldet sich das Betriebsprogramm MAT 85. Nachdem es eine Liste aller zur Verfügung stehenden Kommandos ausgegeben hat, erscheint die "Bereit"-Meldung (Prompt) von MAT 85:

```
KMD >_
```

Durch Betätigen der Leertaste kann die Monitor-Erweiterung MAT 85+ aufgerufen werden. Die Erweiterung meldet sich mit dem Prompt:

```
KMD+>_
```

Zum Aufruf des Programms für den EPROM-Programmierer - im folgenden Text PROMMER genannt - muß nun die Taste "P", und anschließend die "CR"-Taste (Carriage Return, Wagenrücklauf), betätigt werden:

`KMD+ > PROMMER`	P [CR] eintippen, "ROMMER" wird ergänzt
`BFZ-EPROM-PROGRAMMER V2.1`	Das Programm meldet sich,

`COMPARE`
`HELP`
`PROGRAMM`
`QUIT`
`READ`
`TEST`
	druckt eine Liste der verfügbaren Kommandos aus

`P>_`	und ist bereit, Kommandos entgegenzunehmen.

4.2.1. Kommando-Eingabe

Die Bereitschaft zur Annahme eines Kommandos zeigt der Prommer durch den Ausdruck "P>" an. Jedes der oben aufgelisteten Kommandos kann durch die Eingabe seines ersten Buchstabens und durch anschließendes Betätigen der Taste "CR" (Carriage Return, Wagenrücklauf) aufgerufen werden. Daraufhin druckt das Programm den vollständigen Kommandonamen und fordert eventuell zusätzliche Informationen an. Falsch eingegebene Zeichen können durch die Betätigung der Taste "DEL" (Delete, Löschen) gelöscht werden. Soll das Kommando abgebrochen werden, so muß die Taste "ESC" (Escape, Flucht) betätigt werden. Man wechselt dann automatisch vom Prommer zum Betriebsprogramm MAT 85. Dieses quittiert die Eingabe von "ESC" durch ein akustisches Signal und fordert durch das Ausdrucken von "KMD >" ein neues Kommando an. Will man wieder den Prommer aufrufen, so muß zuerst die Leertaste betätigt werden (Aufruf von MAT 85+) und anschließend das Zeichen "P" (gefolgt von CR) eingegeben werden.

4.2.2. Bildschirm-Modus, Drucker-Modus

Das Prommer-Programm unterscheidet zwischen Bildschirmmodus und Drucker-Modus. Immer dann, wenn das Programm sein Prompt "P>" ausgibt und auf eine neue Kommando-Eingabe wartet, kann durch gleichzeitiges Drücken der Tasten "CONTROL" und "P" zwischen dem Bildschirm- und dem Drucker-Modus gewechselt werden. Im Drucker-Modus erfolgen alle Ausgaben auf dem Bildschirm und auf dem Drucker.

4.2.3. Bediener-Führung

Unabhängig vom Bildschirm- bzw. Drucker-Modus wird der System-Bediener vom Programm geführt, indem es eventuell zusätzliche Informationen für die Kommando-Ausführung (z.B. Adressen) anfordert. Dabei erfolgt sofort eine Kontrolle, ob die Eingabedaten dem notwendigen Format entsprechen. Ist dies nicht der Fall, wird der Bediener durch ein akustisches Signal auf seinen Fehler aufmerksam gemacht.

4.2.3.1. RAM-START/STOP-Adresse, EPROM-START-Adresse

Bei fast allen Kommandos erfragt das Programm die Werte

```
RAM:
START-ADR =
STOP -ADR =
EPROM:
START-ADR =
```

Die Bedeutung dieser Werte ist vom jeweiligen Kommando abhängig und wird in der Beschreibung dieser Kommandos erläutert.

Generell gilt:
Die RAM-START-ADR ist frei wählbar. Der Vorgabe-Wert der RAM-STOP-Adresse wird vom Programm durch Addition von 07FF zur RAM-START-Adresse ermittelt. Der Speicherbereich, der durch die RAM-START-Adresse und die RAM-STOP-Adressen-Vorgabe begrenzt wird, hat dadurch automatisch die Größe von 2-K-Byte. Dies entspricht genau der Speichergröße eines EPROMs vom Typ 2716.

Wenn die RAM-START-Adresse größer als F7FF ist, dann wird als RAM-STOP-Adresse der Wert FFFF vorgeschlagen.

Eine Fehlermeldung wird ausgegeben, wenn die vom Bediener eingegebenen RAM-START- und RAM-STOP-Adressen um mehr als 07FF auseinanderliegen oder wenn die RAM-START-Adresse größer als die RAM-STOP-Adresse ist.

Der Wert für die EPROM-START-Adresse darf im Bereich von 0000 bis 07FF liegen. Der Vorschlagswert für die EPROM-START-ADR ist immer 0000.

Es wird eine Fehlermeldung ausgegeben, wenn der durch RAM-START- und RAM-STOP-Adresse eingegrenzte Speicherbereich größer ist als der Bereich von EPROM-START-Adresse bis 07FF (dem Maximalwert für die EPROM-Adresse).

4.2.4. Kommando-Kurzbeschreibung

COMPARE — Mit dem COMPARE-Kommando kann der EPROM-Inhalt (oder ein Teil davon) mit dem Inhalt eines Speicherbereichs verglichen werden.

HELP — Das HELP-Kommando listet alle Kommandos des Prommers in alphabetischer Reihenfolge.

PROGRAM — Mit dem PROGRAM-Kommando kann das EPROM (oder ein Teil davon) mit dem Inhalt eines Speicherbereichs programmiert werden.

QUIT — Durch das Kommando QUIT kann man das Prommer-Programm verlassen. Man gelangt dann zur Monitorerweiterung MAT 85+ zurück.

READ — Das READ-Kommando ermöglicht das Einlesen des EPROM-Inhaltes (oder eines Teils davon) in einen RAM-Speicherbereich.

TEST — Mit dem TEST-Kommando kann geprüft werden, ob das EPROM gelöscht (und damit für die Programmierung bereit) ist.

4.2.5. Beschreibung der Kommandos

4.2.5.1. Das HELP-Kommando

Mit dem HELP-Kommando lassen sich die Namen der zulässigen Kommandos des Prommers in alphabetischer Reihenfolge ausdrucken.

Aufruf und Handhabung:

P > HELP	H [CR] eintippen, "ELP" wird ergänzt
(Kommando-Ausführung)	
P>_	nächstes Kommando

Zur Kommando-Ausführung:

- Nach dem Ausdrucken aller Kommandonamen erfolgt ein Rücksprung in die Kommando-Routine (P>).
- Zum Aufruf eines der Kommandos muß nur der 1. Buchstabe, gefolgt von der Taste [CR], eingegeben werden.
- Eingaben, die vor der Betätigung von [CR] erfolgen, können mit der Taste [DEL] gelöscht werden.

4.2.4.2. Das COMPARE-Kommando

Mit dem COMPARE-Kommando kann ein Speicherinhalt (von RAM-START- bis RAM-STOP-Adresse einschließlich) mit dem EPROM-Inhalt (oder einem Teilinhalt) ab EPROM-START-Adresse verglichen werden. Sind EPROM-Inhalt und Speicher-Inhalt völlig identisch, so meldet der Prommer: READY. Sind die Inhalte nicht völlig identisch, so meldet er: NOT READY.

Aufruf und Handhabung:

```
P > COMPARE

RAM:

 START-ADR = X1X1 Y1Y1


 STOP -ADR = X2X2 Y2Y2


EPROM:

 START-ADR = X3X3 Y3Y3
```

C [CR] eintippen,
"OMPARE" wird ergänzt

zuerst werden die Adressen
für den Speicher angefordert

X1X1 = Vorgabe;
Neu: Y1Y1 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X2X2 = Vorgabe;
Neu: Y2Y2 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

die Adresse für das EPROM wird
angefordert

X3X3 = Vorgabe;
Neu: Y3Y3 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

Der Speicherinhalt von RAM-
START-ADR bis RAM-STOP-ADR wird
mit dem EPROM-Inhalt ab EPROM-
START-ADR verglichen.

Die EPROM-START-ADR ist bis
07FF wählbar.

Beispiel 1:

Schirmbild	Eingabe,Kommentar
P> COMPARE	C [CR]
RAM:	
START-ADR =0000 F800	F800 [CR] oder [SP]
STOP -ADR =FFFF	[CR] oder [SP] (Vorgabe)
EPROM:	
START-ADR =0000	[CR] oder [SP] (Vorgabe)
	Der gesamte EPROM-Inhalt (EPROM-ADR 0000 bis 07FF) wird mit dem Inhalt des Speichers von F800 bis FFFF verglichen.
NOT READY	} Je nach Vergleichsergebnis
READY	

Beispiel 2:

Schirmbild	Eingabe,Kommentar
P> COMPARE	C [CR]
RAM:	
START-ADR =F800	[CR] oder [SP] (Vorgabe)
STOP -ADR =FFFF F802	F802 [CR] oder [SP]
EPROM:	
START-ADR =0000 0100	0100 [CR] oder [SP]
	Die Inhalte der RAM-Speicherzellen F800, F801 und F802 werden mit den Inhalten der EPROM-Speicherzellen 0100, 0101 und 0102 verglichen
NOT READY	} Je nach Vergleichsergebnis
READY	

Beispiel 3:

Schirmbild	Eingabe, Kommentar
P> COMPARE RAM: START-ADR = 0000 STOP -ADR = 07FF 0010 EPROM: START-ADR = 0000 07FE *** ADR ZU GROSS ***	C [CR] [CR] oder [SP] (Vorgabe) 0010 [CR] oder [SP] 07FE [CR] oder [SP] Der zu vergleichende RAM-Speicherbereich umfaßt 17 Speicherstellen (0000 bis 0010), während das EPROM von der gewählten START-Adresse aus nur 2 Speicher-stellen enthält.

Beispiel: 4

Schirmbild	Eingabe, Kommantar
P> COMPARE RAM: START-ADR =0000 STOP -ADR =07FF 0010 EPROM: START-ADR =0000 08FF *** ADR ZU GROSS ***	C [CR] [CR] oder [SP] (Vorgabe) 0010 [CR] oder [SP] 08FF [CR] oder [SP] Die gewählte EPROM-START-ADR liegt außerhalb des Bereichs 0000 bis 07FF.

4.2.5.3. Das PROGRAM-Kommando

Mit dem PROGRAM-Kommando ist es möglich, den Inhalt eines Speicherbereiches in das EPROM zu programmieren. Für das Programmieren ist eine Spannung von 27 V erforderlich (Anschluß siehe Kapitel 4.2.).
Der Speicher, dessen Inhalt in das EPROM programmiert werden soll, wird durch die RAM-START- und die RAM-STOP-Adresse eingegrenzt und wird ab der EPROM-START-Adresse in das EPROM programmiert. Wenn das Programm während des Programmiervorganges feststellt, daß das EPROM nicht programmierbar ist, wird die Fehlermeldung

```
*** NICHT PROGRAMMIERBAR ***
```

ausgegeben.

Aufruf und Handhabung:

```
P > PROGRAMM

RAM:


 START-ADR = X1X1 Y1Y1


 STOP -ADR = X2X2 Y2Y2


EPROM:

 START-ADR = X3X3 Y3Y3
```

P [CR] eintippen,
"ROGRAMM" wird ergänzt

Das Programm erfragt die Adressen des Speicherbereichs, dessen Inhalt in das EPROM programmiert werden soll:

X1X1 = Vorgabe;
Neu: Y1Y1 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X2X2 = Vorgabe;
Neu: Y2Y2 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

Das Programm erfragt nun die EPROM-START-Adresse:

X3X3 = Vorgabe;
Neu: Y3Y3 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

Der Speicher-Inhalt von RAM-START- bis RAM-STOP-Adresse wird ab EPROM-START-Adr. in das EPROM programmiert.

Beispiel 1:

Schirmbild	Eingabe,Kommentar
P> PROGRAM RAM: START-ADR =0000 F800 STOP -ADR =FFFF EPROM: START-ADR =0000	P [CR] F800 [CR] oder [SP] [CR] oder [SP] (Vorgabe) [CR] oder [SP] (Vorgabe) Der Speicher-Inhalt von F800 bis FFFF wird ab EPROM-Adresse 0000 in das EPROM programmiert.

Beispiel 2:

Schirmbild	Eingabe,Kommentar
P> PROGRAM RAM: START-ADR =F800 STOP -ADR =FFFF F802 EPROM: START-ADR =0000 0100	P [CR] [CR] oder [SP] (Vorgabe) F802 [CR] oder [SP] 0100 [CR] oder [SP] Der Inhalt der Speicher-Zellen F800, F801 und F802 wird in die EPROM-Speicher-zellen 0100, 0101 und 0102 programmiert.

Beispiel 3:

Schirmbild	Eingabe, Kommentar
P > PROGRAM RAM: START-ADR =0000 F800 STOP -ADR =FFFF FE00 EPROM: START-ADR =0000 *** NICHT PROGRAMMIERBAR ***	P [CR] F800 [CR] oder [SP] FE00 [CR] oder [SP] [CR] oder [SP] (Vorgabe) Die Fehlermeldung kann mehrere Ursachen haben: 1. Es befindet sich kein EPROM-Programmiergerät unter der richtigen Adresse im Baugruppenträger. 2. Die Programmierspannung fehlt. 3. Das zu programmierende EPROM ist bereits programmiert. 4. Das EPROM ist defekt.

4.2.5.4. Das QUIT-Kommando

Mit dem QUIT-Kommando kann man das Programm für den EPROM-Programmierer verlassen. In diesem Fall wird automatisch die Monitorerweiterung MAT 85+ aufgerufen.

Aufruf und Handhabung:

```
P > QUIT

KMD+ > _
```

Q [CR] eintippen,
"UIT" wird ergänzt

Die Monitorerweiterung MAT85+ meldet sich mit ihrem Prompt ("Bereit"-Meldung) und ist bereit, Kommandos entgegenzunehmen.

4.2.5.5. Das READ-Kommando

Mit dem READ-Kommando kann ein EPROM-Inhalt (oder ein Teil davon) ab EPROM-START-Adresse in den RAM-Speicher (von RAM-START- bis RAM-STOP-Adresse einschließlich) eingelesen werden.

Aufruf und Handhabung:

```
P > READ

RAM:


 START-ADR = X1X1 Y1Y1


 STOP- ADR = X2X2 Y2Y2


EPROM:

 START-ADR = X3X3 Y3Y3
```

R [CR] eintippen,
"EAD" wird ergänzt

Das Programm fragt nach der START- und STOP-Adresse des RAMs, in den der EPROM-Inhalt eingelesen werden soll.

X1X1 = Vorgabe;
Neu: Y1Y1 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

X2X2 = Vorgabe;
Neu: Y2Y2 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

Das Programm erfragt nun die EPROM-START-Adresse:

X3X3 = Vorgabe;
Neu: Y3Y3 [CR] oder [SP]
Vorgabe: [CR] oder [SP]

Der EPROM-Inhalt ab EPROM-START-Adresse wird in den RAM-Speicher von RAM-START- bis RAM-STOP-Adresse eingelesen

Beispiel 1:

Schirmbild	Eingabe,Kommentar
P > READ RAM: START-ADR =0000 E000 STOP -ADR =E7FF EPROM: START-ADR =0000	R [CR] E000 [CR] oder [SP] [CR] oder [SP] (Vorgabe) [CR] oder [SP] (Vorgabe) Der EPROM-Inhalt ab EPROM-START-Adresse 0000 wird in den RAM-Speicher von E000 bis E7FF eingelesen.

Beispiel 2:

Schirmbild	Eingabe,Kommentar
P > READ RAM: START-ADR =E000 STOP -ADR =E7FF E002 EPROM: START-ADR =0000 0100	R [CR] [CR] oder [SP] (Vorgabe) E002 [CR] oder [SP] 0100 [CR] oder [SP] Der Inhalt der EPROM-Zellen 0100, 0101 und 0102 wird in die Speicherzellen E000, E001 und E002 eingelesen.

Fehlermeldungen:

Eine Fehlermeldung wird ausgegeben, wenn der Speicherbereich von RAM-START- bis RAM-STOP-Adresse nicht mit RAM-Speicherbausteinen bestückt ist. Weitere mögliche Fehlermeldungen sind dem Abschnitt 4.2.3.1. zu entnehmen.

Hinweis:

Lesen Sie keine Daten in den Speicherbereich FC00 bis FFFF ein, da dort wichtige Daten gespeichert sind, die nicht verändert werden dürfen!

4.2.5.6. Das TEST-Kommando

Mit dem TEST-Kommando kann geprüft werden, ob ein EPROM vollständig gelöscht ist. Dies ist dann der Fall, wenn alle Speicherzeilen des EPROMs den Inhalt FF haben. Wenn das EPROM vollständig gelöscht ist, gibt das Programm die Meldung "READY" aus, andernfalls die Meldung "NOT READY".

Aufruf und Handhabung:

P > TEST	T [CR] eintippen, "EST" wird ergänzt
	Das EPROM wird getestet und es erscheint entweder die Meldung
READY	wenn das EPROM vollständig gelöscht ist, oder es erscheint die Meldung
NOT READY	wenn das EPROM nicht vollständig gelöscht ist.

4.3. Das SPS-Programm

Im Gegensatz zu einer klassischen Schützsteuerung, bei der die Verknüpfung zwischen den Eingangssignalen (-Kontakten) und den Ausgangssignalen (-Kontakten) der Steuerung durch die Verdrahtung der Kontakte und Schütze hergestellt wird, erfolgt diese Verknüpfung bei einer Speicherprogrammierbaren Steuerung (SPS) durch ein Programm. Dieses Programm ist zyklisch, das heißt: sind alle Anweisungen abgearbeitet, so wird erneut mit der ersten Anweisung begonnen.
Das SPS-Programm ermöglicht die Programmierung des Mikrocomputers mit Hilfe von Anweisungen und Befehlen, die dem Problem "Steuerungstechnik" angepaßt sind. Häufige Verknüpfungen in der Steuerungstechnik sind die UND- und ODER-Verknüpfungen von Signalen, die der Reihen- und Parallelschaltung von Kontakten entsprechen. Für die Realisierung solcher Verknüpfungen bietet das SPS-Programm einfache symbolische Anweisungen.

Zum Betrieb des SPS-Programms wird zusätzlich zur Speichermindestbestückung für MAT 85+ ab der Adresse E000 RAM-Speicher benötigt. Für die Ein- und Ausgabe wird mindestens je eine 8-Bit-Parallel-Eingabe-Karte bzw. Parallel-Ausgabe-Karte (BFZ/MFA 4.1. bzw. BFZ/MFA 4.2.) benötigt. Weitere Ein- und Ausgabe-Karten können je nach Bedarf eingesetzt werden. Für die Hardware-Timer und für die Anzeige der Merker-Zustände ist als Mindestbestückung die Baugruppe "Zeitwerk (4fach)" BFZ/MFA 4.3.c notwendig. Diese Karte kann entfallen, wenn statt der Hardware-Timer die Software-Timer verwendet werden und auf die Anzeige der Merker-Zustände verzichtet wird.

4.3.1. Die SPS-Operanden

Eingänge, Ausgänge usw. nennt man allgemein Operanden, da mit ihnen etwas gemacht (operiert) wird. Das BFZ-SPS-Programm kennt folgende Operanden:

- Eingänge	abgekürzt: E
- Ausgänge	abgekürzt: A
- Merker	abgekürzt: M
- Hardware-Timer	abgekürzt: T
- Software-Timer	abgekürzt: Z
- Zähler	abgekürzt: C

Da von allen Operanden je 32 verschiedene vorkommen können, müssen sie durch eine Kennzahl unterschieden werden. Die Kennzahlen, die das SPS-Programm akzeptiert, sind zweistellig. Die erste Ziffer darf die Werte 0 bis 3, die zweite Ziffer darf die Werte 0 bis 7 annehmen. Durch diese Ziffern ist eine Zuordnung der einzelnen Operanden zu den Baugruppen gegeben.

4.3.1.1. Eingänge (E)

Durch die Eingangs-Operanden können über die Eingabe-Baugruppe externe Signale (z.B. Schalter, Temperatur-Sensoren ...) abgefragt werden. Eingänge dürfen nur auf der Bedingungsseite eines Ausdrucks vorkommen.

Für je acht Eingänge ist eine 8-Bit-Parallel-Eingabe-Baugruppe (BFZ/MFA 4.1.) erforderlich. Die erste Ziffer der Kennzahl eines Eingangs entspricht der Port-Nummer (Adresse) der verwendeten Baugruppe:

E00 - E07 ⟶ Port 00
E10 - E17 ⟶ Port 01
E20 - E27 ⟶ Port 02
E30 - E37 ⟶ Port 03

Die zweite Ziffer der Kennzahl entspricht der Bit-Nummer:

Ex0 ⟶ Bit 0
Ex1 ⟶ Bit 1
Ex2 ⟶ Bit 2
Ex3 ⟶ Bit 3
Ex4 ⟶ Bit 4
Ex5 ⟶ Bit 5
Ex6 ⟶ Bit 6
Ex7 ⟶ Bit 7

Einige Beispiele:

E00 ⟶ Port 00, Bit 0
E12 ⟶ Port 01, Bit 2
E34 ⟶ Port 03, Bit 4

Die Zuordnung der Operanden-Kennzahlen zu den Port-Nummern kann der Tabelle 1 entnommen werden.

4.3.1.2. Ausgänge (A)

Durch die Ausgangs-Operanden können über die Ausgabe-Baugruppe Signale an externe Geräte (z.B. Motoren, Heizungen ...) gegeben werden. Ausgänge dürfen auf der Bedingungs- und auf der Zuweisungsseite eines Ausdrucks vorkommen.

Für je acht Ausgänge ist eine 8-Bit-Parallel-Ausgabe-Baugruppe (BFZ/MFA 4.2.) erforderlich. Die Kennzahl eines Ausgangs hat die gleiche Bedeutung wie bei den Eingängen (siehe Abschnitt 4.3.1.1.). Die Zuordnung der Operanden-Kennzahlen zu den Port-Nummern kann der Tabelle 1 entnommen werden.

4.3.1.3. Merker (M)

Merker werden verwendet, um Zwischenergebnisse für die spätere Verwendung in anderen Verknüpfungen zu speichern.

Sollen die Merker-Zustände nicht angezeigt werden, so ist für die Merker keine Hardware erforderlich. Wird aber die Anzeige der Merker-Zustände gewünscht, dann ist für je acht Merker eine Baugruppe "Zeitwerk (4fach)" BFZ/MFA 4.3.c erforderlich. Die erste Kennziffer der Merker gibt die Zuordnung zum Port an:

M00 - M07 ⟶ Port 1x
M10 - M17 ⟶ Port 2x
M20 - M27 ⟶ Port 3x
M30 - M37 ⟶ Port 4x

alle Portnummern in hexadezimaler Schreibweise

Beachten Sie bitte den Unterschied zu den Ein- und Ausgängen:

Bei den Ein- und Ausgängen gab die erste Ziffer direkt die Port-Nummer an. Bei den Merkern muß zur ersten Ziffer eine Eins addiert werden. Das Ergebnis gibt dann die erste Ziffer der Port-Nummer (in hexadezimaler Schreibweise) an. Auf den Zeitwerk-Baugruppen wird nur die erste Ziffer der Port-Nummer eingestellt. Die zweite Kennziffer der Merker gibt, wie schon bei den Ein- und Ausgängen, die Bit-Nummer an:

Mx0 ⟶ Bit 0
Mx1 ⟶ Bit 1
Mx2 ⟶ Bit 2
Mx3 ⟶ Bit 3
Mx4 ⟶ Bit 4
Mx5 ⟶ Bit 5
Mx6 ⟶ Bit 6
Mx7 ⟶ Bit 7

Einige Beispiele:

M00 ⟶ Port 1x, Bit 0
M23 ⟶ Port 3x, Bit 3
M33 ⟶ Port 4x, Bit 3

Die Zuordnung der Operanden-Kennzahlen zu den Port-Nummern kann der Tabelle 1 entnommen werden.

4.3.1.4. Die Hardware-Timer (T)

Mit Hilfe der Timer lassen sich Verzögerungszeiten realisieren. Die Timer können durch die einfache Zuweisung

...=Txx

oder durch die SETZ-Anweisung

...=STxx

gestartet werden. Wenn der Timer abgelaufen ist, liefert er das Zustands-Signal "1". Der Timer-Zustand kann in Verknüpfungen auf der Bedingungsseite benutzt werden:

Bedingungs-Seite		Zuweisungsseite
*Txx	=	A00

schalte den Ausgang A00 ein, wenn der Timer Txx abgelaufen ist

Startet man Timer mit der SETZ-Anweisung, werden die Timer nicht gestoppt, wenn die SETZ-Bedingung bei einem späteren Programm-Durchlauf nicht mehr erfüllt ist. Der Zustand "1" (abgelaufen) wird in diesem Fall gespeichert und kann nur mit dem RÜCKSETZ-Befehl gelöscht werden.
Startet man Timer mit einer einfachen Zuweisung, wird der Timer angehalten - bzw. der Zustand "abgelaufen" gelöscht - wenn die Start-Bedingung nicht mehr erfüllt ist.
Dieses Verhalten wird durch Bild 4a und Bild 4b verdeutlicht.

Hardware-Timer dürfen auf der Bedingungs- und auf der Zuweisungsseite eines Ausdrucks vorkommen.

Zum Betrieb der Hardware-Timer wird die Baugruppe "Zeitwerk (4fach)" BFZ/MFA 4.3.c benötigt. Die Laufzeiten der Timer (etwa 1 bis 57 Sekunden) lassen sich mit den auf der Karte befindlichen Spindeltrimmern einstellen. Auf jeder Baugruppe sind vier Zeitwerke (Tx0 - Tx3) vorhanden. Die Baugruppe kann jedoch bis auf acht Zeitwerke ausgebaut werden. Die Kennzahl der Hardware-Timer hat die gleiche Bedeutung wie bei den Merkern. Die Zuordnung der Operanden-Kennzahlen zu den Port-Nummern kann der Tabelle 1 entnommen werden.

4.3.1.5. Kennzahlen der Software-Timer (Z) und der Zähler (C)

Um die Software-Timer und die Zähler verwenden zu können, werden keine zusätzlichen Baugruppen benötigt. Obwohl die Kennzahlen der Software-Timer und der Zähler daher keinen Bezug zur Hardware haben, gilt auch für diese Kennzahlen:

gültige Werte für die erste Ziffer : 0 ... 3
gültige Werte für die zweite Ziffer: 0 ... 7

4.3.1.6. Die Software-Timer (Z)

Mit Hilfe der Timer lassen sich Verzögerungszeiten realisieren. Die Timer können durch die einfache Zuweisung

...=Zxx

oder durch die SETZ-Anweisung

...=SZxx

gestartet werden. Wenn der Timer abgelaufen ist, liefert er das Zustands-Signal "1". Der Timer-Zustand kann in Verknüpfungen auf der Bedingungsseite benutzt werden:

Bedingungsseite	Zuweisungsseite
*Zxx =	A00

schalte den Ausgang A00 ein, wenn der Timer Zxx abgelaufen ist

Startet man Timer mit der SETZ-Anweisung, werden die Timer nicht gestoppt, wenn die SETZ-Bedingung bei einem späteren Programm-Durchlauf nicht mehr erfüllt ist. Der Zustand "1" (abgelaufen) wird in diesem Fall gespeichert und kann nur mit dem RÜCKSETZ-Befehl gelöscht werden.
Startet man Timer mit einer einfachen Zuweisung, wird der Timer angehalten - bzw. der Zustand "abgelaufen" gelöscht - wenn die Start-Bedingung nicht mehr erfüllt ist.
Dieses Verhalten wird durch Bild 4a und Bild 4b verdeutlicht.

Die Laufzeiten der Software-Timer müssen per Programm mit dem Lade-Befehl eingestellt werden. Die Ausführung dieses Befehls verändert den augenblicklichen Timer-Wert und -Zustand nicht. Die Laufzeiten können mit dem Lade-Befehl in Schritten von Zehntel-Sekunden (von 1/10 Sekunde bis ca. 2 Stunden) eingestellt werden. Der Lade-Wert gibt die Laufzeit in Zehntel-Sekunden an:

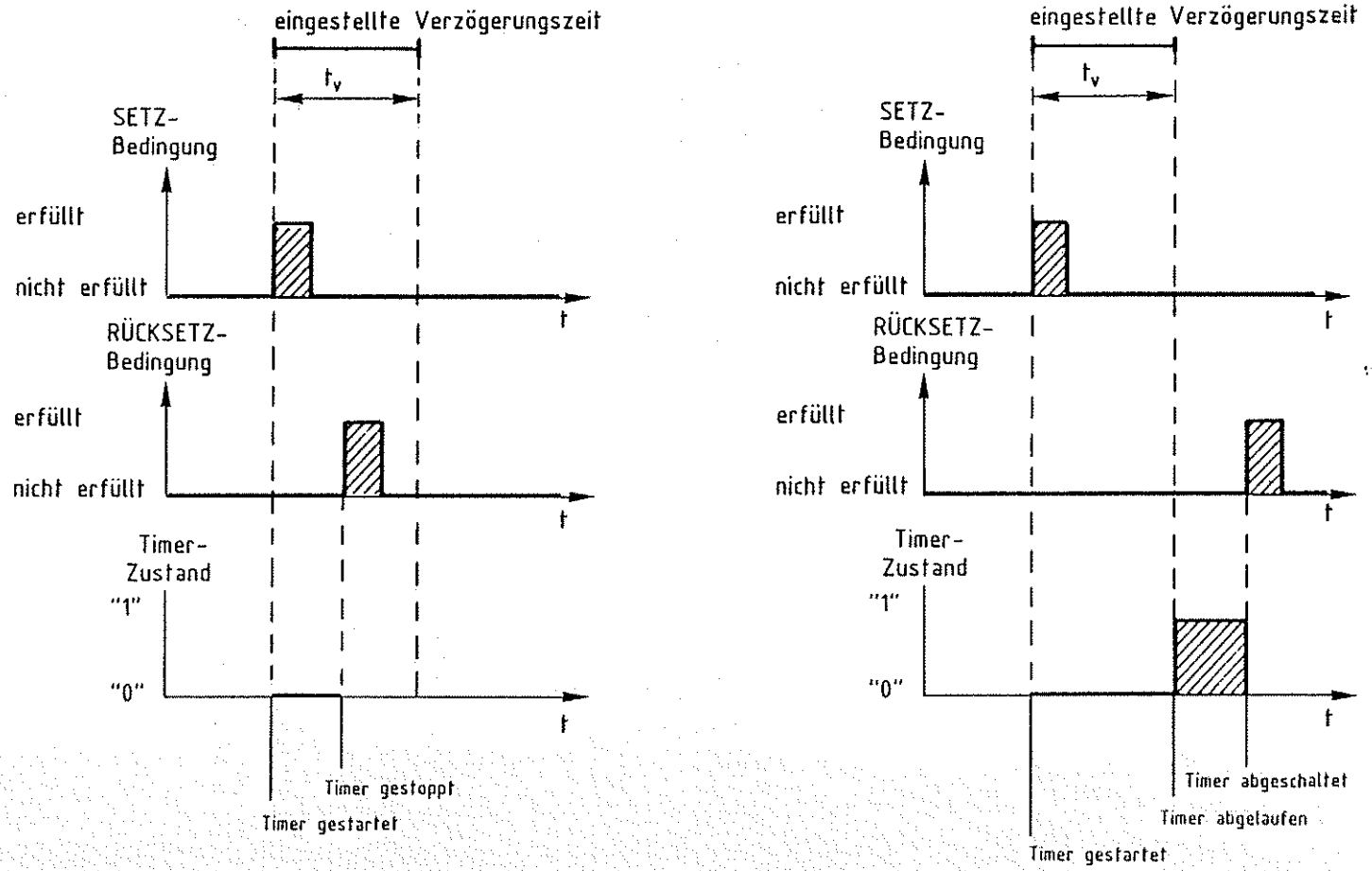
....=LZ00,100

entspricht: Lade den Software-Timer Z00 mit der Laufzeit 10 Sekunden (100/10 Sekunden). Der Ladewert wird gespeichert und durch die Aktivität des Timers nicht verändert. Dies hat zur Folge, daß die einmal abgespeicherte Laufzeit mehrfach im Programm verwendet werden kann. Eine Änderung des Wertes ist durch einen neuen Ladebefehl möglich. Software-Timer dürfen auf der Bedingungs- und auf der Zuweisungsseite eines Ausdrucks vorkommen.

Hinweis: Zum Betrieb der Software-Timer ist die im Anhang beschriebene Schaltungserweiterung erforderlich.

Bei der einfachen Zuweisung " . . . = Txx" bzw. " . . . = Zxx liefert der Timer nur dann einen logischen "1"-Zustand, wenn t_e größer als t_v ist.

Bild 4a: Timerverhalten bei der einfachen Zuweisung



Ein Timer, der über die SETZ-Anweisung " . . . = STxx" bzw. " . . . = SZxx" gestartet wird, liefert nur dann einen logischen "1"-Zustand, wenn ein möglicher RÜCKSETZ-Impuls erst nach Ablauf der eingestellten Laufzeit erzeugt wird. Der logische "1"-Zustand bleibt erhalten, bis er durch eine RÜCKSETZ-Anweisung gelöscht wird.

Bild 4b: Timerverhalten bei den SETZ- und RÜCKSETZ-Anweisungen

4.3.1.7. Die Zähler (C)

Die Zähler dienen zum Zählen von Ereignissen. Alle Zähler zählen vom Ausgangswert ,der mit der Lade-Anweisung geladen wurde, abwärts. Wird der Wert Null erreicht, so wird der logische Zustand "1" angenommen. Dieser Zustand bleibt gespeichert bis ein neuer Lade-Befehl erfolgt. Der Lade-Befehl ändert den Zähler-Wert auf den Lade-Wert und ändert den Zähler-Zustand auf "nicht abgelaufen" (logisch "0"). Zähler dürfen auf der Bedingungs- und auf der Zuweisungsseite eines Ausdrucks vorkommen.

Beispiel:

"Bedingung"=C00

Wenn "Bedingung" erfüllt, dann zähle den Zähler C00 um Eins herab.

4.3.1.8. Beispiele für die Kennzahlen

E00 --- gültige Kennzahl, Port 00, Bit 0
E13 --- gültige Kennzahl, Port 01, Bit 3
E42 --- ungültige Kennzahl, erste Ziffer größer als 3
E18 --- ungültige Kennzahl, zweite Ziffer größer als 7

A00 --- gültige Kennzahl, Port 00, Bit 0
A13 --- gültige Kennzahl, Port 01, Bit 3
A42 --- ungültige Kennzahl, erste Ziffer größer als 3
A18 --- ungültige Kennzahl, zweite Ziffer größer als 7

M00 --- gültige Kennzahl, Port 1x, Bit 0
M13 --- gültige Kennzahl, Port 2x, Bit 3
M42 --- ungültige Kennzahl, erste Ziffer größer als 3
M18 --- ungültige Kennzahl, zweite Ziffer größer als 7

T00 --- gültige Kennzahl, Port 1x, Bit 0
T13 --- gültige Kennzahl, Port 2x, Bit 3
T42 --- ungültige Kennzahl, erste Ziffer größer als 3
T18 --- ungültige Kennzahl, zweite Ziffer größer als 7

Z00 --- gültige Kennzahl, keine Hardware-Zuordnung
Z13 --- gültige Kennzahl, keine Hardware-Zuordnung
Z42 --- ungültige Kennzahl, erste Ziffer größer als 3
Z18 --- ungültige Kennzahl, zweite Ziffer größer als 7

C00 --- gültige Kennzahl, keine Hardware-Zuordnung
C13 --- gültige Kennzahl, keine Hardware-Zuordnung
C42 --- ungültige Kennzahl, erste Ziffer größer als 3
C18 --- ungültige Kennzahl, zweite Ziffer größer als 7

Port-Nr. (Adresse)	Kennzahlen			
	Eingänge	Ausgänge	Merker	Hardware-Timer
00	00 - 07	00 - 07	—	—
01	10 - 17	10 - 17	—	—
02	20 - 27	20 - 27	—	—
03	30 - 37	30 - 37	—	—
1X	—	—	00 - 07	00 - 07
2X	—	—	10 - 17	10 - 17
3X	—	—	20 - 27	20 - 27
4X	—	—	30 - 37	30 - 37

X = wird nicht eingestellt

Tabelle 1: Zuordnung der Operanden-Kennzahlen zu den Port-Nummern

4.3.2. Die Operationen

Um mit den Operanden arbeiten zu können, muß man sie verknüpfen. So soll z.B. eine Lampe nur dann leuchten, wenn die Eingangsschalter in einer bestimmten Stellung stehen. Neben diesen Verknüpfungen sind aber noch andere Befehle notwendig. So muß man zum Beispiel einen Zähler auf einen bestimmten Wert setzen können. Die Verknüpfungen und die zusätzlichen Befehle nennt man im allgemeinen Operationen, da sie mit den Operanden (Eingänge, Ausgänge, Merker usw.) etwas machen. Das BFZ-SPS-Programm kennt folgende Operationen:

Operation	Symbol
UND	*
UND NICHT	*/
ODER	+
ODER NICHT	+/
GLEICH	=
GLEICH NICHT	=/
SETZEN	=S
NICHT SETZEN	=/S
RÜCKSETZEN	=R
NICHT RÜCKSETZEN	=/R
LADEN	=L
NICHT LADEN	=/L

4.3.2.1. Ein SPS-Ausdruck mit Bedingungs- und Zuweisungs-Teil

Ein Ausdruck enthält mindestens ein Gleichheitszeichen. Den Teil links vom Gleichheitszeichen nennt man Bedingungsteil, den Teil rechts vom Gleichheitszeichen nennt man Zuweisungsteil.

```
..........  =  ..........
Bedingungs-Teil      Zuweisungs-Teil
```

Der Bedingungsteil besteht aus einer logischen Verknüpfung (die im Sonderfall nur aus einem Operanden besteht). Die Bedingung ist dann erfüllt (wahr, true), wenn das Verknüpfungsergebnis den logischen Wert "1" annimmt. Um das Verknüpfungsergebnis berechnen zu können, muß man die Zustände der einzelnen Operanden kennen:

Eingänge sind dann logisch "1", wenn die entsprechende LED auf der Frontplatte der Eingabe-Baugruppe leuchtet.

Ausgänge sind dann logisch "1", wenn die entsprechende LED auf der Ausgabe-Baugruppe leuchtet. Die Ausgangsbuchse führt dann H-Pegel.

Merker sind logisch "1", wenn die entsprechende LED auf der Zeitwerk-Baugruppe leuchtet. Der Merker ist dann gesetzt.

Software-Timer, Hardware-Timer und Zähler sind dann logisch "1", wenn sie "abgelaufen" sind.

Das Ergebnis der Verknüpfung im Bedingungsteil wird, wie in der Digitaltechnik, nach den Regeln der boolschen Algebra bestimmt. Die SETZ-, RÜCKSETZ- und LADE-Anweisungen im Zuweisungsteil des Ausdrucks werden nur dann ausgeführt, wenn die Bedingung den Wert "1" hat. Das Gleiche gilt für die Anweisung "......=Cxx" (zähle Zähler xx um Eins herab). In allen anderen Fällen nimmt der Operand im Zuweisungs-Teil das Ergebnis der Bedingung an (ihm wird das Ergebnis zugewiesen).
Manche Ausdrücke haben mehrere Zuweisungen, aber nur eine Bedingung. In diesem Fall ist die Bedingung für alle Zuweisungen gültig.

4.3.2.2. Die GLEICH-Anweisung (=)

Soll eine Lampe, die am Ausgang A00 angeschlossen ist, immer dann leuchten, wenn der Schalter am Eingang E00 betätigt wird, so kann man in einem SPS-Programm schreiben:

*E00=A00

Am Anfang einer Anweisung muß in einem BFZ-SPS-Programm entweder das Symbol "*" oder das Symbol "+" stehen. Durch diese Symbole können die Anweisungen vom SPS-Programm leichter interpretiert werden.
Wie in der Mathematik, so sind auch hier die Ausdruck-Teile rechts und links vom Gleicheitszeichen gleichwertig.
Immer wenn der Schalter E00 (im linken Teil des Ausdrucks) in der EIN-Stellung ist, ist auch die Lampe A00 eingeschaltet.

Das BFZ-SPS-Programm erlaubt Mehrfach-Zuweisungen:

*E00=A00=A01

Bei diesem Ausdruck wurde die Anweisung "=A01" angefügt. Wenn der Schalter E00 in EIN-Stellung ist, so wird neben Ausgang A00 auch der Ausgang A01 eingeschaltet. Das Ergebnis des Bedingungsteils wird beiden Ausgängen zugewiesen.

4.3.2.3. Die UND-Verknüpfung (*)

Will man, daß die Lampe am Ausgang nur dann leuchtet, wenn der Schalter am Eingang E00 und der Schalter am Eingang E01 betätigt sind, so muß man schreiben:

*E00*E01=A00

Das "*"-Symbol steht für die UND-Verknüpfung.

4.3.2.4. Die ODER-Verknüpfung (+)

Eine andere mögliche Forderung wäre, daß die Lampe am Ausgang A00 nur dann leuchten soll, wenn entweder der Schalter am Eingang E00 oder der Schalter am Eingang E01 oder beide Schalter betätigt werden. Diese Forderung kann man durch folgende SPS-Anweisung beschreiben:

*E00+E01=A00

Die ODER-Verknüpfung wird durch das "+"-Symbol dargestellt.

4.3.2.5. Die SETZ-Anweisung (=S)

Manchmal muß ein Impuls gespeichert werden. Dies ist z.B. dann notwendig, wenn eine Lampe durch die kurzzeitige Betätigung eines Tasters eingeschaltet werden soll. Ist die Lampe am Ausgang A00 angeschlossen und der Taster am Eingang E00, dann würde die Lampe bei dem Ausdruck

*E00=A00

nur solange leuchten, wie der Taster betätigt wird.

Die Forderung wird erfüllt, wenn man die folgende Anweisung verwendet:

*E00=SA00

Der Buchstabe "S" steht für die SETZ-Anweisung. Der Ein-Zustand wird gespeichert, wie bei einem Schütz mit Selbsthaltung oder einem bistabilen Kippglied. Dieser gespeicherte Zustand muß durch eine RÜCKSETZ-Anweisung gelöscht werden.

4.3.2.6. Die RÜCKSETZ-Anweisung (=R)

Mit der RÜCKSETZ-Anweisung kann eine SETZ-Anweisung rückgängig gemacht werden.

Ein Beispiel:

*E00=SA00
*E01=RA00

Der erste Ausdruck wurde bereits im Abschnitt 4.3.2.5. (SETZ-Anweisung) erläutert: wird der Taster am Eingang E00 kurzzeitig betätigt, so wird der Ein-Zustand gespeichert und die Lampe am Ausgang A00 leuchtet auch dann noch, wenn man den Taster los läßt. Diese Speicherung des Ein-Zustandes wird durch den zweiten Ausdruck erst dann aufgehoben, wenn der Taster (oder Schalter) am Eingang E01 betätigt wird. Ist die Bedingung "Taster E01 betätigt" erfüllt, so wird die RÜCKSETZ-Anweisung ausgeführt und der Ausgang A00 wird ausgeschaltet (der gespeicherte Zustand wird gelöscht). In diesem Beispiel wird davon ausgegangen, daß nie beide Schalter (E00, E01) gleichzeitig betätigt sind.

4.3.2.7. Die LADE-Anweisung (=L)

Im BFZ-SPS-Programm können auch Software-Timer (Z) und Zähler (C, Counter) verwendet werden. Will man mit diesen Timern und Zählern arbeiten, so müssen sie mit einem bestimmten Wert geladen werden. Die Software-Timer zählen von diesem Wert im 1/10-Sekunden-Takt abwärts. Die Zähler zählen, abhängig von bestimmten Bedingungen, ebenfalls abwärts. Eine Lade-Anweisung hat die Form:

"Bedingung"=LZ00,12345

In diesem Beispiel wird der Software-Timer Z00 mit dem dezimalen Wert 12345 geladen, wenn die Bedingung erfüllt ist. Die Lade-Anweisung für einen Zähler sieht entsprechend aus:

"Bedingung"=LC00,12345

Der Zähler (Counter) C00 wird dann mit dem dezimalen Wert 12345 geladen, wenn die Bedingung erfüllt ist.

Der Lade-Wert darf zwischen 0 und 65535 (dezimal) liegen.

4.3.2.8. Die Negation (/)

Durch das Symbol "/" kann ein Operanden-Zustand negiert (umgekehrt) werden. Es ist ebenso möglich, das gesamte Verknüpfungsergebnis zu negieren:

*/E00=A00	Der Ausgang A00 wird immer dann auf "Ein" geschaltet, wenn der Eingang E00 nicht "Ein" ist. Sonst wird der Ausgang auf "Aus" geschaltet.
E00/E01=SA00	Der Ausgang A00 wird nur dann gesetzt (Speicherung des "Ein"-Zustands), wenn E00 logisch "1" ist und E01 nicht logisch "1" ist.
E00/E01=/SA00	Der Ausgang A00 wird nur dann gesetzt, wenn E00 nicht logisch "1" ist und E01 logisch "1" ist. (Beachten Sie den Unterschied zum vorhergehenden Beispiel).

4.3.2.8. Syntax-Diagramm

Aus dem folgenden Syntax-Diagramm für SPS-Anweisungen (Bild 6) kann man die richtige Schreibweise aller Programmbefehle ablesen. Um die Handhabung zu verdeutlichen, soll ein Syntax-Diagramm erläutert werden, das die Bildung von Zahlen darstellt:

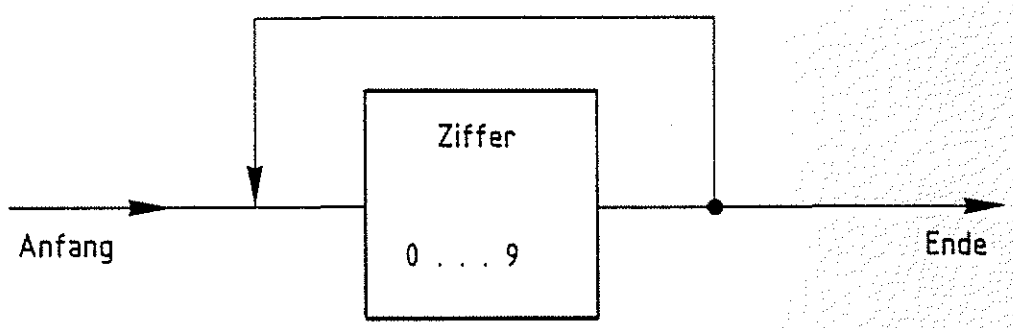
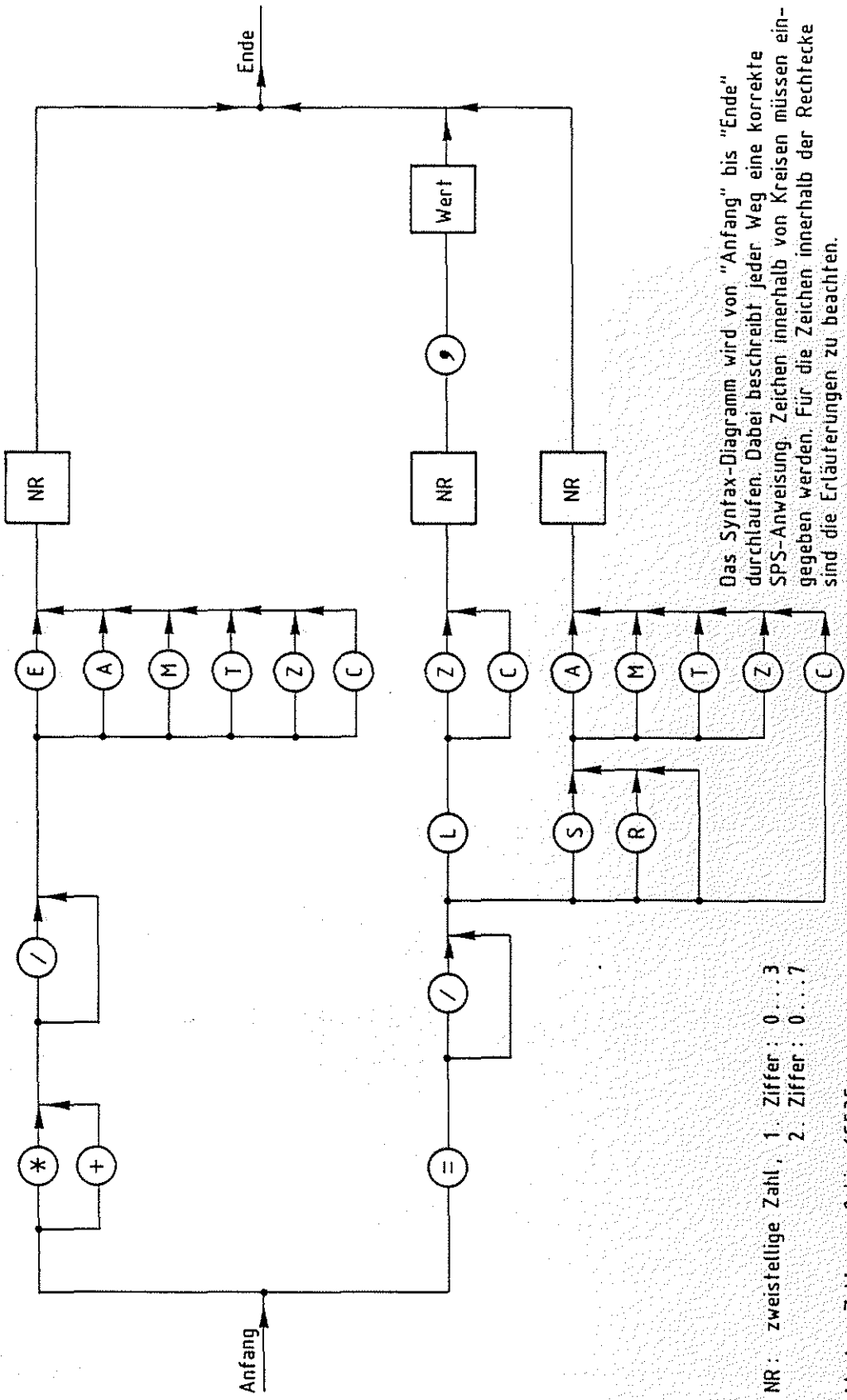


Bild 5: Syntax-Diagramm zur Bildung von Zahlen

Das Syntax-Diagramm besteht aus mehreren Linien, die Wege darstellen. Um eine Zahl zu bilden, muß das Diagramm von "Anfang" bis "Ende" durchlaufen werden. Dabei darf man sich nie gegen die Pfeilrichtung bewegen.
Der erste Weg führt auf ein Rechteck. Der Text im Rechteck besagt, daß zur Bildung einer Zahl eine Ziffer (0 ... 9) geschrieben werden muß. Nach dem Rechteck verzweigt der Weg. Man kann entweder direkt zum "Ende" gehen, oder man geht zurück zur linken Seite des Rechtecks. Im ersten Fall ist die Zahl gebildet. Sie besteht dann nur aus einer Ziffer. Im zweiten Fall kann man der ersten Ziffer eine weitere anfügen. Nachdem man eine Ziffer angefügt hat, besteht erneut die Möglichkeit sich zu entscheiden, ob eine weitere Ziffer angehängt werden soll oder nicht.

SPS-Programm / Syntax-Diagramm



Das Syntax-Diagramm wird von "Anfang" bis "Ende" durchlaufen. Dabei beschreibt jeder Weg eine korrekte SPS-Anweisung. Zeichen innerhalb von Kreisen müssen eingegeben werden. Für die Zeichen innerhalb der Rechtecke sind die Erläuterungen zu beachten.

NR: zweistellige Zahl, 1. Ziffer: 0...3
2. Ziffer: 0...7

Wert: Zahl von 0 bis 65535

Bild 6: Syntax-Diagramm für SPS-Anweisungen

SPS-Programm

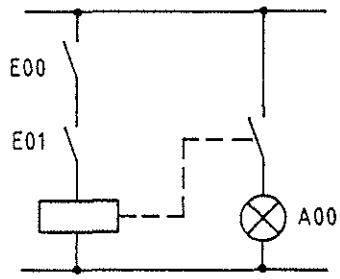
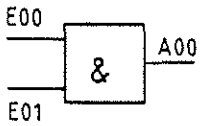
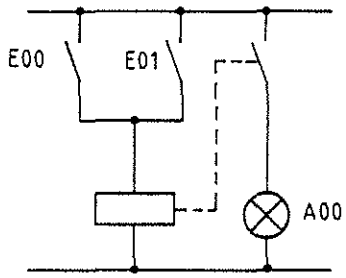
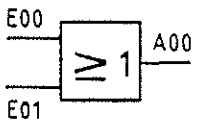
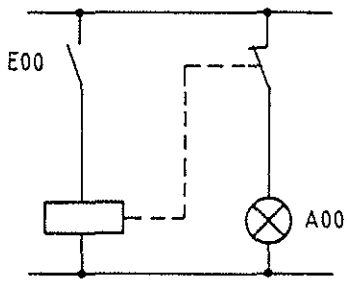
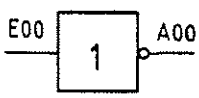
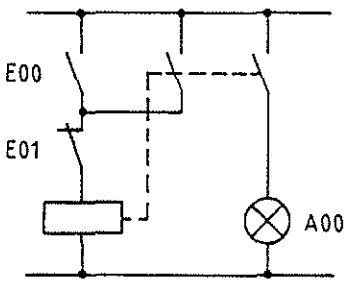
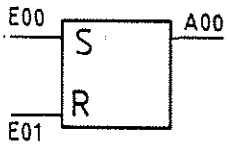
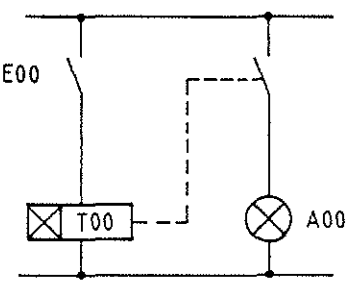
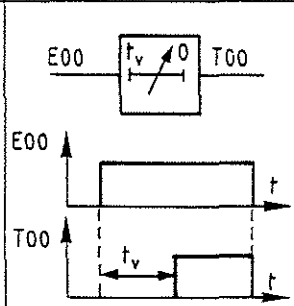
	Kontaktschaltung	Logikplan	Sprachliche Beschreibung	SPS-Programm
UND			(UND) E00 UND E01 GLEICH A00	* E00 * E01 = A00
ODER			(UND) E00 ODER E01 GLEICH A00	* E00 + E01 = A00
NICHT			(UND) NICHT E00 GLEICH A00 oder UND E00 GLEICH NICHT A00	*/E00 = A00 oder * E00 =/A00
SETZE/RÜCKSETZE			(UND) E00 UND NICHT E01 SETZE A00 (UND) E01 RÜCKSETZE A00	* E00 */ E01 = SA00 * E01 = RA00
ZEITWERK			(UND) E00 GLEICH T00 (UND) T00 GLEICH A00	* E00 = T00 * T00 = A00

Bild 7: Vergleich: Kontaktschaltung, Logikplan, SPS-Programm

4.3.3. Grundzustand der Operanden

Immer wenn das Programm sein Prompt "SPS>" ausgibt und auf die Eingabe eines Befehls wartet, nehmen die Operanden folgende Zustände ein:

Eingänge	entsprechend den Eingangssignalen
Ausgänge	logisch "0" (L-Pegel)
Merker	logisch "0"
Hardware-Timer	logisch "0"
Software-Timer	logisch "0"
Zähler	logisch "0"

Diese Zustände werden auch beim Start eines SPS-Programms eingenommen und können nur durch entsprechende Anweisungen bzw. äußere Signale verändert werden.

4.3.4. Programm-Beispiele

4.3.4.1. Blinklicht

Ausgang A00 blinkt:

Programm	Kommentar
*/M00=LZ00,10=LZ01,10	Wenn der Merker M00 nicht logisch "1" ist (alle Operanden sind beim Programm-Start logisch "0"), dann lade die Software-Timer Z00 und Z01 mit dem Wert 10 (Laufzeit 1 Sekunde).
*/A00=SZ00	Wenn Ausgang A00 nicht logisch "1" (Bedingung ist beim Programm-Start erfüllt), dann setze (starte) Timer Z00.
*Z00=SA00=SZ01=RZ00	Wenn Timer Z00 abgelaufen, dann setze Ausgang A00, setze (starte) Timer Z01 und setze Timer Z00 zurück.
*Z01=RA00=RZ01	Wenn Timer Z01 abgelaufen, dann setze Ausgang A00 und setze Timer Z01 zurück.

4.3.4.2. Zähler

Der Zähler zählt die Häufigkeit der "EIN"-Zustände von E00. Nach 5 mal "EIN" wird über A00 ein Signal ausgegeben. Dieses Signal (und der Zähler) kann mit einem "Ein"-Signal von E01 rückgesetzt werden.

Programm	Kommentar
*/M01=LC00,5=SM01	Wenn Merker M01 nicht logisch "1" ist (dies ist beim Programm-Start der Fall), dann lade den Zähler C00 mit dem Wert 5 und setze den Merker M01. Da M01 nun gesetzt ist, ist die Bedingung beim nächsten Programm-Durchlauf nicht erfüllt und der Zähler-Wert wird nicht durch den LADE-Befehl verändert.
*/M00*E00=C00=SM00	Wenn der Merker M00 nicht logisch "1" ist (beim Programm-Start erfüllt) und wenn der Eingang E00 logisch "1" ist, dann ziehe vom augenblicklichen C00-Zählerstand Eins ab und setze M00. Durch das Setzen des Merkers wird der Zähler gesperrt.
*/E00=RM00	Wenn E00 nicht mehr auf logisch "1" ist, dann setze M00 zurück und gebe so den Zähler frei.
*C00=SA00	Wenn der Zähler abgelaufen ist, dann setze den Ausgang A00 (Signal).
*E01=RM01=RA00	Wenn E01 logisch "1" ist, dann setze M01 (gebe Lade-Befehl frei) und A00 (lösche Signal) zurück.

4.3.5. Aufruf des SPS-Programms

Um mit dem SPS-Programm arbeiten zu können, muß erst die Monitor-Erweiterung MAT 85+ aufgerufen werden:

Nach dem Einschalten des Mikrocomputers und dem Betätigen der Leertaste (Space) meldet sich das Betriebsprogramm MAT 85. Nachdem es eine Liste aller zur Verfügung stehenden Kommandos ausgegeben hat, erscheint die "Bereit"-Meldung (Prompt) von MAT 85:

```
KMD >_
```

Durch Betätigen der Leertaste kann nun die Monitor-Erweiterung MAT 85+ aufgerufen werden. Die Erweiterung meldet sich mit dem Prompt:

```
KMD+>_
```

Zum Aufruf des SPS-Programms muß nun die Taste "S" und anschließend die "CR"-Taste (Carriage Return, Wagenrücklauf), gedrückt werden:

```
KMD+>SPS                          S [CR] eintippen,
                                  "PS" wird ergänzt

BFZ-SPS-PROGRAMM V2.1             Das Programm meldet
                                  sich, listet seine
  EDIT                            Kommandos auf
  GO
  HELP
  LIST
  NEW
  READ
  STEP
  TRACE
  WRITE
  QUIT

 SPS>_                            und ist bereit, Be-
                                  fehle entgegenzu-
                                  nehmen.
```

Wurde das SPS-Programm bereits zuvor aufgerufen, so meldet es sich mit:

```
BFZ-SPS-PROGRAMM V2.1 RESTART
```

Durch das Wort "RESTART" und ein akustisches Signal zeigt das Programm, daß es schon einmal aufgerufen wurde. Der Programmspeicher-Inhalt entspricht dem Zustand vor der Ausführung des letzten QUIT-Kommandos.

4.3.6. Kommando-Eingabe

Die Bereitschaft zur Annahme eines Kommandos zeigt das Programm durch den Ausdruck "SPS>" an.
Jedes der oben aufgelisteten Kommandos kann durch die Eingabe seines ersten Buchstabens und durch anschließendes Betätigen der Taste "CR" (Carriage Return, Wagenrücklauf) aufgerufen werden. Daraufhin druckt das Programm den vollständigen Kommandonamen und fordert eventuell zusätzliche Informationen an. Falsch eingegebene Zeichen können durch die Betätigung der Taste "DEL" (Delete, Löschen) oder "BS" (Backspace, Rückwärtsschritt) gelöscht werden. Soll das Kommando abgebrochen werden, muß die Taste "ESC" (Escape, Flucht) betätigt werden. Man wechselt dann automatisch vom SPS-Programm zum Betriebsprogramm MAT 85. Dieses quittiert die Eingabe von "ESC" durch ein akustisches Signal und fordert durch das Ausdrucken von "KMD >" ein neues Kommando an.

4.3.7. Bildschirm-Modus, Drucker-Modus

Das SPS-Programm unterscheidet zwischen Bildschirm-Modus und Drucker-Modus. Immer dann, wenn das Programm auf eine neue Eingabe wartet, kann durch gleichzeitiges Betätigen der Tasten "CONTROL" und "P" zwischen dem Bildschirm- und dem Drucker-Modus gewechselt werden. Im Drucker-Modus erfolgen alle Ausgaben gleichzeitig auf dem Bildschirm und auf dem Drucker

4.3.8. Bediener-Führung

Bei allen Eingaben prüft das Programm, ob die Eingabedaten dem notwendigen Format entsprechen. Ist dies nicht der Fall, wird der Bediener durch ein akustisches Signal auf seinen Fehler aufmerksam gemacht.

4.3.9. Beschreibung der Kommandos

4.3.9.1. Das HELP-Kommando

Mit dem HELP-Kommando lassen sich die Namen der zulässigen Kommandos in alphabetischer Reihenfolge ausdrucken.

Aufruf und Handhabung:

```
SPS > HELP                    H [CR] eintippen,
                              "ELP" wird ergänzt

(Kommando-Ausführung)

SPS>_                         nächstes Kommando
```

Zur Kommando-Ausführung:

- Nach dem Ausdrucken aller Kommandonamen erfolgt ein Rücksprung in die Kommando-Routine (SPS>).
- Zum Aufruf eines der Kommandos muß nur der 1. Buchstabe, gefolgt von der Taste [CR], eingegeben werden.
- Eingaben, die vor der Betätigung von [CR] erfolgen, können mit den Tasten [DEL] (Delete, löschen) und [BS] (Backspace, Rückwärtsschritt) gelöscht werden.

4.3.9.2. Das EDIT-Kommando

Mit dem Kommando EDIT wird ein Unterprogramm, der Editor, aufgerufen. Mit ihm können die einzelnen Programmschritte des SPS-Programms eingegeben werden. Außerdem ermöglicht er das Einfügen, Löschen, Verändern und Suchen einzelner Programmschritte. Der Editor überprüft alle Eingaben die der Bediener macht. Fehler werden durch ein akustisches Signal angezeigt und in einem Großteil der Fälle durch eine Meldung erläutert.

Aufruf:

```
SPS > EDIT
```

E [CR] eintippen,
"DIT" wird ergänzt

Steht noch kein Programm im Speicher, meldet der EDITOR:

```
ANF: <ENDE>
```

"ANF:" zeigt an, daß man sich am Programm-Anfang befindet. <ENDE> bedeutet, daß hier das Programm-Ende ist. Da kein Programm im Speicher ist, fallen Anfang und Ende zusammen. Wenn ein Programm im Speicher ist, wird statt <ENDE> die erste Anweisung angezeigt.

Zum Beispiel:

```
ANF: *E00
```

Die Ausdrücke, die man mit Hilfe des Editors eingeben kann, bestehen nur aus kleinen Teilausdrücken. So muß z.B. der Ausdruck "*E00*/E01+E02=SA00=LZ03,44" in die fünf Teilausdrücke "*E00", "*/E01", "+E02", "=SA00" und "=LZ03,44" zerlegt werden. Die Teilausdrücke, die einer Anweisung entsprechen, dürfen maximal ein Verknüpfungs-Symbol ("+" oder "*") bzw. ein Gleichheitszeichen enthalten. Beim Eingeben eines SPS-Programms muß die Eingabe eines solchen Teilausdrucks durch die Betätigung der [CR] - Taste abgeschlossen werden.

Um die Bedienung des Editors näher zu erläutern, soll das SPS-Programm aus Abschnitt 4.3.2.3. eingegeben werden:

*E00*E01=A00

Schirmbild	Eingabe, Kommentar
SPS > EDIT	E [CR] eintippen, "DIT" wird ergänzt
ANF: <ENDE> *E00	noch keine Anweisungen im Speicher. Eingabe: *E00 [CR]
<ENDE> *E01	Anfang nun nicht mehr gleich Ende, da bereits eine Anweisung (*E00) eingegeben wurde. Eingabe: *E01 [CR]
<ENDE> =A00	Eingabe der letzten Anweisung: =A00 [CR]
<ENDE>	Beenden der Programmeingabe durch Betätigung von [CR]
SPS > _	

Jede Eingabe muß durch die Betätigung von [CR] abgeschlossen werden. Um den Editor zu verlassen, muß [CR] als erstes Zeichen eingegeben werden. Wurden Eingaben noch nicht durch [CR] abgeschlossen, so ist die Korrektur von Tippfehlern durch Betätigen der Tasten [DEL], [BS] oder [←] möglich.

ANZEIGEN der Anweisungen:

Mit [SP] kann man zur nächsten Anweisung im Programmspeicher gehen, mit [-] kann man zur vorherigen Anweisung im Programmspeicher zurück gelangen:

Schirmbild	Eingabe, Kommentar
SPS > EDIT	E [CR] eintippen "DIT" wird ergänzt
ANF: *E00	Anzeige der ersten Anweisung. Eingabe: [SP]
*E01	Anzeige der nächsten Anweisung. Eingabe: [SP]
=A00	Anzeige der nächsten Anweisung. Eingabe: [SP]
< ENDE >	Ende des SPS-Programms. Eingabe: [-]
=A00	Anzeige der vorherigen Anweisung. Eingabe: [-]
*E01	Anzeige der vorherigen Anweisung. Eingabe: [-]
ANF: *E00	Anzeige der vorherigen Anweisung (Programm-Anfang)

Wird statt [SP] oder [-] aus Versehen [CR] betätigt, wird der Editor wieder verlassen.

ANZEIGEN von Programm-Anfang und -Ende:

Durch Drücken von | kann man an das Programm-Ende springen:

```
ANF: *E00
      <ENDE>
```

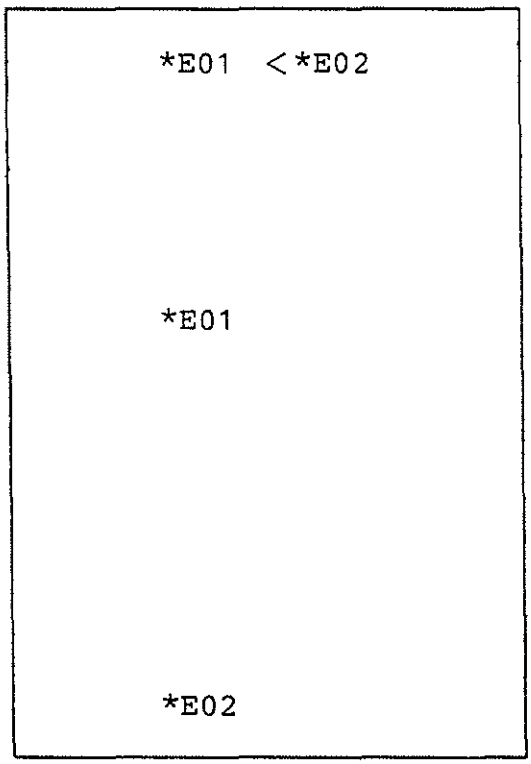
Eingabe: |

Durch Drücken von | kann man an den Programm-Anfang springen:

```
      <ENDE>
ANF: *E00
```

Eingabe: |

ANWEISUNGEN EINFÜGEN:

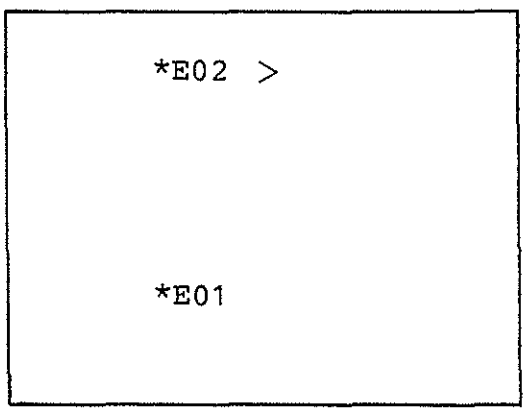


Eingabe: <*E02 CR
Füge vor der angezeigten Anweisung (*E01) die Anweisung *E02 ein.
"<" ist das Einfügungszeichen.

Es wird die gleiche Anweisung angezeigt wie vor der Einfügung.
Die neue Anweisung (*E02) steht vor der angezeigten Anweisung.
Eingabe: - (zeige die vorhergehende Anweisung an)

Diese Anweisung wurde eingefügt.

ANWEISUNGEN LÖSCHEN:



Eingabe: > CR
Lösche die angezeigte Anweisung.
">" ist das Symbol zum Löschen der angezeigten Anweisung.

Es wird die Anweisung angezeigt, die der gelöschten Anweisung folgt.

ÜBERSCHREIBEN falscher Anweisungen:

```
      *E01   *E02


      =A00




      *E02   *E01




      =A00


SPS > _
```

Eingabe: *E02 CR
(Überschreibe *E01 mit *E02)

Es wird die folgende Anweisung angezeigt.
Eingabe: -
(Zeige die vorhergehende Anweisung an)

Diese Anweisung (*E02) hat die Anweisung *E01 ersetzt.
Eingabe: *E01 CR
(Überschreibe *E02 mit *E01)

Es wird die folgende Anweisung angezeigt.
Eingabe: CR
(Beende den Editor)

Der Editor akzeptiert keine fehlerhaften Eingaben. Die Fehlerstelle wird vom Editor durch ein Fragezeichen markiert:

```
SPS > EDIT

ANF: *E00   E02




ANF: *E00    E02
             ?


SPS > _
```

E CR eintippen
"DIT" wird ergänzt

Anzeige der ersten Anweisung *E00

Versuch die Anweisung *E00 zu überschreiben:
Eingabe: E02 CR
(Eingabe fehlerhaft, da Verknüpfungssymbol fehlt)

Anzeige der alten Anweisung (*E00).
Die Fehlerstelle wird durch ein Fragezeichen markiert.
Eingabe: CR
(Beenden des Editors)

Die SUCHFUNKTION:

```
ANF: *E00   ?0

     *E01   ?0

     =A00   ?0

     =A00   ?0 NICHT GEFUNDEN
     =A00

ANF: *E00   ?E01

     *E01
```

Eingabe: ?0 CR
Suche nächste Anweisung, die eine Null enthält.

Die nächste Anweisung, die eine Null enthält, wird angezeigt.
Eingabe: ? CR
Suche das zuletzt gesuchte Zeichen (im Beispiel "0") erneut. Die Eingabe von "0" kann entfallen, da das Programm sich das Zeichen "gemerkt" hat. Das Zeichen wird auf dem Bildschirm ergänzt.

Die nächste Anweisung,die eine Null enthält, wird angezeigt.
Eingabe: ? CR
Suche das gesuchte Zeichen ("0") erneut.

Keine weiteren Anweisungen mit "0". Es wird die gleiche Anweisung angezeigt wie vor dem Suchbefehl.
Eingabe: | (Sprung an den Programmanfang)

Eingabe: ?E01 CR
Suche nächste Anweisung mit dem Operanden E01.

Anzeige der nächsten Anweisung mit dem Operanden E01

Die folgende Tabelle zeigt eine Zusammenfassung aller EDIT-Kommandos:

Eingabe	Wirkung
xxxx [CR]	ÜBERSCHREIBEN der angezeigten Programmzeile mit xxxx. Wird <ENDE> angezeigt, so wird xxxx vor <ENDE> eingefügt
> [CR]	LÖSCHEN der angezeigten Programmzeile
< xxxx [CR]	EINFÜGEN von xxxx vor der angezeigten Programmzeile
[SP]	ANZEIGEN der nächsten Programmzeile
[—]	ANZEIGEN der vorhergehenden Programmzeile
[↑]	ANZEIGEN von <ENDE>. Vorbereitung zum Anfügen von Programmzeilen
[↓]	ANZEIGEN der ersten Programmzeile
[?] y [CR]	SUCHEN von y in den folgenden Programmzeilen
[CR]	BEENDEN des Edit-Modus

xxxx = vollständige SPS-Anweisung
y = zu suchendes Zeichen

Tabelle 2: EDIT-Kommandos

Softwarepaket SP 1

Name:

SPS-Programm / EDIT-Kommando

Datum:

E9 SPS

Geben Sie das folgende SPS-Programm ein:

```
*E00=M00
*M00=A00
```

Fügen Sie die Anweisungen

```
*M00
=/M00
```

in der Programm-Mitte ein.

Ändern Sie die zweite Anweisung (=M00) in:

```
=/A00
```

Löschen Sie die Anweisungen

```
*M00
=/M00
```

Gehen Sie mit [↑] an den Programm-Anfang und suchen Sie die nächste Anweisung, die einen Merker enthält.

Löschen Sie die Anweisungen

```
*M00
=A00
```

Das Programm sollte nun aus folgenden Anweisungen bestehen:

```
*E00
=/A00
```

4.3.9.3. Das GO-Kommando

Durch das GO-Kommando kann ein im Programm-Speicher befindliches SPS-Programm gestartet werden. Zu Beginn haben alle Ausgänge, Merker, Timer und Zähler den Zustand logisch "0".

Aufruf:

```
   SPS > GO


*** SPS-PROGRAMM GESTARTET ***
```

G [CR] eintippen
"O" wird ergänzt

Ein laufendes SPS-Programm kann durch Drücken einer beliebigen Taste der Tastatur (außer CONTROL, BREAK und SHIFT) abgebrochen werden, wenn die im Anhang beschriebene Schaltungsergänzung durchgeführt wurde.

Wenn die erste Anweisung im SPS-Programm keine Bedingung ist, oder wenn die letzte Anweisung im SPS-Programm keine Zuweisung ist, wird die Fehlermeldung

```
*** PROGRAMM-FEHLER ***
```

ausgegeben. Das Programm meldet sich dann mit seinem Prompt "SPS " und ist bereit, ein neues Kommando entgegenzunehmen.

Arbeitsblatt | BFZ / MFA 7.2. - 91

Softwarepaket SP 1

Name:

SPS-Programm / GO-Kommando

Datum:

G2 SPS

Geben Sie folgendes Programm ein:

*E00=/A00

Starten Sie das Programm und protokollieren Sie den Signalzustand des Ausgangssignals A00 in Abhängigkeit vom Eingangsignal E00:

Eingang E00	Ausgang A00
logisch "0"	
logisch "1"	

Was bewirkt dieses Programm ?

Antwort: _____

Verlassen Sie das SPS-Anwenderprogramm

a) bei erfolgter Umrüstung, wie im Anhang beschrieben, durch die Betätigung einer Taste der Tastatur.

b) durch Betätigung des RESET-Tasters.

Zu a) In welchem Programmteil befinden Sie sich ?

Antwort: _____

Zu b) In welchem Programmteil befinden sie sich ?

Antwort: _____

4.3.9.4. Das LIST-Kommando

Mit dem LIST-Kommando kann das im Programm-Speicher befindliche SPS-Programm aufgelistet werden.

Aufruf:

SPS > LIST	L [CR] eintippen "IST" wird ergänzt
(Auflistung des Programms)	
SPS > _	Das Programm ist bereit, weitere Kommandos entgegenzunehmen.

Bei der Auflistung des Programms steht jeder SPS-Ausdruck in einer eigenen Zeile. Wenn der Bildschirm voll ist, erscheint die Meldung "==> SPACE". Die Auflistung wird fortgesetzt, wenn die Space-Taste betätigt wird.
Wenn der Drucker eingeschaltet ist, erfolgt ein kontinuierlicher Ausdruck.

4.3.9.5. Das NEW-Kommando

Mit dem NEW-Kommando kann ein im Programmspeicher befindliches SPS-Anwender-Programm komplett gelöscht werden.

Aufruf:

```
SPS > NEW
```

N CR eintippen
"EW" wird ergänzt

Anzeige:

```
SPS > _
```

Das Programm ist bereit, weitere Kommandos entgegenzunehmen.

4.3.9.6. Das QUIT-Kommando

Mit dem QUIT-Kommando kann das SPS-Programm verlassen werden. Man gelangt dann automatisch zur Monitorerweiterung MAT 85+ zurück.

Aufruf:

```
SPS> QUIT

KMD+> _
```

Q [CR] eintippen
"UIT" wird ergänzt

Die Monitorerweiterung MAT 85+ meldet sich und ist bereit, Kommandos entgegenzu nehmen.

Wird SPS nun erneut aufgerufen, so meldet es sich mit:

```
BFZ-SPS-PROGRAMM V2.1 RESTART
```

Durch den Zusatz "RESTART" und ein akustisches Signal zeigt das Programm an, daß es schon einmal aufgerufen wurde. Der Programmspeicher-Inhalt ist gegenüber dem letzten Aufruf unverändert, wenn er nicht durch andere Kommandos (z.B. "ASSEMBLER") verändert wurde.

4.3.9.7. Das READ-Kommando

Mit dem READ-Kommando kann ein SPS-Programm, das zuvor mit dem WRITE-Kommando auf Kassette abgespeichert wurde, in den Programmspeicher geladen werden. Dazu wird das Kassetten-Interface BFZ/MFA 4.4. benötigt.

Aufruf:

```
SPS > READ

 SPACE, DANN BAND EINSCHALTEN
```

R [CR] eintippen
"EAD" wird ergänzt

Um das Programm von einer Kassette in den Programmspeicher zu laden, muß erst die Space-Taste betätigt werden. Anschließend ist der Recorder einzuschalten.

Nach erfolgreichem Einlesen meldet sich das Programm mit "SPS>" und ist bereit, weitere Befehle entgegenzunehmen.
Wenn ein Lade-Fehler auftritt, wird eine entsprechende Fehlermeldung ausgegeben. Das Programm meldet sich anschließend mit "SPS>" und ist bereit, weitere Befehle entgegenzunehmen. Trat ein Lesefehler auf, so ist der Programmspeicher leer.

4.3.9.8. Das STEP-Kommando

Das SPS-Programm kann schrittweise abgearbeitet werden. Dazu muß vor dem Starten des Programms mit dem GO-Kommando der Einzelschritt-Modus mit dem STEP-Kommando eingeschaltet werden. Das Abschalten des Einzelschritt-Betriebes erfolgt ebenfalls mit dem STEP-Kommando.

Aufruf:

```
SPS > STEP

 EIN/AUS = X









SPS > _
```

S [CR] eintippen
"TEP" wird ergänzt

X = aktueller Modus

X: A = Aus
E = Ein

Mögliche Eingaben:
A = STEP-Modus aus
E = STEP-Modus ein
[CR] oder [SP] = STEP-Modus unverändert

Das Programm ist bereit, weitere Kommandos entgegenzunehmen.

Das SPS-Programm kann bei eingeschaltetem STEP-Modus, wie im Abschnitt 4.3.9.3. beschrieben, mit dem GO-Kommando gestartet werden. Es wird dann jede Anweisung vor der Ausführung angezeigt. Die Anweisung wird erst ausgeführt, wenn die Space-Taste betätigt wird. In diesem Fall wird das Ergebnis einer Anweisung (logisch Null oder logisch Eins) angezeigt. Bei SETZ- und RÜCKSETZ-Anweisungen, sowie bei den Anweisungen "=Zxx", "=Txx", "=L..." und "=Cxx", entspricht das angezeigte Ergebnis nicht dem logischen Zustand, sondern gibt an, ob die Anweisung ausgeführt wurde <1> oder nicht <0>. Bei der Anzeige des Ergebnisses, das innerhalb von spitzen Klammern steht, wird eine eventuelle Negation ("/") berücksichtigt. Betätigt man statt der Space-Taste die CR-Taste, so wird das Programm abgebrochen und es erscheint "SPS>_".
Der STEP-Modus kann entweder durch gezieltes Ausschalten, wie oben beschrieben, oder durch Verlassen des SPS-Programms ausgeschaltet werden.

Beispiel für die Anzeige im Einzelschritt-Modus bei dem Programm

```
*E00
*E01
=A00
```

```
*** SPS-PROGRAMM GESTARTET ***

*E00<0>




*E01<1>




=A00<0>
```

*E00 wird angezeigt. Die Anweisung, den Eingabekanal E00 zu lesen, wird erst ausgeführt, wenn SP betätigt wird. Das Ergebnis, hier "0", wird dann ergänzt.

*E01 wird angezeigt. Die Anweisung, den Eingabekanal E01 zu lesen, wird erst ausgeführt, wenn SP betätigt wird. Das Ergebnis, hier "1", wird dann ergänzt.

=A00 wird angezeigt. Die Anweisung, das Verknüpfungsergebnis "E00*E01" dem Ausgang A00 zuzuweisen, wird erst ausgeführt, wenn SP betätigt wird. Das Ergebnis, hier "0", wird dann ergänzt.

Softwarepaket SP 1

Name:

SPS-Programm / STEP-Kommando

Datum:

S3 SPS

Geben Sie folgendes Programm ein:

*E00=/A00

- Stellen Sie den Schalter für E00 auf "AUS" (LED aus).
- Schalten Sie den STEP-Modus ein und starten Sie das Programm.
- Füllen sie die nachstehende Tabelle für drei Schritte aus:

Schritt	Anweisung	Ergebnis
1		
2		
3		

- Schalten Sie nun E00 auf "EIN"
- Protokollieren Sie die nächsten Schritte:

Schritt	Anweisung	Ergebnis
4		
5		
6		

4.3.9.9. Das TRACE-Kommando

Das TRACE-Kommando ähnelt dem STEP-Kommando (siehe Abschnitt 4.3.9.7.). Bei der Bearbeitung eines SPS-Programms in der Betriebsart "TRACE" hinterläßt jede Anweisung eine Spur (Trace=Spur), indem sie zusammen mit dem jeweiligen Ergebnis angezeigt wird. Bei SETZ- und RÜCKSETZ-Anweisungen, sowie bei den Anweisungen "=Zxx", "=Txx", "...=L" und "=Cxx", entspricht das angezeigte Ergebnis nicht dem logischen Zustand, sondern gibt an, ob die Anweisung ausgeführt wurde <1> oder nicht <0>. Bei der Anzeige der Ergebnisse sind eventuelle Negationen ("/") berücksichtigt. Der Programmlauf wird unterbrochen, wenn der Bildschirm voll ist, oder wenn das SPS-Programm einmal vollständig durchlaufen wurde. In diesem Fall wird "==> SPACE" angezeigt. Durch Druck auf die Space-Taste wird der Programm-Ablauf fortgesetzt. Wird statt der SPACE-Taste CR betätigt, wird das Programm abgebrochen. Wenn der Drucker angeschlossen ist, erfolgt ein kontinuierlicher Programmablauf.
Das Programm kann, wenn die im Anhang beschriebene Schaltungserweiterung durchgeführt wurde, durch die Betätigung einer beliebigen Taste (außer CONTROL, BREAK und SHIFT) abgebrochen werden.

Aufruf:

```
SPS > TRACE

 EIN/AUS = X









 SPS > _
```

T [CR] eintippen
"RACE" wird ergänzt

X = aktueller Modus

X: A = Aus
E = Ein

Mögliche Eingaben:
A = Aus
E = Ein
[CR] oder [SP] = TRACE-Modus unverändert.

Das Programm ist bereit, weitere Kommandos entgegenzunehmen.

Beispiel für die Ausgabe im TRACE-Modus für das Programm

```
*E00
*E01
=A00
```

```
*** SPS-PROGRAM GESTARTET ***

*E00<0>


*E01<1>


=A00<0>




<ENDE>



==> SPACE
```

Anzeige der Anweisung "Lese Eingabe-Kanal E00" mit Ergebnis "0"

Anzeige der Anweisung "Lese Eingabe-Kanal E01" mit Ergebnis "1"

Anzeige der Anweisung "Weise das Verknüpfungsergebnis dem Ausgang A00 zu" mit Ergebnis "0"

Ein vollständiger Durchlauf ist abgeschlossen.

Nur bei ausgeschaltetem Drucker: Weiterlauf erst bei Betätigung der Space-Taste

CR bricht das Programm ab.

Softwarepaket SP 1

Name:

SPS-Programm / TRACE-Kommando

Datum:

T3 SPS

Geben Sie folgendes Programm ein:

*E00=/A00

- Stellen Sie den Schalter für E00 auf "AUS" (LED aus).
- Schalten Sie den TRACE-Modus ein und starten Sie das Programm.
- Füllen sie die nachstehende Tabelle aus:

Zeile	Anweisung	Ergebnis
1		
2		
3		

- Schalten Sie nun E00 auf "EIN".
- Starten Sie den nächsten Durchlauf durch Betätigen der [SP] - Taste.
- Vervollständigen Sie die folgende Tabelle:

Zeile	Anweisung	Ergebnis
4		
5		
6		

4.3.9.10. Das WRITE-Kommando

Mit dem WRITE-Kommando kann das im Programmspeicher befindliche Programm auf einer Magnetband-Kassette abgespeichert werden. Dazu ist das Kassetteninterface BFZ/MFA 4.4. erforderlich.

Aufruf:

```
SPS > WRITE

 BAND EINSCHALTEN, DANN SPACE


SPS > _
```

W [CR] eintippen
"RITE" wird ergänzt

Erst den Recorder einschalten, dann [SP] drücken

Nach dem Abspeichern des Programms ist die SPS bereit, neue Kommandos entgegenzunehmen.

4.4. Das BFZ-Steuer-BASIC

Zum Betrieb des BFZ-Steuer-BASIC ist zusätzlich zur RAM-Mindestbestückung für MAT 85+ RAM-Speicher ab Adresse 6000 notwendig. Die Obergrenze des RAM-Speichers ist beliebig.
Um laufende BASIC-Programme durch Betätigen einer Taste der Tastatur (außer CONTROL, BREAK und SHIFT) abbrechen zu können, ist die im Anhang beschriebene Schaltungsergänzung notwendig.
Es wird empfohlen, die gezeigten Beispiele jeweils unmittelbar mit Hilfe des BFZ/MFA-Mikrocomputers nachzuvollziehen.

4.4.1. Zulässiger Zahlenbereich, interne Zahlendarstellung

Das BFZ-Steuer-BASIC arbeitet nur mit ganzen Zahlen (Integer) ohne Nachkommastellen. Es sind positive und negative Werte erlaubt. Dabei gelten die Grenzen:

maximaler positiver Wert: 32767 maximaler negativer Wert: -32768

Jeder Zahlenwert wird intern in zwei Bytes gespeichert. Diese zwei Bytes lassen einen Zahlenbereich von 0 bis 65535 zu. Da das BFZ-Steuer-BASIC mit positiven und negativen Zahlen arbeitet, muß dieser Bereich geteilt werden. Eine Hälfte repräsentiert den positiven Zahlenbereich, die andere repräsentiert den negativen Zahlenbereich.

BASIC / Zahlenbereich, interne Zahlendarstellung

Die getroffene Einteilung teilt den Zahlenbereich wie folgt auf:

dezimal	hexadezimal	binär
0	0000	0000000000000000
1	0001	0000000000000001
2	0002	0000000000000010
3	0003	0000000000000011
.	.	.
.	.	.
.	.	.
32765	7FFD	0111111111111101
32766	7FFE	0111111111111110
32767	7FFF	0111111111111111
-32768	8000	1000000000000000
-32767	8001	1000000000000001
-32766	8002	1000000000000010
-32765	8003	1000000000000011
.	.	.
.	.	.
.	.	.
-3	FFFD	1111111111111101
-2	FFFE	1111111111111110
-1	FFFF	1111111111111111

Aufgrund dieser Definition ergibt die Umwandlung der Hexadezimalzahl FFFF in eine Dezimalzahl nicht 65535 sondern -1. Grundsätzlich gilt: bei allen negativen Dezimalzahlen ist das höchstwertige Bit in der internen Darstellung gleich Eins.

Diese interne Zahlendarstellung hat zur Folge, daß bei den Befehlen PEEK und POKE alle Speicheradressen oberhalb 7FFF als negative Dezimalzahlen angegeben werden müssen. Es wird deshalb die hexadezimale Schreibweise "DEC(.....)" empfohlen.

Bei den logischen Befehlen AND, OR und NOT ist die interne Zahlendarstellung ebenso zu berücksichtigen:

So ist z.B. NOT(0) = -1 (NOT (0000) = FFFF)
oder NOT(3) = -4 (NOT (0003) = FFFC)

Die logischen Befehle AND, OR und NOT finden ihre Anwendung allerdings nicht in mathematischen Ausdrücken, sondern werden zur Bitmanipulation bei Steuerungsaufgaben (in Zusammenhang mit den INP- und OUT-Befehlen) benutzt.

1. Beispiel:

Es soll geprüft werden, ob der Schalter B0 der Eingabe-Baugruppe 02 betätigt ist. Alle anderen Schalter sollen dabei unberücksichtigt bleiben:

```
10 I=INP(2): REM WERT VON EINGABE-BAUGRUPPE EINLESEN
20 A=I AND 1: REM DIE BITS B1 BIS B15 WERDEN AUF NULL GESETZT
30 IF A=0 THEN GOTO 10: REM WENN ALLE BITS AUF NULL, DANN GOTO 10
40 PRINT "SCHALTER BETAETIGT": REM SONST DRUCKE TEXT
```

2. Beispiel:

Die Daten am Eingabe-Port 01 sollen negiert am Ausgabe-Port 02 ausgegeben werden:

```
10 I=INP(1): REM LESE EINGABE-WERT
20 N=NOT(I): REM NEGIERE EINGABE-WERT
30 REM ES KOENNEN NUR WERTE VON 0 BIS 255 EINGELESEN WERDEN.
40 REM DAHER SIND NACH DER NEGATION DIE BITS B8 BIS B15,
50 REM DIE VORHER NULL WAREN, AUF EINS GESETZT.
60 REM ES KOENNEN NUR WERTE VON 0 BIS 255 AUSGEGEBEN WERDEN.
70 REM DAHER MUESSEN DIE BITS B8 BIS B15 AUF NULL GESETZT WERDEN:
80 A=N AND DEC(00FF): REM BITS B8 BIS B15 AUF NULL
90 OUT 2,A: REM WERT AUSGEBEN
```

4.4.2. Zulässige Variablen-Namen

Das BFZ-Steuer-BASIC kennt 26 einfache Variablen. Diese haben die Namen: A, B, C, ... , X, Y, Z.

Zusätzlich gibt es eine Feldvariable: @
Die Anzahl der möglichen Feldelemente wird durch den zur Verfügung stehenden Programmspeicher bestimmt, da der ungenutzte Programmspeicherplatz für die Speicherung der einzelnen Elemente benutzt wird.

Beispiel für die Anwendung der Feldvariablen:

```
10 FOR M=1 TO 12
20 PRINT "ANZAHL DER KURSTEILNEHMER IM MONAT";M
30 INPUT @(M)
40 NEXT M
50 PRINT "BITTE GEBEN SIE EINE MONATS-ZAHL (1-12) EIN:"
60 INPUT M
70 PRINT "ANZAHL DER KURSTEILNEHMER IM MONAT";M;"=";@(M)
80 GOTO 50
```

Das Programm-Beispiel "erfragt" für die Monate Januar bis Dezember die Anzahl der Kursteilnehmer. Dabei dient die Variable M als Index. Wenn die Anzahl für den Monat Januar erfragt wird, hat M den Wert 1. Bei der Frage nach der Anzahl für den Februar hat M den Wert 2 usw. Der Eingabe-Wert für Januar wird im ersten Element der Feldvariablen @ gespeichert (@(1)), der Wert für Februar wird im zweiten Element der Feldvariablen gespeichert (@(2)). Wenn alle Werte eingegeben sind, können die einzelnen Werte durch Angabe der Monats-Nummer abgefragt werden.

Eine Feldvariable kann mit einem Regal verglichen werden. Die einzelnen Elemente entsprechen dann den einzelnen Fächern. Auf den Fachinhalt kann durch Angabe des Regalnamens "@" und der Fachnummer (Index) zugegriffen werden.

Innerhalb der Klammern darf beim BFZ-Steuer-BASIC nur ein Index-Wert stehen.

4.4.3. Die Eingabe von Befehlen

BEFEHLSEINGABE:

Grundsätzlich gilt: Befehle, denen keine Nummer vorangestellt ist, werden sofort ausgeführt (Direkt-Modus).
Befehle, denen eine Nummer vorangestellt ist, werden in den Programmspeicher übernommen. Die Befehle im Programmspeicher werden nach den vorangestellten Nummern geordnet. Als Zeilennummer sind Werte von 1 bis 32767 erlaubt. Die Zeilennummer 0 wird ignoriert, d. h. der Mikrocomputer behandelt diese Zeile, als ob sie keine Zeilennummer hätte.

> Jede Eingabezeile muß mit [CR] abgeschlossen werden!

> Eine Programmzeile darf mehrere Befehle enthalten. Diese sind durch einen Doppelpunkt zu trennen:
>
> FOR I=1 TO 10 : PRINT I : NEXT I

Die maximale Länge einer Zeile beträgt 80 Zeichen.

Um die Lesbarkeit zu erhöhen, können beliebig viele Leerzeichen in den Programmtext eingefügt werden. Leerzeichen am Zeilenanfang (nach der Zeilennummer) werden nicht in den Programmspeicher übernommen.

KORREKTUR VON EINGABEFEHLERN:

Eingabefehler können, wenn die Eingabe noch nicht mit [CR] abgeschlossen wurde, mit den Tasten [DEL], [BS] und [←—] korrigiert werden.

LÖSCHEN VON PROGRAMMZEILEN:

Komplette BASIC-Programmzeilen können gelöscht werden, indem man die Nummer der zu löschenden Zeile eingibt:

20 [CR] (Löscht Zeile 20)

DRUCKER-BETRIEB:

Immer wenn das BASIC eine Eingabe erwartet, kann durch gleichzeitiges Betätigen der Tasten "CONTROL" und "P" der Drucker ein- bzw. ausgeschaltet werden.

4.4.4. Aufruf des BFZ-Steuer-BASIC

Um das BASIC starten zu können, muß erst die Monitor-Erweiterung MAT 85+ aufgerufen werden:

Nach dem Einschalten des Mikrocomputers und dem Drücken der Leertaste (Space) meldet sich das Betriebsprogramm MAT 85. Nachdem es eine Liste aller zur Verfügung stehenden Kommandos ausgegeben hat, erscheint die "Bereit"-Meldung (Prompt) von MAT 85:

```
KMD >_
```

Durch Betätigen der Leertaste kann nun die Monitor-Erweiterung MAT 85+ aufgerufen werden. Die Erweiterung meldet sich mit dem Prompt:

```
KMD+>_
```

Zum Aufruf des BASIC muß nun die Taste "B", gefolgt von der "CR"-Taste (Carriage Return, Wagenrücklauf), gedrückt werden:

```
KMD+ > BASIC                    B [CR] eintippen
                                "ASIC" wird ergänzt

BFZ-STEUER-BASIC V2.4           Das Programm meldet sich.

READY                           Es ist bereit,
>_                              Befehle entgegenzunehmen.
```

Durch die Ausgabe von ">_" zeigt der BASIC-Interpreter an, daß er bereit ist, Befehle entgegenzunehmen.

Handelt es sich nicht um den ersten Aufruf des BASIC-Interpreters, so meldet er sich mit:

```
BFZ-STEUER-BASIC V2.4 RESTART
```

Durch den Zusatz "RESTART" und ein akustisches Signal zeigt der BASIC-Interpreter an, daß es sich nicht um den ersten Aufruf handelt. Der Programmspeicher-Inhalt ist gegenüber dem letzten Aufruf unverändert, wenn er nicht durch andere Befehle verändert wurde.

Geben Sie folgende Befehle ein:

PRINT "AAA" CR
PRINT "BBB" CR

Der Rechner führt die Befehle sofort aus (er druckt "AAA" bzw. "BBB"), da den Befehlen keine Zeilen-Nummer vorangestellt ist. Man spricht hier vom Direkt-Modus.

Geben Sie nun die gleichen Befehle erneut ein. Stellen Sie diesmal die angegebene Zeilennummer vor die Befehle:

20 PRINT "BBB" CR
10 PRINT "AAA" CR

Der Rechner führt die Befehle diesmal nicht sofort aus, sondern speichert sie im Programm-Speicher. Der Programmspeicher-Inhalt kann mit dem Befehl

LIST CR

ausgedruckt werden. Lassen Sie sich den Programmspeicher-Inhalt ausdrucken und achten Sie auf die Reihenfolge der Zeilen. Der Rechner hat die Zeilen nach den einzelnen Zeilennummern sortiert!

Das Programm im Speicher kann mit dem Befehl

RUN CR

gestartet werden. Wenn Sie das Programm starten, werden die Buchstabenfolgen "AAA" und "BBB" ausgedruckt.

Löschen Sie nun Zeile 20, indem Sie

20 CR

eingeben. Kontrollieren Sie das Ergebnis mit dem LIST-Befehl.

Man kann mehrere Befehle in einer Zeile zusammenfassen, wenn man sie durch einen Doppelpunkt trennt. Geben Sie nun Zeile 10 neu ein ohne die alte Zeile vorher zu löschen:

10 PRINT "AAA" : PRINT "BBB" CR

Drucken Sie den Programmspeicher-Inhalt mit dem LIST-Befehl aus. Die alte Zeile 10 wurde durch die neue Zeile 10 überschrieben!

Starten Sie das Programm mit dem RUN-Befehl. Der Rechner gibt wieder die Buchstabenfolgen "AAA" und "BBB" aus.

4.4.5. Der Befehlssatz des BFZ-Steuer-BASICs

Das BFZ-Steuer-BASIC "kennt" folgende Befehle und Befehlssymbole:

ABS	AND	CLS	DATA
DEC	END	FOR	FREE
GOSUB	GOTO	IF	INP
INPUT	LET	LIST	LOAD
LPOFF	LPON	NEW	NEXT
NOT	OUT	OR	PEEK
POKE	PRINT	QUIT	READ
REM	RESTORE	RETURN	RND
RUN	SAVE	STEP	STOFF
STON	STOP	THEN	TO
TROFF	TRON	USR	WAIT
+	-	*	/
>=	<>	>	=
<=	<	$	#

4.4.5.1. Der ABS-Befehl

Der ABS-Befehl dient zur Berechnung des Absolutwertes.

Beispiele:

```
X=ABS(-4)
```

Weise der Variablen X den Absolutwert (Betrag) von -4 zu.

```
Y=ABS(Z)
```

Weise der Variablen Y den Absolutwert der Variablen Z zu.

```
Z=ABS(3*X+1)
```

Weise der Variablen Z den Absolutwert des Ausdrucks 3*X+1 zu.

```
PRINT ABS(22+Z/Y*X)
```

Drucke den Absolutwert des Ausdrucks 22+Z/Y*X.

ACHTUNG:
Es gilt: Punkt- vor Strichrechnung



4.4.5.2. Der AND-Befehl

Mit dem AND-Befehl kann eine logische UND-Verknüpfung durchgeführt werden. Bitte beachten Sie hierzu den Abschnitt 4.4.1.

Beispiele:

```
X = 11 AND 88
```

Weise der Variablen X das Ergebnis der UND-Verknüpfung der Werte 11 und 88 zu.

```
PRINT 123 AND 55
```

Drucke das Ergebnis der UND-Verknüpfung der Werte 123 und 55.

```
IF (X<33) AND (Y=7) THEN GOTO 123
```

Wenn der Wert von X kleiner als 33 ist und wenn der Wert von Y gleich 7 ist, dann setze die Programmausführung mit der Zeile 123 fort.

Bitte beachten Sie, daß die VERGLEICHSAUSDRÜCKE "X<33" und "Y=7" im BFZ-Steuer-Basic IN KLAMMERN gesetzt werden müssen.

4.4.5.3. Der CLS-Befehl

Mit dem CLS-Befehl kann der Bildschirm gelöscht werden.

Beispiel:

```
CLS
```

Lösche den Bildschirm.
(Clear Screen)

4.4.5.4. Der DATA-Befehl

Der DATA-Befehl ermöglicht das Ablegen von Daten im Programm. Diese Daten können mit dem READ-Befehl gelesen werden. Der DATA-Befehl darf nur am Anfang einer Programmzeile stehen. Datentabellen (DATA-Zeilen) können überall im Programm stehen. Jede Datenzeile muß mit dem Befehl "DATA" beginnen. Eine DATA-Zeile wird bei der Programm-Abarbeitung übersprungen. Daraus folgt, daß einem DATA-Befehl in der gleichen Zeile kein anderer Befehl (außer REM (Kommentar)) folgen darf. Siehe auch Abschnitt 4.4.5.28. (READ) und Abschnitt 4.4.5.30. (RESTORE).

Beispiele:

```
DATA 1,2,3,4,5,6
```

Lege die Werte 1,2,3,4,5,6 als Daten im Programm ab.

```
DATA 5,7 : REM DIES SIND DATEN
```

Lege die Werte 5 und 7 als Daten im Programm ab.
Zusätzlich Kommentar:
"Dies sind Daten".

Beispiels-Programm:

```
10 DATA 11,22,33,44,55

20 FOR I=1 TO 5
30 READ D


40 PRINT D


50 NEXT I
```

In der Zeile 10 werden 5 Datenwerte abgelegt.

Mit der READ-Anweisung in Zeile 30 wird je ein Wert gelesen.
Der gelesene Wert wird mit der PRINT-Anweisung in Zeile 40 gedruckt.
Die beiden Anweisungen "READ" und "PRINT" werden durch die FOR-NEXT-Schleife je fünf mal abgearbeitet.
Dadurch werden alle Daten gelesen.

4.4.5.5. Der DEC-Befehl

Mit dem DEC-Befehl kann eine Hexadezimal-Konstante in einen Dezimal-Wert gewandelt werden.

Beispiele:

```
PRINT DEC(FF)
```

Wandle den Hexadezimal-Wert FF in einen Dezimal-Wert und drucke ihn.

```
X = DEC(3C00)+22
```

Wandle den Hexadezimal-Wert 3C00 in den entsprechenden Dezimal-Wert. Addiere 22 und weise das Ergebnis der Variablen X zu.

Hexadezimal-Konstanten können überall dort verwendet werden, wo auch Dezimal-Konstante erlaubt sind. Die Hexadezimal-Konstanten müssen in Klammern eingeschlossen sein und vor den Klammern muß das Befehls-Wort "DEC" stehen.

4.4.5.6. Der END-Befehl

Dieser Befehl kennzeichnet das logische Programmende. Er beendet die Programm-Ausführung. Das logische Programmende muß nicht mit dem tatsächlichen (physischen) Programmende übereinstimmen. Stimmen logisches und physisches Programmende überein, so kann der Befehl "END" entfallen.

Beispiele:

```
END
```

Beende die Programmausführung.

```
IF X=33 THEN END
```

Beende die Programmausführung, wenn die Variable X den Wert 33 hat.

4.4.5.7. Der FOR-Befehl

Der FOR-Befehl bildet zusammen mit dem NEXT-Befehl eine Programmschleife.

Allgemeine Form:

```
FOR Schleifenvarible = Anfangswert TO Endwert STEP Schrittweite
```

Beim ersten Schleifendurchlauf ist der Wert der Schleifenvariablen gleich dem Anfangswert. Nach jedem Durchlauf wird die Schrittweite zum aktuellen Wert der Schleifenvariablen addiert. Die Schrittweite kann mit der STEP-Anweisung festgelegt werden. Ist sie gleich Eins, so kann der STEP-Befehl entfallen. Wenn das Additionsergebnis von Schleifenvariable und Schrittweite größer als der Endwert ist, wird die Schleife abgebrochen.
Da die Addition der Schrittweite und der Vergleich mit dem Endwert erst nach einem Schleifendurchlauf stattfinden, wird jede FOR-NEXT-Schleife mindestens einmal durchlaufen.

Beispiel:

```
10 FOR I=1 TO 10
20 PRINT I
30 NEXT I
```

Für die I-Werte von 1 bis 10 ...
... drucke den I-Wert ...
... nächster I-Wert.

Das obige Beispiels-Programm druckt die Zahlen 1,2,3,4,5,6,7,8,9 und 10 aus. Bei jedem Durchlauf durch die Schleife wird der Wert der Zahl I um Eins erhöht.

Soll die Schrittweite ungleich Eins sein, so muß sie mit Hilfe der STEP-Anweisung angegeben werden:

```
10 FOR I=1 TO 10 STEP 2
20 PRINT I,
30 NEXT I
```

Für die I-Werte von 1 bis 10 ...
... drucke den I-Wert ...
... nächster I-Wert.
Der nächste I-Wert ergibt sich aus dem alten I-Wert plus der Schrittweite (hier: 2). Der maximale I-Wert ist in diesem Beispiel in Zeile 10 auf den Wert 10 (TO 10) festgelegt worden.

Dieses Programm druckt die ungeraden Zahlen von 1 bis 9 aus.

4.4.5.8. Der FREE-Befehl

Mit dem FREE-Befehl kann man den freien Programmspeicherplatz abfragen.

Beispiele:

```
PRINT FREE
```

Drucke die Anzahl der freien Bytes im Programmspeicher.

```
F = FREE
```

Weise der Variablen F die Anzahl der freien Bytes im Programm-Speicher als Wert zu.

4.4.5.9. Der GOSUB-Befehl

Mit dem GOSUB-Befehl kann ein BASIC-Unterprogramm aufgerufen werden.

Beispiele:

unbedingter Aufruf:

```
GOSUB 123
```

Rufe das Unterprogramm auf, das bei der Zeile 123 beginnt.

bedingter Aufruf:

```
IF Z>7 THEN GOSUB 800
```

Wenn der Wert der Variablen Z größer als 7 ist, dann rufe das Unterprogramm auf, das bei der Zeile 800 beginnt.

Unterprogramme müssen mit dem Befehl "RETURN" abgeschlossen sein. Der Befehl "RETURN" bewirkt, daß die Programmausführung mit dem Befehl fortgesetzt wird, der dem "GOSUB"-Befehl folgt. Dieser Befehl kann in der gleichen Programm-Zeile wie der GOSUB-Befehl stehen.

4.4.5.10. Der GOTO-Befehl

Mit dem GOTO-Befehl kann der Rechner angewiesen werden, die Programmausführung bei einer anderen Programmzeile fortzusetzen.

Beispiele:

unbedingter Sprung:

```
GOTO 123
```

Setze die Programmausführung bei der Zeile 123 fort.

bedingter Sprung:

```
IF Z>7 THEN GOTO 800
```

Wenn der Wert der Variablen Z größer als 7 ist, dann setze die Programmausführung bei der Zeile 800 fort.

4.4.5.11. Der IF-Befehl

Mit dem IF-Befehl können Bedingungen abgefragt werden. Ist die Bedingung erfüllt, soll der Rechner eine bestimmte Anweisung oder Anweisungsfolge durchführen. Diese Anweisungsfolge muß durch das Schlüsselwort THEN eingeleitet werden. Ist die Bedingung nicht erfüllt, so wird der Rest der Programmzeile übersprungen. Die Programmausführung wird dann mit der nächsten Zeile fortgesetzt.

Grundaufbau:

```
IF ......... THEN .......
wenn          dann
```

Beispiele:

```
IF Z<5 THEN PRINT "Z<5"
```

Wenn der Wert der Variablen Z kleiner als fünf ist, dann drucke "Z<5".

```
IF X+Y=7 THEN GOTO 10
```

Wenn die Summe der Variablen X und Y den Wert 7 ergibt, dann setze die Programmausführung bei der Zeile 10 fort.

Gültige Vergleichsoperatoren sind:

<	kleiner
<=	kleiner gleich
=	gleich
>=	größer gleich
>	größer
<>	ungleich

4.4.5.12 Der INP-Befehl

Mit dem INP-Befehl können Werte von Eingabe-Baugruppen gelesen werden.

Beispiele:

```
X = INP(5)
```

Lese einen Wert von der Eingabebaugruppe mit der Adresse 5 (dezimal) und weise den gelesenen Wert der Variablen X zu.

```
PRINT INP( DEC(1A) )
```

Lese einen Wert von der Eingabebaugruppe mit der Adresse 1A (hexadezimal) und drucke den gelesenen Wert.

Der Adress-Wert darf zwischen 0 und 255 (dezimal) bzw. 0 und FF (hexadezimal) liegen.

4.4.5.13. Der INPUT-Befehl

Mit dem INPUT-Befehl können dezimale und hexadezimale Werte von der Tastatur eingelesen werden. Eine Eingabe muß durch CR abgeschlossen werden. Jeder Eingabe-Wert wird einer Variablen zugewiesen. Der Name der Variablen erscheint als Eingabeaufforderung auf dem Bildschirm.
Wenn vom Programm speziell die Eingabe eines hexadezimalen Wertes erwartet wird, so erscheint vor dem Variablen-Namen das Zeichen "#" auf dem Bildschirm. Hexadezimale Eingaben müssen in Klammern eingeschlossen sein.

Beispiel:

```
INPUT A,B,C
```

Lese drei Werte von der Tastatur. Weise diese Werte den Variablen A, B und C zu.

Auf dem Bildschirm erscheint:

```
A=
```

Das Programm fordert den Wert für die Variable A an. Nach der Eingabe des Wertes erfolgen entsprechende Eingabeanforderungen für die zwei anderen Variablen.

BASIC / INPUT-Befehl

Eine INPUT-Anweisung kann auch Texte enthalten, die erläutern, welche Eingabe vom Programm gefordert wird:

Beispiel:

```
INPUT "WERT 1 ",A,"WERT 2 ",B
```

Das Programm erwartet die Eingabe von zwei Werten. Die Eingabe-Werte werden den Variablen A und B zugewiesen.

Auf dem Bildschirm erscheint:

```
WERT 1 A=
```

Der erste Text wird ausgedruckt und die Eingabe für die Variable A wird angefordert. Nach der Eingabe des Wertes erfolgt ein entsprechender Ausdruck für die Variable B.

Durch die INPUT-Anweisung kann auch die Eingabe eines hexadezimalen Wertes angefordert werden.

Beispiel:

```
INPUT #A
```

Das Zeichen "#" vor dem Variablennamen gibt an, daß ein hexadezimaler Wert eingegeben werden muß.

Auf dem Bildschirm erscheint:

```
#A=
```

Das Zeichen "#", das in der INPUT-Anweisung enthalten ist, wird als Information für den Benutzer auf dem Bildschirm ausgegeben. Der hexadezimale Eingabe-Wert muß in Klammern eingeschlossen sein.

Das BFZ-Steuer-BASIC erlaubt nicht nur Konstante wie 3, 1234 und 455 als Eingabewerte. Es können auch Ausdrücke wie 4096/1024, 2*DEC(3C00) und 88+X eingegeben werden. Enthalten diese Ausdrücke Variable, so wird der augenblickliche Variablen-Wert zur Berechnung verwendet.

4.4.5.14. Die LET-Anweisung

Bei der Zuweisung von Variablenwerten erfordern manche BASIC-Versionen die Anweisung LET. Diese Anweisung kann im BFZ-Steuer-BASIC entfallen.

Beispiele:

`LET X=3*Z (oder X=3*Z)`	Weise der Variablen X den Wert des Ausdrucks 3*Z zu.
`LET A=B (oder A=B)`	Weise der Variablen A den Wert der Variablen B zu.

4.4.5.15. Der LIST-Befehl (Nur im Direkt-Modus)

Der LIST-Befehl bewirkt das Auflisten der Programmzeilen auf dem Bildschirm. Wenn der Bildschirm voll ist, wird die Meldung "==> SPACE" ausgegeben. Durch Druck auf die Space-Taste kann die Auflistung fortgesetzt werden. Bei eingeschaltetem Drucker erfolgt eine kontinuierliche Auflistung.

Beispiele:

`LIST`	Liste alle Programmzeilen.
`LIST 20`	Liste nur die Programmzeile mit der Zeilennummer 20.

4.4.5.16. Der LOAD-Befehl (Nur im Direkt-Modus)

Mit dem LOAD-Befehl kann ein BASIC-Programm von einer Kassette in den Programmspeicher geladen werden. Hierzu wird das Kassetten-Interface BFZ/MFA 4.4.a benötigt. Sollte beim Laden ein Fehler auftreten, so wird eine entsprechende Meldung ausgegeben. Der Programmspeicher ist dann leer.

Beispiel:

Anzeige:

```
READY
>LOAD

 SPACE, DANN BAND EINSCHALTEN


READY
>_
```

Eingabe: LOAD [CR]

Space-Taste drücken, dann Recorder einschalten.

Der Rechner ist bereit, weitere Befehle entgegenzunehmen.

4.4.5.17. Der LPOFF-Befehl

Mit dem Befehl LPOFF kann der Drucker ausgeschaltet werden. Das Einschalten des Druckers erfolgt mit dem LPON-Befehl. Wenn der Drucker ausgeschaltet ist, erfolgt die Ausgabe nur noch auf dem Bildschirm.
Dieser Befehl darf auch in einem Programm enthalten sein.

Beispiel:

```
LPOFF
```

Drucker aus.

4.4.5.18. Der LPON-Befehl

Mit dem Befehl LPON kann der Drucker eingeschaltet werden. Das Ausschalten des Druckers erfolgt mit dem LPOFF-Befehl. Wenn der Drucker eingeschaltet ist, erfolgt die Ausgabe auf dem Bildschirm und auf dem Drucker.
Dieser Befehl darf auch in einem Programm enthalten sein.

Beispiel:

```
LPON
```

Drucker ein.

4.4.5.19. Der NEW-Befehl (Nur im Direkt-Modus)

Mit dem Befehl NEW kann der gesamte Programm- und Variablenspeicher gelöscht werden. Nach der Befehlsausführung haben alle Variablen den Wert Null.

Beispiel:

```
NEW
```

Lösche Programm- und Variablenspeicher.

4.4.5.20. Der NEXT-Befehl

Der NEXT-Befehl schließt eine FOR-NEXT-Schleife ab. Näheres entnehmen Sie bitte dem Abschnitt 4.4.5.7. (FOR).

4.4.5.21. Der NOT-Befehl

Mit dem NOT-Befehl kann eine logische Negation durchgeführt werden. Bitte beachten Sie hierzu auch den Abschnitt 4.4.1.

Beispiele:

```
X = NOT ( DEC(AAAA) )
```

Negiere den hexadezimalen Wert AAAA und weise das Ergebnis (5555 hexadezimal) der Variablen X zu.

```
IF NOT (X=5) THEN PRINT "X IST UNGLEICH 5"
```

Wenn X nicht gleich 5, dann drucke "X IST UNGLEICH 5". Dieses Beispiel zeigt die Negation eines Vergleichs (X=5).

4.4.5.22. Der OUT-Befehl

Mit dem OUT-Befehl können Werte auf Ausgabe-Baugruppen ausgegeben werden.

Beispiele:

```
OUT   32,4
```

Gebe den dezimalen Wert 4 auf der Ausgabebaugruppe mit der Adresse 32 (dezimal) aus.

```
OUT DEC(1A),X
```

Gebe den Wert der Variablen X auf der Ausgabe-Baugruppe mit der Adresse 1A (hexadezimal) aus.

Der Adress-Wert darf zwischen 0 und 255 liegen.
Der Ausgabe-Wert darf zwischen 0 und 255 liegen.

4.4.5.23. Der OR-Befehl

Mit dem OR-Befehl kann eine logische ODER-Verknüpfung durchgeführt werden. Bitte beachten Sie hierzu auch Abschnitt 4.4.1.

Beispiele:

```
X = A OR B
```

Weise der Variablen X das Ergebnis der ODER-Verknüpfung der Variablen A und B zu.

```
Z = 4 OR Y
```

Weise der Variablen Z das Ergebnis der ODER-Verknüpfung zwischen der Zahl 4 und der Variablen Y zu.

```
IF (X=5) OR (X=10) THEN GOTO 10
```

Wenn der Wert von X gleich 5 ist oder wenn der Wert von X gleich 10 ist, dann setze die Programmausführung mit der Zeile 10 fort.

Bitte beachten Sie hier die KLAMMERN UM DIE VERGLEICHS-AUSDRÜCKE "X=5" und "X=10".

4.4.5.24. Der PEEK-Befehl

Mit dem PEEK-Befehl können die Inhalte von Speicherzeilen gelesen werden. Bitte beachten Sie hierzu auch den Abschnitt 4.4.1.

Beispiele:

```
PRINT PEEK( DEC(3C00) )
```

Lese den Inhalt der Speicherzeile mit der Adresse 3C00 (hexadezimal) und drucke den gelesenen Wert.

```
M = PEEK(123)
```

Lese den Inhalt der Speicherzeile mit der Adresse 123 (dezimal) und weise der Variablen M den gelesenen Wert zu.

4.4.5.25. Der POKE-Befehl

Mit dem POKE-Befehl ist es möglich, Werte in RAM-Speicherzeilen zu schreiben. Bitte lesen Sie hierzu auch den Abschnitt 4.4.1.

Beispiele:

```
POKE DEC(E000),DEC(FF)
```

Schreibe in die Speicherzeile mit der Adresse E000 (hexadezimal) den hexadezimalen Wert FF.

```
POKE 1234,56
```

Schreibe in die Speicherzeile mit der Adresse 1234 (dezimal) den dezimalen Wert 56.

Der Wert, der in eine Speicherzeile geschrieben werden soll, darf zwischen 0 und 255 liegen.

4.4.5.26. Der PRINT-Befehl

Mit dem PRINT-Befehl können Werte von Variablen, Rechenergebnisse und Texte auf dem Bildschirm ausgegeben werden.

Beispiele:

```
PRINT A
```

Drucke den Wert der Variablen A.

```
PRINT A+1
```

Drucke das Ergebnis des Ausdrucks A+1.

```
PRINT "TEST"
```

Drucke den Text "TEST". Texte müssen in Anführungsstriche eingeschlossen sein.

Endet die PRINT-Anweisung nicht mit einem Komma oder einem Semikolon, so wird bei der Abarbeitung der PRINT-Anweisung zum Abschluß ein Zeilenvorschub ausgegeben. Dies hat zur Folge, daß die nächste Ausgabe in der nächsten Zeile erfolgt.

Endet die PRINT-Anweisung mit einem Komma, so erfolgt die nächste Ausgabe bei der nächsten Tabulator-Marke. Die Tabulator-Marken stehen fest bei jeder 8-ten Spalte.

Beispiel:

```
PRINT "AAA", : PRINT "BBB"
```

Gebe AAA aus. Drucke BBB an der nächsten Tabulator-Position.

Ausgabe:

```
AAA     BBB
```

Soll die nächste Ausgabe direkt an die letzte Ausgabe anschließen, so muß der PRINT-Befehl mit einem Semikolon abgeschlossen werden.

Beispiel:

```
PRINT "AAA"; : PRINT "BBB"
```

Drucke AAA und direkt anschließend BBB.

Ausgabe:

```
AAABBB
```

Bei der Ausgabe von Zahlen-Werten gelten folgende Besonderheiten:

- Nach jeder Zahl wird ein Leerzeichen ausgegeben
- Vor positiven Zahlen-Werten steht ein Leerzeichen
- Vor negativen Zahlen-Werten steht ein Minuszeichen

Will man mehrere Zahlen oder Texte ausgeben (z.B. "AAA" und "BBB" im obigen Beispiel), so kann man dies mit EINER PRINT-Anweisung machen. Dazu müssen die einzelnen Ausdrücke, Variablen, Zahlen oder Texte durch ein Komma oder ein Semikolon getrennt werden. Das Ausgabeformat richtet sich nach dem Trennzeichen:

Befehl:

```
PRINT "AAA","BBB"
```

Ausgabe:

```
AAA      BBB
```

Befehl:

```
PRINT "AAA";"BBB"
```

Ausgabe:

```
AAABBB
```

Zahlen-Werte werden normalerweise in der dezimalen Schreibweise ausgegeben. Sollen die Werte hexadezimal ausgegeben werden, so muß der Ziffer, der Variablen oder dem Ausdruck das Zeichen "#" vorangestellt werden. Bitte beachten Sie hierzu auch den Abschnitt 4.4.1.

Beispiele:

```
PRINT #255,255
```

Ausgabe:

```
00FF     255
```

Hexadezimal-Werte werden immer 4-stellig ausgegeben. Das "#"-Zeichen bezieht sich nur auf eine Zahl, bzw. eine Variable oder einen Ausdruck.

Ein weiteres Beispiel:

```
10 A=10
20 B=DEC(0F)

30 PRINT #A,#B,#A+B
40 PRINT  A, B, A+B
```

Wertzuweisung (Dezimalzahl).
Wertzuweisung (Hexadezimal-Zahl)
Drucke Hexadezimal-Werte.
Drucke Dezimal-Werte.

Ausgabe:

```
000A     000F     0019
 10       15       25
```

Hexadezimal-Werte
Dezimal-Werte

4.4.5.27. Das QUIT-Kommando (Nur im Direkt-Modus)

Mit dem QUIT-Kommando kann zur Monitorerweiterung MAT 85+ zurückgekehrt werden. Sie ist dann bereit, weitere Kommandos entgegenzunehmen.

Beispiel:

```
READY
>QUIT


KMD+>
```

QUIT [CR] eingeben

MAT 85+ meldet sich und ist bereit, weitere Kommandos entgegenzunehmen.

4.4.5.28. Das READ-Kommando

Mit dem READ-Kommando können Daten gelesen werden, die mit dem DATA-Befehl im Programm abgelegt wurden. Weitere Informationen entnehmen Sie bitte dem Abschnitt 4.4.5.4.

Beispiele:

1.

```
10 READ A
20 READ B
30 PRINT A;B

40 DATA 11,22
```

Lese den 1. Wert (11).
Lese den 2. Wert (22).
Drucke die Werte von A und B.
Werte, die gelesen werden.

Ausgabe:

```
 11  22
```

Ausgabe der Werte von A und B. Die Werte wurden mit den READ-Anweisungen gelesen.

2.

Mit einer READ-Anweisung können auch mehrere Daten-Werte gleichzeitig gelesen werden:

```
10 READ A,B
20 PRINT A;B

30 DATA 11,22
```

Lese 1. und 2. Wert.
Drucke die Werte von A und B.
Werte, die gelesen werden.

Ausgabe:

```
 11  22
```

Ausgabe der Werte von A und B. Die Werte wurden mit der READ-Anweisung gelesen.

Hinweis: Die beiden Beispiele sind gleichwertig.

4.4.5.29. Das REM-Kommando (REMARK)

Mit dem REM-Kommando können Kommentare in das Programm eingefügt werden. Wenn der Rechner das REM-Kommando erkennt, ignoriert er den Rest der Programmzeile. Die Abarbeitung des Programms wird bei der nächsten Zeile fortgesetzt. Aus diesem Grund darf dem REM-Kommando in der gleichen Zeile kein Befehl folgen.

Beispiel:

```
REM DIES IST EIN KOMMENTAR
```

```
X=3*X:REM MULTIPLIZIERE X MIT 3
```

Ein Kommentar kann auch einer anderen Anweisung folgen.

4.4.5.30. Der RESTORE-Befehl

Wenn mit dem READ-Befehl Daten gelesen werden sollen, beginnt der Rechner normalerweise mit dem Datum nach dem ersten DATA-Befehl. Es werden der Reihe nach alle Daten bis zum letzten Datum gelesen. Soll die Reihenfolge geändert oder sollen einige Daten erneut gelesen werden, kann dies mit dem RESTORE-Kommando erreicht werden. Dem RESTORE-Kommando folgt eine Zeilennummer, die angibt, ab welcher Zeile die nächsten Daten gelesen werden sollen. Bitte lesen Sie auch Abschnitt 4.4.5.4. (DATA).

Beispiel:

```
RESTORE  10
```

Lese beim nächsten READ ab Zeile 10.

Programm-Beispiel:

```
10 READ X,Y,Z
20 PRINT X;Y;Z
30 RESTORE 70

40 READ X
50 PRINT X
60 DATA 10
70 DATA 20
80 DATA 30
```

Lese drei Werte.
Drucke die gelesenen Werte.
Lese beim nächsten READ ab Zeile 70.
Lese einen Wert.
Drucke den Wert.
1. Wert.
2. Wert.
3. Wert.

Ausgabe:

```
 10   20   30
 20
```

Die ersten drei Werte.
Der vierte Wert (nach RESTORE).

4.4.5.31. Das RETURN-Kommando

Der letzte Befehl eines Unterprogramms muß RETURN lauten. Die Programmabarbeitung wird bei dem Befehl fortgesetzt, der dem GOSUB-Befehl folgt. Weitere Informationen entnehmen Sie bitte dem Abschnitt 4.4.5.9. (GOSUB).

4.4.5.32. Das RND-Kommando

Mit diesem Kommando lassen sich Zufallszahlen erzeugen. Die erzeugte Zufallszahl liegt zwischen 1 und dem Wert, der in Klammern angegeben wird.

Beispiele:

```
PRINT RND(49)
```

Drucke eine Zufallszahl zwischen 1 und 49.

```
X=RND(Z*3)/5
```

Erzeuge eine Zufallszahl zwischen 1 und dem Ergebnis des Ausdrucks Z*3. Teile diese Zufallszahl durch 5 und weise das Ergebnis der Variablen X zu.

4.4.5.33. Das RUN-Kommando (Nur im Direkt-Modus)

Mit dem RUN-Kommando kann ein BASIC-Programm gestartet werden. Die Programmabarbeitung beginnt mit der ersten Programmzeile.Beim Programmstart werden alle Variablenwerte auf Null gesetzt.

Beispiel:

```
READY
>RUN
```

RUN [CR] eintippen.

Das BASIC-Programm wird gestartet.

4.4.5.34. Das SAVE-Kommando (Nur im Direkt-Modus)

Mit dem SAVE-Kommando ist es möglich, ein im Programmspeicher befindliches Basic-Programm auf eine Kassette abzuspeichern. Dazu ist das Kassetten-Interface BFZ/MFA 4.4.a erforderlich.

Beispiel:

```
READY
>SAVE

 BAND EINSCHALTEN, DANN SPACE




READY
>_
```

SAVE [CR] eintippen.

Erst muß der Recorder eingeschaltet werden, dann ist die Space-Taste zu drücken. Das Programm wird dann aufgezeichnet.

Der Rechner ist anschließend bereit, weitere Kommandos entgegenzunehmen.

4.4.5.35. Das STEP-Kommando

Mit dem STEP-Kommando kann die Schrittweite bei FOR-NEXT-Schleifen festgelegt werden. Weitere Informationen entnehmen Sie bitte dem Abschnitt 4.4.5.7. (FOR).

4.4.5.36. Das STOFF-Kommando

Mit dem STOFF-Kommando kann der Einzelschritt-Modus abgeschaltet werden. Dieser Befehl ist auch innerhalb eines Programms zulässig. Dadurch ist es im Zusammenhang mit STON möglich, einzelne Programmteile im Einzelschritt-Betrieb bearbeiten zu lassen. Weitere Informationen entnehmen Sie bitte dem Abschnitt 4.4.5.37. (STON).

Beispiel:

```
STOFF
```

Einzelschritt-Modus aus.

4.5.37. Das STON-Kommando

Mit dem STON-Kommando kann der Einzelschritt-Modus eingeschaltet werden. Dieser Befehl ist auch innerhalb eines Programms zulässig. Dadurch ist es möglich, einzelne Programmteile im Einzelschritt-Betrieb bearbeiten zu lassen. In diesem Modus wird jeder Programm-Befehl vor seiner Ausführung angezeigt.
Der Befehl wird erst ausgeführt, wenn die Space-Taste betätigt wird. Drückt man statt dessen CR, so wird das Programm abgebrochen.
Im STEP-Modus wird außerdem angezeigt, daß eine neue Programmzeile abgearbeitet wird oder daß sich ein Variablenwert ändert. Diese Angaben sind zur besseren Unterscheidung von PRINT-Ausgaben in spitze Klammern eingeschlossen. Wenn der Bildschirm voll ist, erscheint die Meldung "==> SPACE". Der Programm-Ablauf wird fortgesetzt, wenn die Space-Taste betätigt wird. Betätigt man CR, so wird das Programm abgebrochen. Bei eingeschaltetem Drucker erfolgt ein kontinuierlicher Ausdruck.

Beispiel:

```
STON
```

Einzelschritt-Modus ein.

Ausgabe bei eingeschaltetem Einzelschritt-Modus:

Programm:

```
10 PRINT "START"
20 I=I+1
30 PRINT I
40 PRINT "ENDE"
```

Dieses Programm kann auch bei eingeschaltetem Einzelschritt-Modus eingegeben werden.

Ausgabe nach RUN CR :

```
<10> <PRINT "START">START
```

Zeile 10.
Anweisung: PRINT "START".

```
<20> <I=I+1><I= 0 , 1>
```

Zeile 20.
Anweisung: I=I+1.
Alter I-WERT: 0,
neuer I-Wert: 1.

```
<30><PRINT I> 1
```

Zeile 30.
Anweisung: PRINT I.

```
<40> <PRINT "ENDE">ENDE
```

Zeile 40.
Anweisung: PRINT "ENDE".

Alle Ausgaben, die nicht in spitze Klammern eingeschlossen sind, sind normale PRINT-Ausgaben.

4.4.5.38. Die STOP-Anweisung

Die STOP-Anweisung bricht die Programmausführung ab. Der Rechner druckt dann die Nummer der Zeile aus, in der der Abbruch erfolgte. Enthält ein Programm mehrere STOP-Befehle, so wird die Programm-Bearbeitung gestoppt, sobald einer der STOP-Befehle erreicht wird. Dies ist beim Testen von Verzweigungen nützlich. Durch die Eingabe von PRINT-Befehlen in Verbindung mit der Angabe von Variablen können dann Variablen-Werte abgefragt werden. Dieser Befehl ist bei der Programm-Entwicklung nützlich.

Geben Sie folgendes Programm ein und starten Sie es mit RUN:

```
10 FOR I=1 TO 10
20 A=I*2
30 B=A-1


35 STOP


40 PRINT A,B
50 NEXT I
```

Dieser Befehl wurde zu Testzwecken eingefügt.

Ausgabe nach dem Start mit RUN:

```
STOP IN ZEILE 35
```

Die Variablen-Werte können nun abgefragt werden:

```
PRINT I,A,B
```

CR

Anzeige der Werte:

```
1        2        1
```

4.4.5.39. Der THEN-Befehl

Der THEN-Befehl wird im Zusammenhang mit dem IF-Befehl verwendet. Nähere Informationen entnehmen Sie bitte dem Abschnitt 4.4.5.11. (IF).

4.4.5.40. Der TO-Befehl

Mit dem TO-Befehl wird der Endwert der Schleifenvariablen einer FOR-NEXT-Schleife festgelegt. Nähere Informationen entnehmen Sie bitte dem Abschnitt 4.4.5.7. (FOR).

4.4.5.41. Der TROFF-Befehl

Mit dem TROFF-Befehl kann der TRACE-Modus ausgeschaltet werden. Das Einschalten erfolgt mit dem TRON-Befehl. Dieser Befehl kann auch innerhalb eines Programms vorkommen. Dadurch ist es möglich, einzelne Programmteile im TRACE-Modus auszuführen. Nähere Informationen entnehmen Sie bitte dem Abschnitt 4.4.5.42. (TRON).

4.4.5.42. Der TRON-Befehl

Mit dem TRON-Kommando kann der TRACE-Modus eingeschaltet werden (trace = Spur). Dieser Befehl ist auch innerhalb eines Programms zulässig. Dadurch ist es möglich, einzelne Programmteile im TRACE-Betrieb durchzuführen. In diesem Modus wird jeder Programm-Befehl und das Ergebnis seiner Bearbeitung angezeigt. Die Änderung von Variablen-Werten wird dadurch sichtbar, daß ihre alten und neuen Werte ausgegeben werden. Diese Angaben sind, zur besseren Unterscheidung von PRINT-Ausgaben, in spitze Klammern eingeschlossen. Wenn der Bildschirm voll ist, erscheint die Meldung "==> SPACE". Der Programm-Ablauf wird fortgesetzt, wenn die Space-Taste betätigt wird. Betätigt man CR wird das Programm abgebrochen. Bei eingeschaltetem Drucker erfolgt ein kontinuierlicher Ausdruck.

Beispiel:

```
TRON
```

TRACE-Modus ein

Ausgabe bei eingeschaltetem TRACE-Modus:

Programm:

```
10 PRINT "START"
20 I=I+1
30 PRINT I
40 PRINT "ENDE"
```

Dieses Programm kann auch bei eingeschaltetem TRACE-Modus eingegeben werden.

Ausgabe:

```
<10><PRINT "START">START

<20><I=I+1><I= 0 , 1>


<30><PRINT I> 1

<40><PRINT "ENDE">ENDE
```

Zeile 10.
Anweisung: PRINT "START".

Zeile 20.
Anweisung: I=I+1.
Alter I-WERT: 0, neuer Wert: 1

Zeile 30.
Anweisung: PRINT I.

Zeile 40.
Anweisung: PRINT "ENDE".

4.4.5.43. Der USR-Befehl

Mit dem Befehl USR ist es möglich, Unterprogramme aufzurufen, die in 8085 Maschinensprache im Speicher des BFZ/MFA-Mikrocomputers abgelegt sind.
Die Anfangsadresse des Unterprogramms wird innerhalb der Klammern des USR-Befehls angegeben. Außerdem kann ein zweiter Wert folgen, der vom ersten durch ein Komma abgetrennt wird. Dieser zweite Wert steht beim Start des Unterprogramms im HL-Registerpaar der CPU 8085 zur Verfügung und kann vom Unterprogramm bearbeitet werden. Wird beim USR-Befehl kein zweiter Wert angegeben, so enthält das HL-Registerpaar beim Start des Unterprogramms dessen Anfangsadresse.
Das Unterprogramm wird mit einem Return-Befehl (RET, RZ, ...) beendet. Der Wert, der dann im HL-Registerpaar der CPU steht, kann vom BASIC-Programm weiter verwendet werden.
Da der USR-Befehl immer einen Wert als Ergebnis liefert, darf er nur

in Wertzuweisungen rechts vom Gleichheitszeichen

und

als Argument von PRINT-Befehlen

verwendet werden (siehe auch Beispiele).

Beispiele:

```
X=USR(1234)
```

Rufe das Unterprogramm auf, das ab der Adresse 1234 (dezimal) im Speicher steht.
Der Wert, der bei der Rückkehr vom Unterprogramm im HL-Registerpaar steht, wird der Variablen X zugewiesen.

```
X=USR(5678,44)*2
```

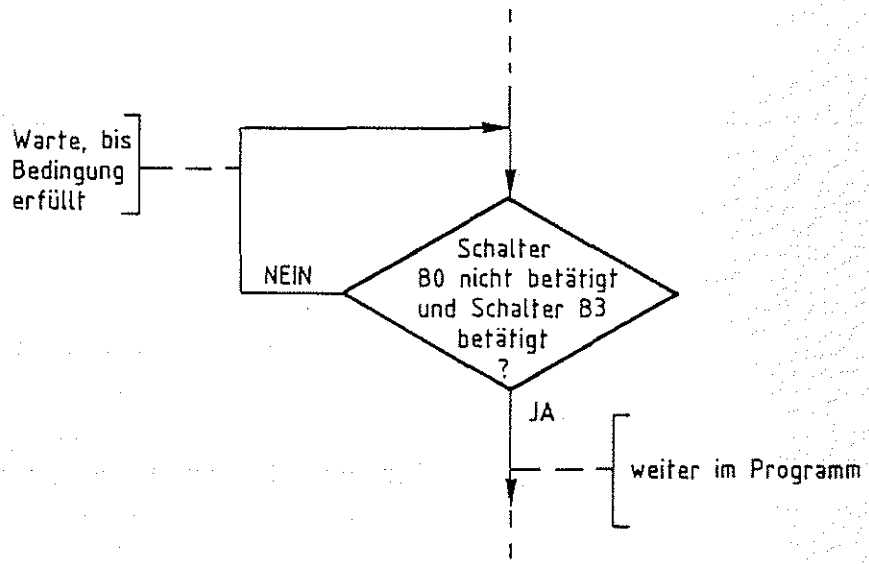
Rufe das Unterprogramm auf, das ab der Adresse 5678 (dezimal) im Speicher steht.
Der Wert 44 (dezimal) steht beim Start des Unterprogramms im HL-Registerpaar der CPU.
Der Wert, der bei der Rückkehr vom Unterprogramm im HL-Registerpaar steht, wird mit 2 multipliziert und der Variablen X zugewiesen.

PRINT USR(DEC(E000))	Rufe das Unterprogramm auf, das ab der Adresse E000 (hexadezimal) im Speicher steht. Der Wert, der bei der Rückkehr vom Unterpro- im HL-Registerpaar steht, wird ausgedruckt.

4.4.5.44. Der WAIT-Befehl

Der WAIT-Befehl soll an einem Beispiel erläutert werden:

In Steuerungsanlagen kann es vorkommen, das die weitere Abarbeitung des Steuerprogramms solange gestoppt werden soll, bis ein bestimmtes Eingangssignal an einer der Eingabebaugruppen anliegt. Das unten dargestellte Flußdiagramm zeigt einen solchen Fall.



Im dargestellten Diagramm soll das Programm solange in einer Warteschleife verharren, bis an der 8-Bit-Parallel-Eingabebaugruppe (BFZ/MFA 4.2.) der Schalter B0 nicht betätigt und der Schalter B3 betätigt wird. Die Stellungen der anderen Schalter sollen ohne Einfluß auf den Programmablauf bleiben. Die Adresse der Eingabebaugruppe sei 03.

BASIC / WAIT-Befehl

Mit dem INP-Befehl des Steuer-Basics kann der Zustand aller Schalter der Eingabebaugruppe abgefragt werden:

```
10 I=INP(3)
```

Der eingelesene Wert muß nun mit dem Soll-Wert verglichen werden. Ist der eingelesene Wert nicht gleich dem Soll-Wert, so wird der INP-Befehl erneut durchgeführt. Andernfalls wird die Programmabarbeitung mit den folgenden Befehlen fortgesetzt.
Da die Schalter B1, B2, B4, B5, B6 und B7 ohne Einfluß auf den Programmablauf sein sollen, müssen die entsprechenden Bits vor dem Vergleich mit dem Soll-Wert auf einen festen Wert ("0" oder "1") gesetzt werden. Mit einer logischen UND-Verknüpfung können einzelne Bits auf "0" gesetzt werden:

```
20 A=I AND 9
```

In der oben dargestellten Programm-Zeile wird der eingelesene Wert (I) mit dem Dezimalwert 9 Bit für Bit UND-verknüpft. Da in der binären Schreibweise der Dezimalzahl 9 nur die Bits B0 und B3 auf "1" gesetzt sind, werden im Verknüpfungsergebnis (A) alle anderen Bits auf "0" gesetzt.

0	0	0	0	0	0	0	0	X	X	X	X	X	X	X	X

Wert von I

∧

0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1

Wert von 9

=

0	0	0	0	0	0	0	0	0	0	0	0	X	0	0	X

Verknüpfungsergebnis A

X ≙ unbestimmt. Von der Schalterstellung abhängig.

Die Bits B0 und B3 des Verknüpfungsergebnisses (Variable A) entsprechen den Bits im eingelesenen Wert (I). Alle anderen Bits sind "0".
Wenn der Schalter B0 nicht betätigt und der Schalter B3 betätigt werden, ist Bit B0 gleich "0" und Bit B3 gleich "1". Die Variable A hat dann den Wert 8.
Dieser Wert ist der Soll-Wert. Wenn A ungleich 8 ist, muß der INP-Befehl in Zeile 10 erneut ausgeführt werden. Der Vergleich mit dem Soll-Wert und der eventuelle Rücksprung nach Zeile 10 kann mit einer IF-Anweisung erreicht werden:

```
30 IF A<>8 THEN GOTO 10
```

BASIC / WAIT-Befehl

Die drei Programmzeilen

```
10 I=INP(3)
20 A=I AND 9
30 IF A <> 8 THEN GOTO 10
```

können durch einen einzigen Befehl ersetzt werden:

```
10 WAIT 3,9,8
```

Dieser Befehl wird in drei Schritten abgearbeitet:

1. Lese einen Wert von der Eingabe-Baugruppe mit der Adresse 3.
2. Verknüpfe den eingelesenen Wert logisch UND mit 9.
3. Vergleiche das Ergebnis mit dem Wert 8. Wenn beide Werte unterschiedlich sind, dann beginne erneut bei Schritt 1. Sonst führe den nächsten Befehl aus.

Beispiele:

```
WAIT DEC(1A),DEC(5F),DEC(1D)
```

Lese einen Wert von der Eingabebaugruppe mit der Adresse 1A (hexadezimal). Führe eine UND-Verknüpfung mit dem hexadezimalen Wert 5F durch. Vergleiche das Ergebnis mit dem hexadezimalen Wert 1D. Wenn das Ergebnis nicht dem Wert 1D entspricht, dann beginne den Zyklus erneut. Sonst führe den nächsten Befehl aus.

BASIC / WAIT-Befehl

```
WAIT   A,B,C
```

Lese einen Wert von der Eingabebaugruppe, deren Adresse dem Wert der Variablen A entspricht. Führe eine UND-Verknüpfung des eingelesenen Wertes mit dem Wert der Variablen B durch. Vergleiche das Ergebnis mit dem Wert der Variablen C. Wenn das Ergebnis nicht dem Wert von C entspricht, dann beginne den Zyklus neu. Sonst führe den nächsten Befehl aus.

Dia Adresse der Eingabe-Baugruppe darf zwischen 0 und 255 liegen.

Achten Sie auch besonders darauf, daß das Ergebnis der UND-Verknüpfung der Vergleichsgröße (Variable C im zweiten Beispiel) entsprechen kann! Sonst wird die Warteschleife endlos durchlaufen.

4.4.6. Die vier Grundrechenarten

Die vier Grundrechenarten werden im BASIC durch die Symbole +,-,* und / dargestellt.

Addition:	+
Subtraktion:	-
Multiplikation:	*
Division:	/

Beispiele:

`X=12+3`	Weise der Variablen X das Ergebnis der Addition 12+3 zu.
`X=12-3`	Weise der Variablen X das Ergebnis der Subtraktion 12-3 zu.
`X=12*3`	Weise der Variablen X das Ergebnis der Multiplikation 12*3 zu.
`X=12/3`	Weise der Variablen X das Ergebnis der Division 12/3 zu.

Als Grundregel für gemischte Ausdrücke gilt: Punkt- vor Strichrechnung. Soll die Reihenfolge geändert werden, so sind Klammern zu setzen:

$$\frac{A+B}{C+D}$$ muß als (A+B)/(C+D) programmiert werden

4.4.7. Das $-Symbol

Mit dem $-Symbol kann die Tastatur direkt abgefragt werden, ohne das CR betätigt werden muß:

`PRINT $`	Lese Tastatur-Eingabe, drucke den Dezimalwert des entsprechenden ASCII-Codes.
`Y=$`	Lese Tastatur-Eingabe, weise der Variablen Y den entsprechenden Dezimalwert des ASCII-Codes zu.
`Y=$-DEC(30)`	Lese Tastatur-Eingabe, subtrahiere von dem entsprechenden Dezimalwert des ASCII-Codes den hexadezimalen Wert 30 und weise das Ergebnis der Variablen Y zu.

Beachten Sie die UNTERSCHIEDLICHEN Codes für Groß- und Kleinschreibung!

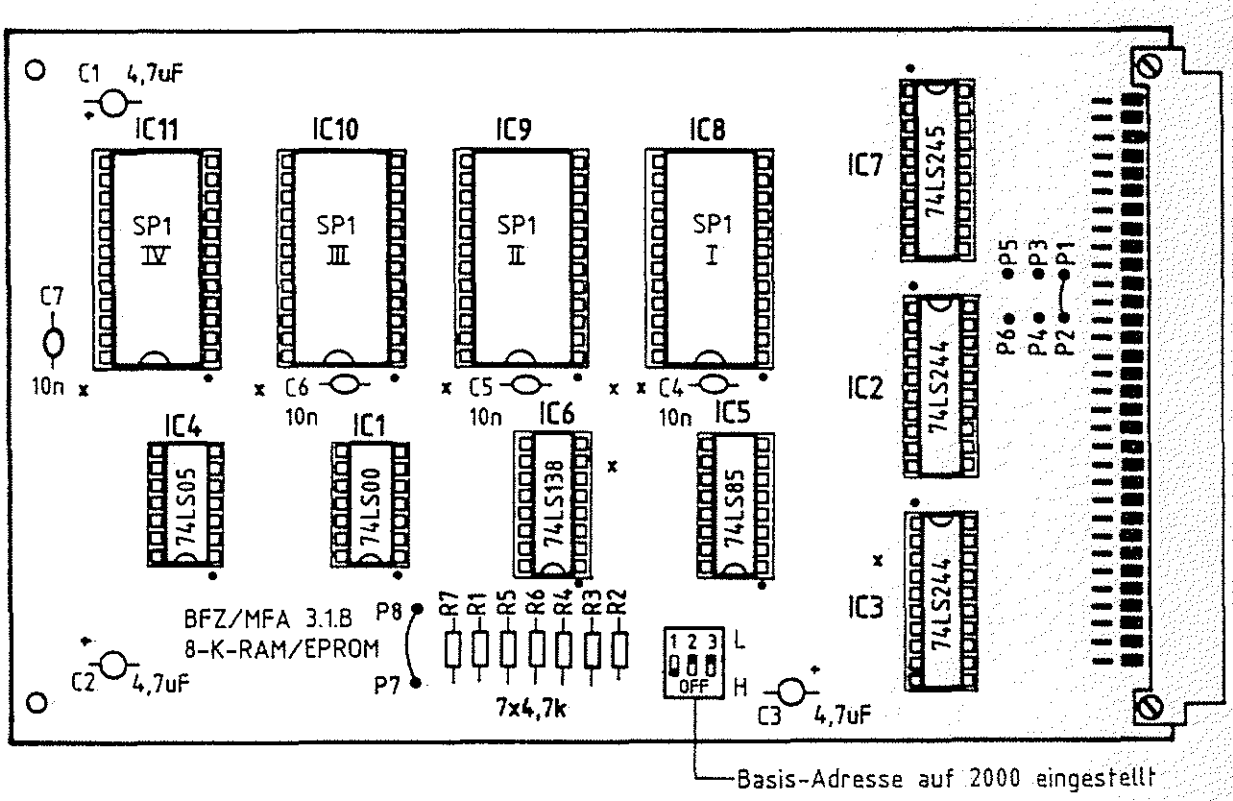
4.4.8. Das #-Symbol

Das #-Symbol wird in Zusammenhang mit dem INPUT- bzw. PRINT-Befehl verwendet. Bitte entnehmen Sie weitere Informationen den Abschnitten 4.4.5.13. (INPUT) bzw. 4.4.5.26. (PRINT).

5. Anhang

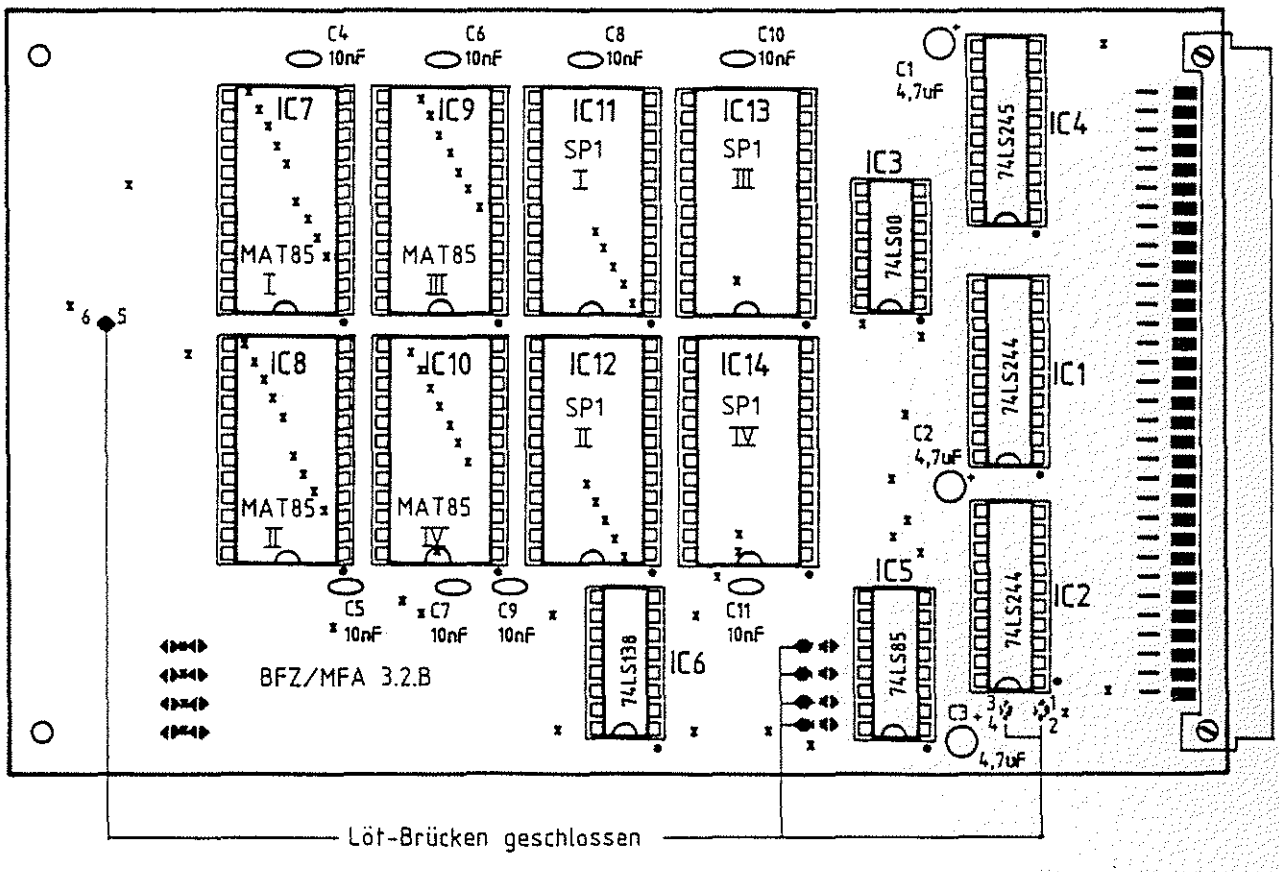
5.1. ROM-Bestückung

Das Software-Paket SP 1 ist in vier EPROMs vom Typ 2716 gespeichert. Es belegt den Speicherplatz von 2000 bis 3FFF. Die vier EPROMs können entweder zusammen mit dem Betriebsprogramm MAT 85 auf eine 16-K-RAM/EPROM-Baugruppe BFZ/MFA 3.2. oder alleine auf eine 8-K-RAM/EPROM-Baugruppe BFZ/MFA 3.1. gesteckt werden. Die beiden folgenden Abbildungen zeigen die richtige Anordnung der Speicherbausteine.
Beachten Sie, daß das Software-Paket SP 1 nur im Zusammenhang mit MAT 85 lauffähig ist !



8-K-RAM/EPROM-Baugruppe bestückt mit SP 1

Anhang



(Lötbrücken 1-2 und 3-4 auf der Leiterbahnseite)

16-K-RAM/EPROM-Baugruppe bestückt mit MAT 85 und SP 1

5.2. Schaltungserweiterungen

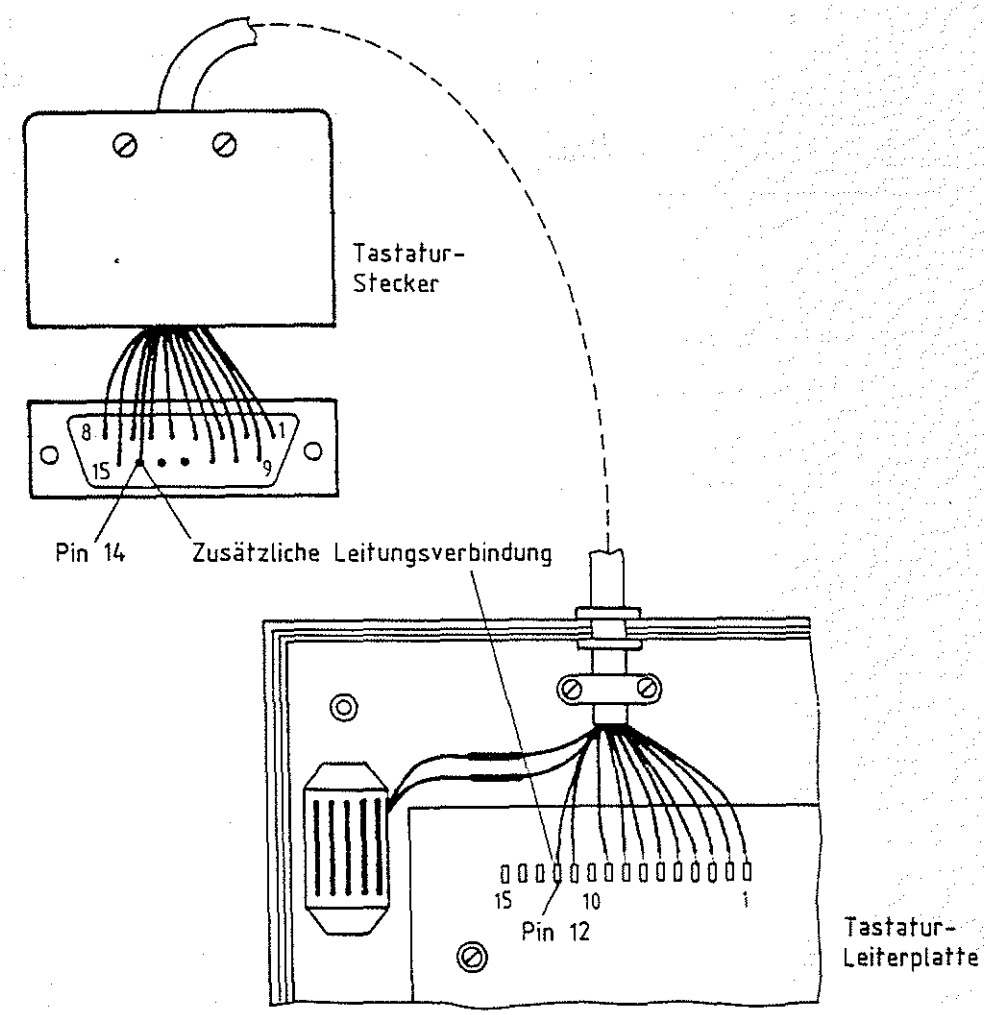
Um den SPS- und den BASIC-Interpreter voll nutzen zu können, sind zwei Schaltungs-Ergänzungen notwendig:

5.2.1. Programm-Abbruch durch Tastaturbetätigung

Ein laufendes SPS- bzw. BASIC-Programm kann durch Betätigung einer beliebigen Taste der Tastatur (außer CONTROL, BREAK und SHIFT) abgebrochen werden. Hierzu sind folgende Verdrahtungen durchzuführen:

1. Der Anschlußpin 14 des Tastatursteckers muß über eine freie Ader des Tastaturkabels mit Pin 12 der Tastaturplatine verbunden werden (siehe Abbildung). An diesem Platinenanschluß steht das Signal AKD (Any Key Down) zur Verfügung. Solange eine Taste (außer BREAK, CONTROL oder SHIFT) betätigt wird, ist AKD auf High-Pegel.

Verdrahtungsplan Tastatur- Stecker und -Leiterplatte



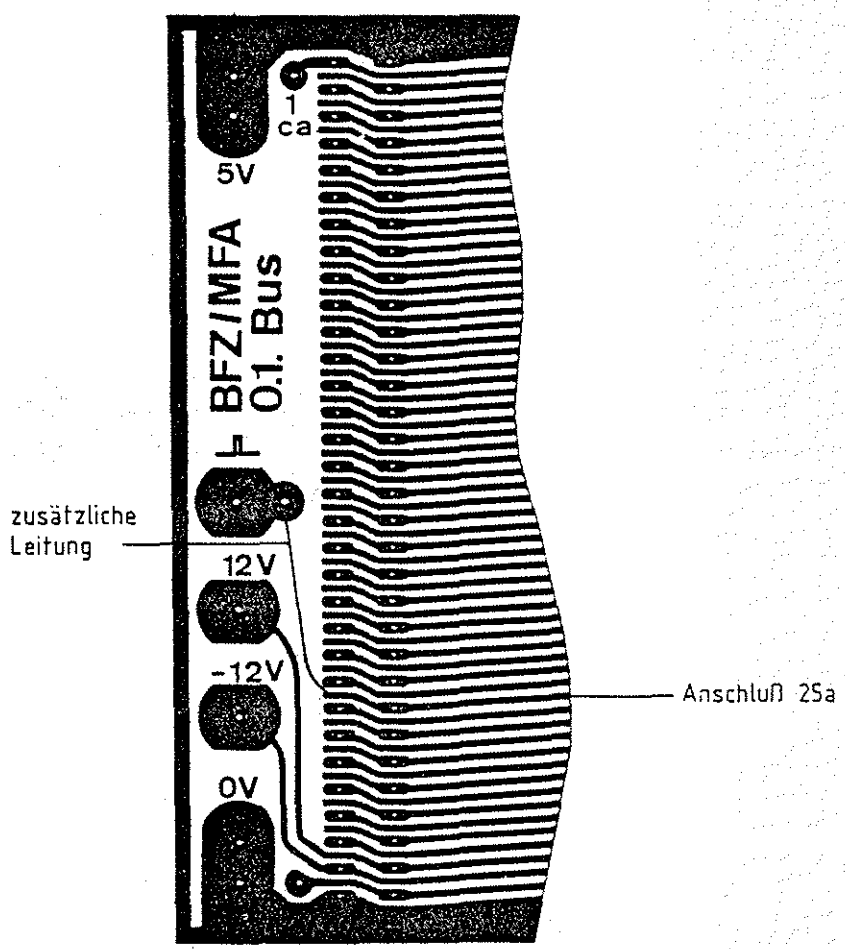
2. Der Anschlußpin 14 der Tastaturbuchse auf der Videokarte muß mit dem Anschluß-Stift 27c der 64-poligen Steckerleiste verbunden werden.

Durch diese Schaltungserweiterung erhält die CPU bei jedem Tastendruck eine Interrupt-Anforderung (RST 6.5). Dadurch kann ein laufendes Programm durch Druck auf eine Taste abgebrochen werden. Ohne diese Schaltungsergänzung ist ein Abbruch nur durch die Betätigung des RESET-Tasters möglich.

5.2.2. Zählimpuls für SPS-Software-Timer

Diese Schaltungserweiterung ist nur erforderlich, wenn die Software-Timer im SPS-Programm eingesetzt werden sollen. Dazu muß das Rechteck-Signal, das am Anschluß 23 der Spannungsregelung (BFZ/MFA 1.2.) zur Verfügung steht, auf den RST-7.5-Eingang der CPU geschaltet werden. Verbinden Sie dazu den Anschluß der Busplatine, der durch ein Rechtecksignal gekennzeichnet ist, mit der Leitung 25a der Busplatine. Will man auch andere Interrupt-Signale (z.B. bei Verwendung des Zähler- und Zeitgebers BFZ/MFA 4.6.) auf den RST-7.5-Eingang legen, so empfiehlt es sich, diese Verbindung steckbar auszuführen.

Verdrahtung der Bus-Platine



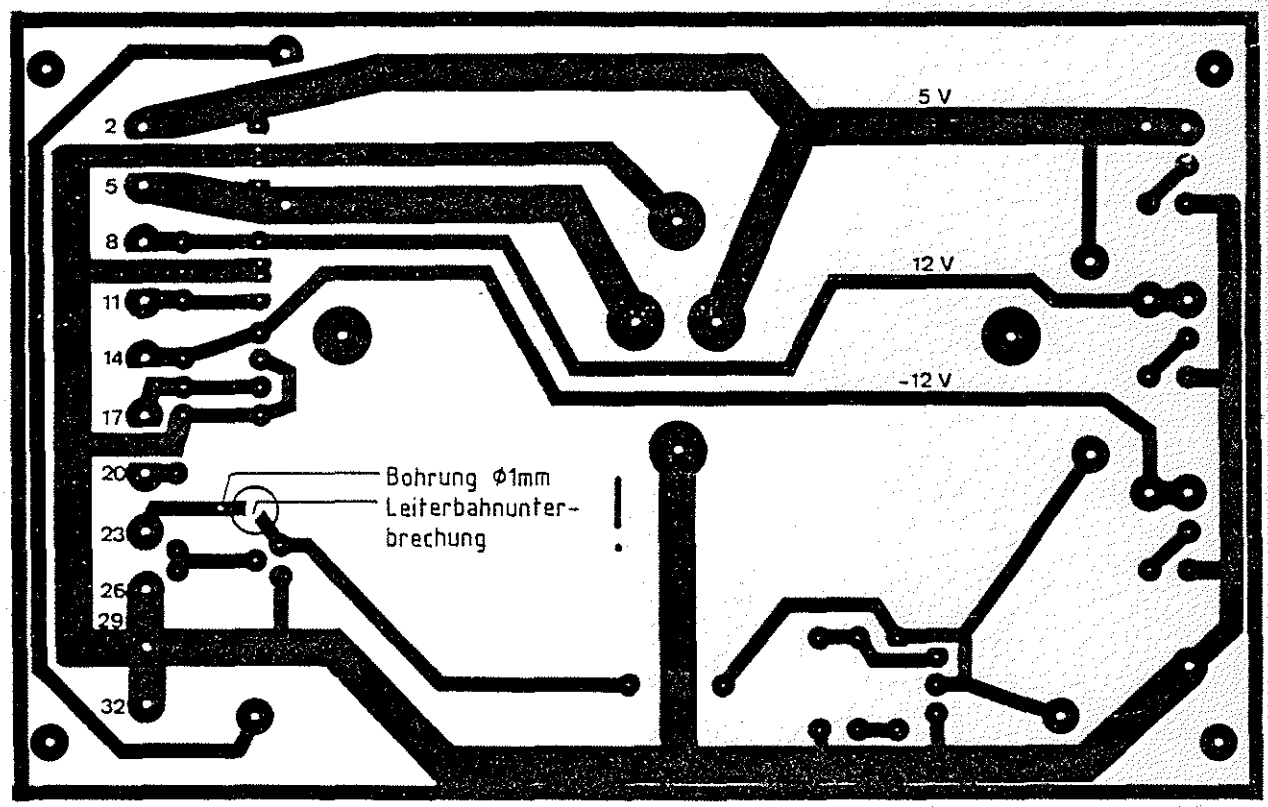
Anhang

Aufgrund von Bauteilstreuungen kann es vorkommen, daß der Rechteckimpuls verformt ist. In diesem Fall sollte die Schaltung der Spannungsregelung folgendermaßen ergänzt werden:

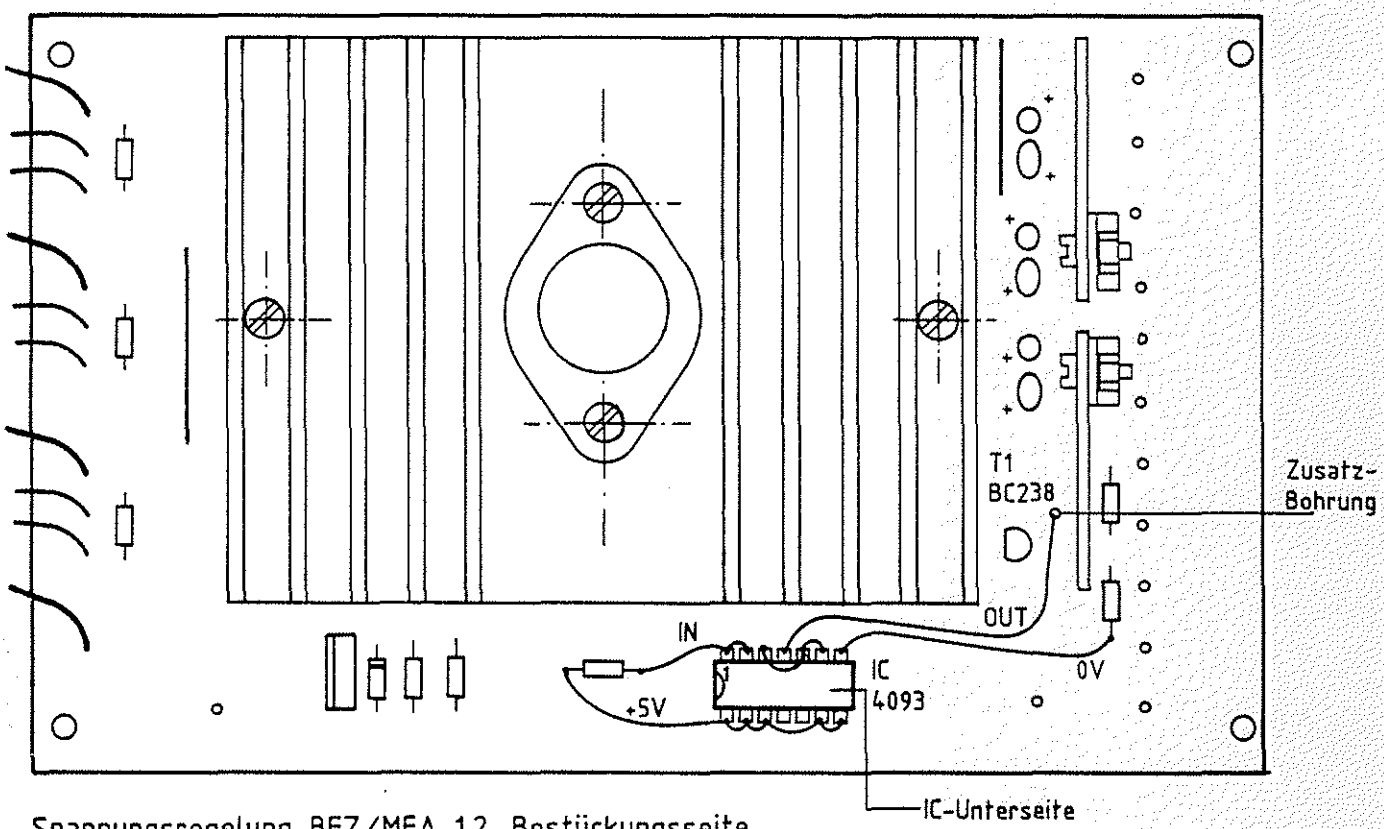
1. Trennen Sie die im Layout gekennzeichnete Leiterbahn auf und bohren Sie an der gekennzeichneten Stelle ein Loch von 1mm-Durchmesser.

2. Kleben Sie ein CMOS-IC 4093 (4-fach NAND mit Schmitt-Trigger) mit dessen Oberseite auf die Leiterplatte wie im Bestückungsplan auf der folgenden Seite dargestellt.

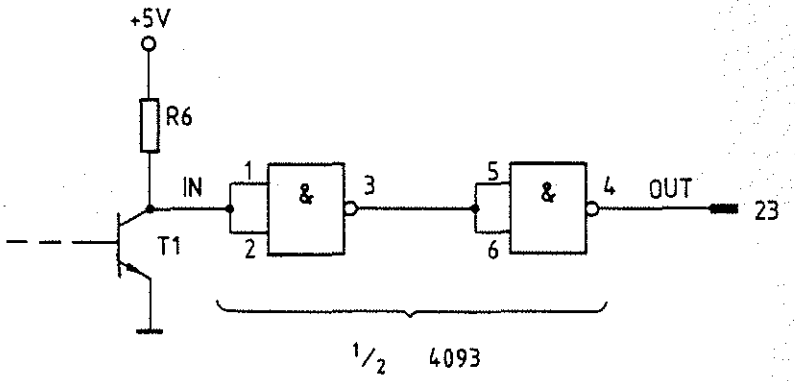
3. Verdrahten Sie das IC entsprechend dem Schaltbild.



Spannungsregelung BFZ/MFA 1.2. Leiterbahnseite



Spannungsregelung BFZ/MFA 1.2. Bestückungsseite



+5V: Anschlüsse 8, 9, 12, 13, 14
0V: Anschluß 7

Schaltbild

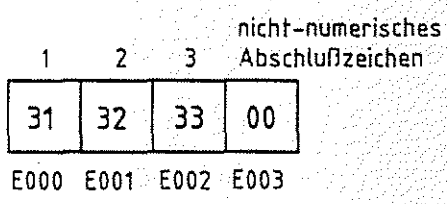
5.3. Unterprogramme im Software-Paket SP 1

Unterpr. Name	Eingangs-Adresse	veränd. Register	Funktion des Unterprogramms
COPY	219E	A,B,C,D, E,H,L	Kopiert einen Speicherinhalt. Anfangsadr. d. Quellber. in HL Endadresse d. Quellber. in DE Anfangsadr. d. Zielber. in BC. Fehlermeldung, wenn Zielbereich nicht vollständig mit RAM bestückt und Rücksprung zur Kommando-Eingabe von MAT 85+
COPY0	219F	A,B,C,D, E,H,L	Wie COPY, nur Bedeutung der Registerpaare DE und HL vertauscht.
COPY1	21A8	A,B,C,D, E,H,L	Kopiert einen Speicherinhalt. Anfangsadr. d. Quellber. in DE Länge d. Quellber. in BC Anfangsadr. d. Zielber. in HL. Fehlermeldungen: wie COPY
FLOOP	2371	A,B,D,H,L	Liest EINEN Wert von der Tastatur im Format ASCII, binär, dezimal, hexadezimal entsprechend dem Format-Code im C-Register (0,1,2,3) in die Speicherzelle ein, deren Adresse im HL-Registerpaar steht. Die Eingabe kann mit SP oder CR beendet werden. Nach dem RETURN steht das Abschlußzeichen (SP oder CR) im Akku. Der Inhalt des HL-Registerpaares ist dann um Eins erhöht und das eingelesene Zeichen steht im D-Register.

Anhang

Unterpr. Name	Eingangs-Adresse	veränd. Register	Funktion des Unterprogramms
INCHAR	2815	A	Liest ein Zeichen von der Tastatur. Falls CONTROL-P eingelesen wird, wird der Drucker umgeschaltet (EIN/AUS). Kleinbuchstaben werden in Großbuchstaben gewandelt. Nach dem Rücksprung von diesem Unterprogramm steht das eingelesene Zeichen im Akku.
TSTINP	2823	A,D,H,L	Prüft, ob das Zeichen im Akku in einer Wertetabelle enthalten ist. Die Anfangsadresse der Tabelle muß im HL-Registerpaar stehen. Die Tabelle muß mit einem Null-Byte (00) abgeschlossen sein. Wenn das Zeichen in der Tabelle steht, ist das Zero-Flag beim Rücksprung aus dem Unterprogramm gesetzt.
TSTBS	2878		Prüft, ob das Zeichen im Akku 08H (BS) oder 7FH (DEL) ist. Beim Rücksprung ist das Zero-Flag gesetzt, wenn einer der beiden Werte im Akku steht.
R4	31F4	A	Vergleicht die Inhalte der Registerpaare HL und DE. Zero-Flag , Carry-Flag HL < DE 0 1 HL = DE 1 0 HL > DE 0 0
CMPDH	31FA	A	Vergleicht die Inhalte der Registerpaare HL und DE. Zero-Flag , Carry-Flag HL < DE 0 0 HL = DE 1 0 HL > DE 0 1

Unterpr. Name	Eingangs-Adresse	veränd. Register	Funktion des Unterprogramms

TSTNUM	326F	A,B,C,D,E, H,L	Wandelt eine Dezimalzahl, die in ASCII vorliegt, in eine Binärzahl um.
			Die ASCII-Zahl muß im Speicher stehen. Wobei die höherwertigen Stellen die niedrigeren Speicherzellen belegen.
			Beispiel für die Zahl 123:
			 
			
			Der erlaubte Wertebereich reicht von 0 bis +65535.
			Führende Leerzeichen sind erlaubt.
			Die Adresse der ersten Ziffer muß beim Unterprogramm-Aufruf im DE-Registerpaar stehen.
			Die ASCII-Zahl muß mit einem nicht-numerischen Zeichen abgeschlossen sein. Die Binärzahl, die der ASCII-Zahl entspricht, befindet sich nach der Rückkehr im HL-Registerpaar. Die Anzahl der Ziffern steht im B-Register.
			Fehler: Wird keine Ziffer gefunden, so enthält das B-Register den Wert 00. Wird der zulässige Zahlenbereich überschritten, so enthält das B-Register den Wert FF. Nach dem Rücksprung wird daher folgender Fehler-Test empfohlen:
			`INR B`
			`JZ UEBERLAUF`
			`DCR B`
			`JZ KEINE-ZIFFER`
			.

Unterpr. Name	Eingangs-Adresse	veränd. Register	Funktion des Unterprogramms
TN0	3272	A,B,C,D,E, H,L	Wie TSTNUM, aber keine führenden Leerzeichen erlaubt.
DIVIDE:	3AFE	A,B,C,D,E, H,L	Dividiert den Inhalt des HL-Registerpaares durch den Inhalt des DE-Registerpaares. Das Ergebnis steht im BC-Registerpaar, der Rest im HL-Registerpaar. Keine Fehlermeldung bei Division durch Null.
PRTNUM	3C43	A,B,C,D,E, H,L	Druckt eine Zahl in der dezimalen Schreibweise. Die Zahl muß beim Unterprogrammaufruf als Zweierkomplement im HL-Registerpaar stehen (FFFF = -1) Vor positiven Zahlen wird ein Leerzeichen ausgegeben, vor negativen Zahlen wird ein "-" ausgegeben. Nach jeder Zahl wird ein Leerzeichen ausgegeben. Zahlenbereich: -32768 bis +32767.
PRTLNN	3C46	A,B,C,D,E, H,L	Wie PRTNUM. Die Zahl im HL-Registerpaar muß jedoch in normaler Binärcodierung vorliegen (FFFF = 65535). Ist beim Aufruf von PRTLNN das Sign-Bit im Flag-Register gleich Null, so wird eine positive Zahl ausgegeben. Ist das Sign-Bit gleich Eins, so wird eine negative Zahl (mit vorangestelltem "-") ausgegeben. Beim Aufruf PRTLNN+1 wird das Sign-Bit automatisch auf Null (positiv) gesetzt.
DRUSWP	3ECE		Drucker umschalten (Ein/Aus).

Unterpr. Name	Eingangs-Adresse	veränd. Register	Funktion des Unterprogramms
DROFF	3EDA		Drucker aus.
DRON	3EF3		Drucker ein.
BCLEAR	3FD6		Löscht den Bildschirm.

5.4. Systemadressen

5.4.1. SPS-Systemadressen

Die Adresse des letzten SPS-Programm-Bytes ist in den Speicherzellen E003 (LSB) und E004 (MSB) gespeichert.

Der SPS-Programmspeicher beginnt bei der Adresse E0ED.

5.4.2. BASIC-Systemadressen

Die Adresse des ersten FREIEN Bytes im BASIC-Programmspeicher ist in den Speicherzellen 6064 (LSB) und 6065 (MSB) gespeichert.

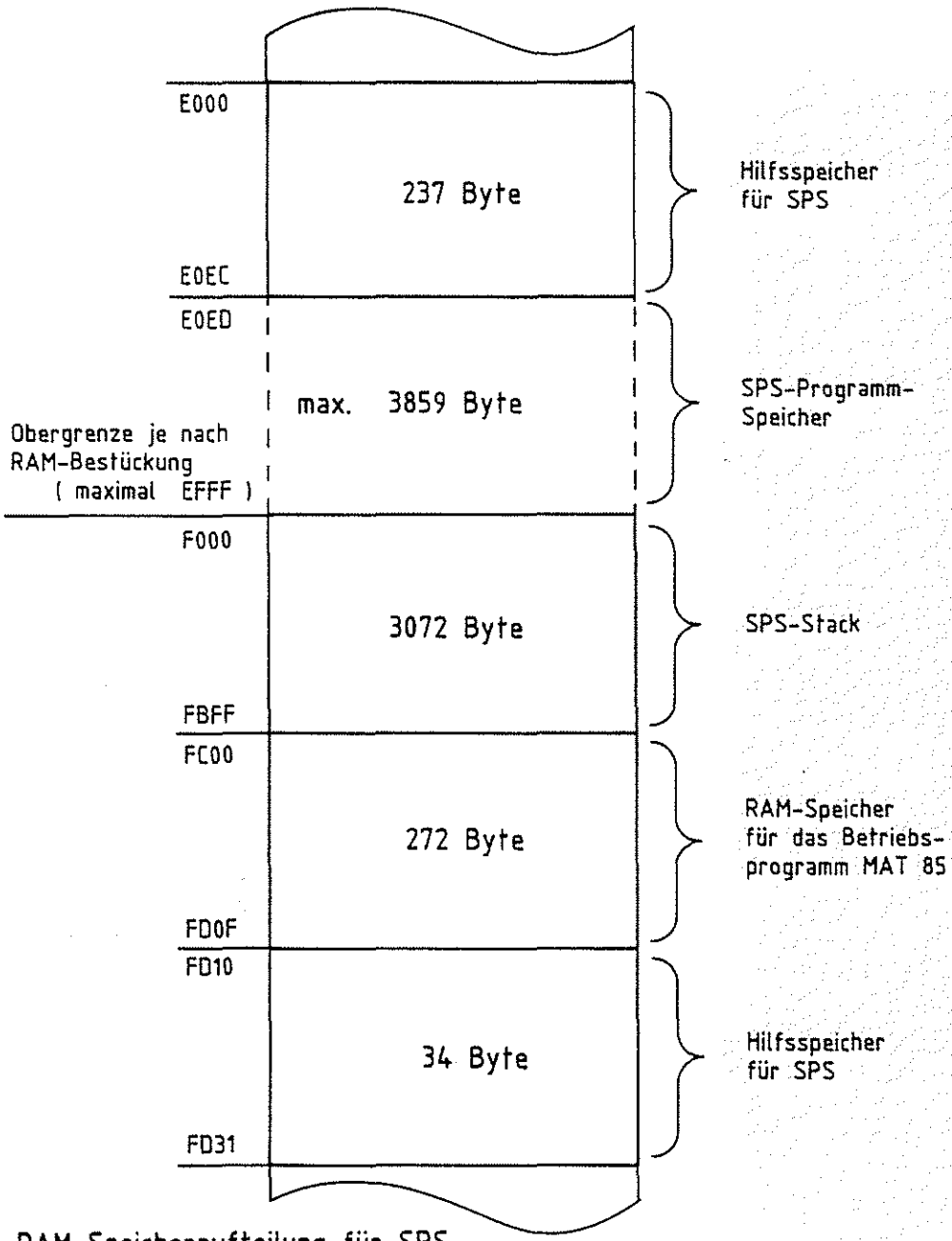
Das Ende des Programm-Speichers ist in den Speicherzellen 6066 (LSB) und 6067 (MSB) gespeichert.

Die Adresse der obersten RAM-Speicheradresse plus Eins, die von BASIC genutzt wird, ist in den Speicherzellen 606C (LSB) und 606D (MSB) gespeichert.

Speicheradressen für BASIC-Variable:

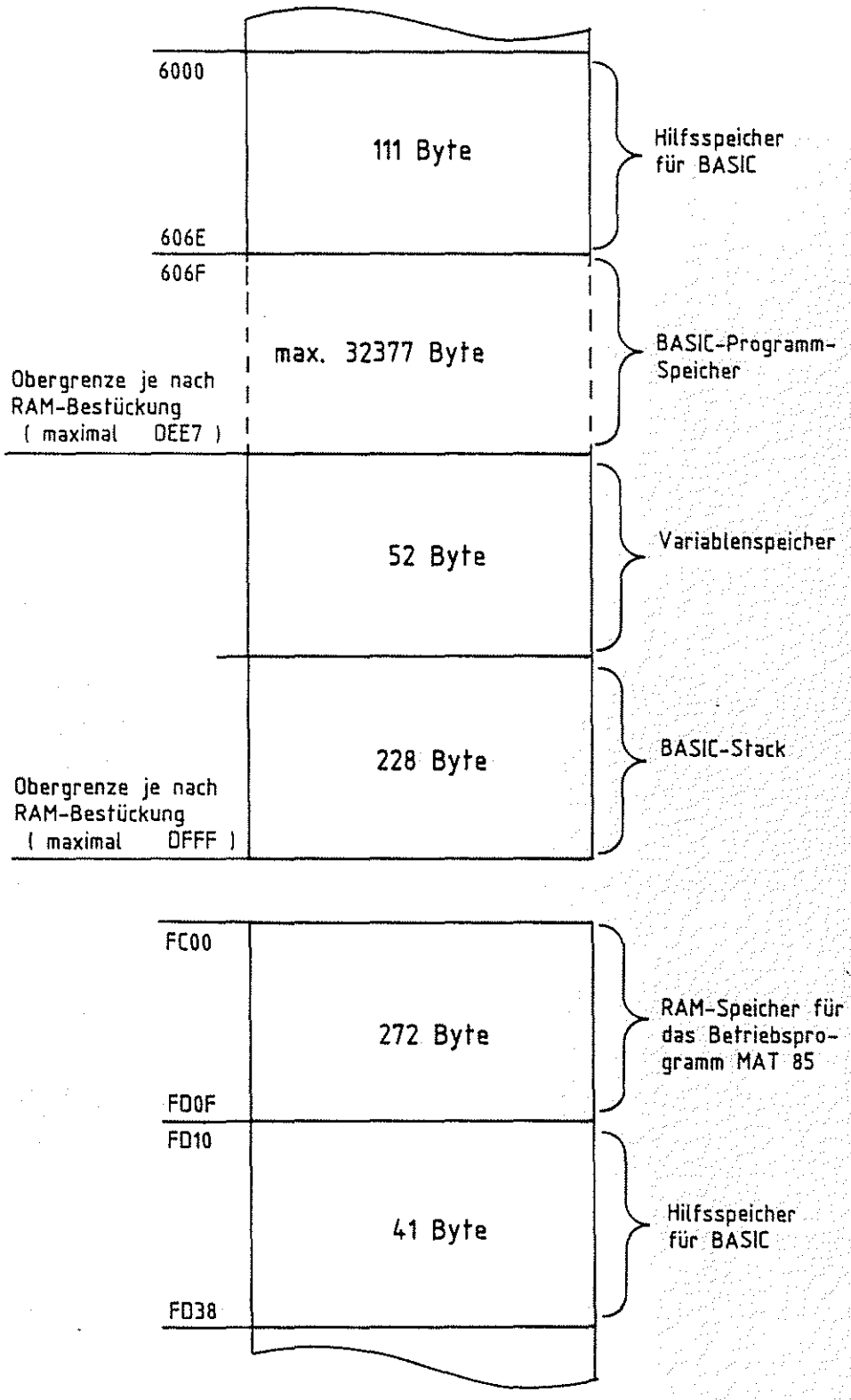
LSB	MSB
A: Ende d. Programmsp. + 2	Ende d. Programmsp. + 3
B: Ende d. Programmsp. + 4	Ende d. Programmsp. + 5
C: Ende d. Programmsp. + 6	Ende d. Programmsp. + 7
.	.
.	.
.	.
Z: Ende d. Programmsp. + 52	Ende d. Programmsp. + 53

Zahlendarstellung siehe Abschnitt 4.4.1.



RAM-Speicheraufteilung für SPS

Anhang



RAM-Speicheraufteilung für BASIC

MAT 85 - MONITOR
VERSION 1.8

```
LOC  OBJ        LINE        SOURCE STATEMENT

                    1 ;*************************************************************************
                    2 ;*                                                                       *
                    3 ;*          ****  ***** *****     MAT85 - MONITOR VERSION 1.8            *
                    4 ;*          *   * *       *       (C) COPYRIGHT 1982 BFZ/MFA             *
                    5 ;*          ****  ***     *        BERUFFOERDERUNGSZENTRUM ESSEN EV       *
                    6 ;*          *   * *       *       ALTENESSENER STR. 80-82                *
                    7 ;*          ****  *     *****     4300 ESSEN 12                          *
                    8 ;*                                                                       *
                    9 ;*************************************************************************
                   10 ;*                                                                       *
                   11 ;*              DEZEMBER 1981, SORENSEN SOFTWARE, SEEHEIM                *
                   12 ;*                                                                       *
                   13 ;*************************************************************************
                   14 ;
                   15 ;         INHALT:
                   16 ;
                   17 ;          1 GLEICHSETZUNGEN (EQU), ROM-RESERVIERUNGEN, KONSTANTEN
                   18 ;          2 KOMMANDOTABELLE
                   19 ;          3 RESET, PROGRAMM-ABORTS, KOMMANDODEKODIERER
                   20 ;          4 ROUTINEN BEI PROGRAMMABBRUCH (TRAP, RST 1, RST 7 ...)
                   21 ;          5 MONITOR-KOMMANDOS (ALPHABETISCH)
                   22 ;          6 I/O-TREIBER: CASSETTE, SERIELLE EIN-/AUSGABE
                   23 ;          7 LISTE ALLER HILFSROUTINEN
                   24 ;          8 EIN-/AUSGABEROUTINEN
                   25 ;          9 ALLGEMEINE HILFSROUTINEN
                   26 ;         10 RAM-RESERVIERUNG
                   27
                   28
```

1. GLEICHSETZUNGEN (EQU), ROM-RESERVIERUNGEN, KONSTANTEN

```
 LOC  OBJ          LINE        SOURCE STATEMENT

                    29
                    30 ;EXTERNE PROGRAMME
                    31
10E4                32 TRINIT  EQU     10E4H   ;PROGRAMM TRACER INITIALISIEREN
10F6                33 TRACE   EQU     10F6H   ;PROGRAMM TRACER
127D                34 INKLB   EQU     127DH   ;PROGRAMM INKREMENTALASSEMBLER LABELVERWALTUNG
19F9                35 DISA1   EQU     19F9H   ;PROGRAMM DISASSEMBLER INSTRUKTION AUSGEBEN
19AE                36 DISASM  EQU     19AEH   ;PROGRAMM DISASSEMBLIERE SPEICHERBEREICH
1BBB                37 LBINIT  EQU     1BBBH   ;PROGRAMM INITIALISIERE LABEL U. LINKTABELLE
                    38
                    39 ; 1. MONITOR-KONSTANTEN
                    40
0004                41 BPTANZ  EQU     4               ;ANZAHL DER MOEGLICHEN BREAKPOINTS
FFFF                42 LEER    EQU     0FFFFH          ;"LEER" Z.B. ALS SPRUNGADRESSE
                    43                                 ; ZWISCHEN 300 UND 600 BAUD
                    44
                    45
                    46 ; 2. KONTROLLZEICHEN
                    47
0007                48 BEL     EQU      7              ;KLINGEL          (BELL)
0008                49 BS      EQU      8              ;RUECKSCHRITT     (BACKSPACE)
000A                50 LF      EQU     0AH             ;ZEILENVORSCHUB   (LINE FEED)
000D                51 CR      EQU     0DH             ;WAGENRUECKLAUF   (CARRIAGE RETURN)
001B                52 ESC     EQU     1BH             ;ABBRUCH          (ESCAPE)
007F                53 RUBOUT  EQU     7FH             ;LOESCHEN         (RUB OUT/DELETE)
                    54
0001                55 FBIN    EQU     1               ;BINAER
0003                56 FHEX    EQU     3               ;HEXADEZIMAL
                    57
                    58
```

1. GLEICHSETZUNGEN (EQU), ROM-RESERVIERUNGEN, KONSTANTEN

```
LOC  OBJ          LINE        SOURCE STATEMENT
                    59
                    60
                    61                          ;     *
                    62                          ;    **
                    63                          ;   ***
                    64                          ;  ************************************
0000                65          ORG     0000    ; ******** MONITOR-ANFANGS-ADRESSE ***
                    66                          ;  ************************************
                    67                          ;   ***
                    68                          ;    **
                    69                          ;     *
                    70
                    71
0000 C34901         72 BFZ:     JMP     RESET   ;***** MONITOR START *****
0003 FF             73          DB      0FFH
0004 90FD           74 LBANF:   DW      LBRAM   ; ANFANG DER LABELTABELLE (ASSEMBLER)
0006 FF             75          DB      0FFH
0007 FF             76          DB      0FFH
0008 C33F02         77          JMP     CMDUSER ;RST 1: ANWENDER-EINSPRUNG IN MONITOR
000B FF             78          DB      0FFH
000C FF             79          DB      0FFH
000D FF             80          DB      0FFH
000E FF             81          DB      0FFH
000F FF             82          DB      0FFH
0010 C38CFC         83          JMP     RST2    ;RST 2: ANWENDER-RESTART
0013 FF             84          DB      0FFH
0014 FF             85          DB      0FFH
0015 FF             86          DB      0FFH
0016 FF             87          DB      0FFH
0017 FF             88          DB      0FFH
0018 C38FFC         89          JMP     RST3    ;RST 3: ANWENDER RESTART
001B FF             90          DB      0FFH
001C FF             91          DB      0FFH
001D FF             92          DB      0FFH
001E FF             93          DB      0FFH
001F FF             94          DB      0FFH
0020 C3DF02         95          JMP     BREAK   ;RST 4: BRAEKPOINT
0023 FF             96          DB      0FFH
0024 C38602         97          JMP     TRAP    ;NICHT-MASKIERBARER INTERRUPT (TASTER)
0027 FF             98          DB      0FFH
0028 C392FC         99          JMP     RST5    ;ANWENDER-RESTART
002B FF            100          DB      0FFH
002C C395FC        101          JMP     RST55   ;RST 5.5: ANWENDER-INTERRUPT
002F FF            102          DB      0FFH
0030 C398FC        103          JMP     RST6    ;RST 6: ANWENDER-RESTART
0033 FF            104          DB      0FFH
0034 C39BFC        105          JMP     RST65   ;RST 6.5: ANWENDER-INTERRUPT
0037 FF            106          DB      0FFH
0038 C34D02        107          JMP     ABORT   ;PROGRAMM ABORT
003B FF            108          DB      0FFH
003C C39EFC        109          JMP     RST75   ;RST 7.5: ANWENDER-INTERRUPT
003F FF            110          DB      0FFH
                   111
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT

                    112 ; JUMP-TABELLE (ZU SYSTEM-E/A-UNTERPROGRAMMEN
                    113
0040 C3AC01         114         JMP     CMD
0043 C3E00A         115         JMP     RCHAR           ;1 ZEICHEN LESEN
                    116
0046 C3FFFF         117         JMP     LEER            ;RESERVE
0049 C3FFFF         118         JMP     LEER            ;RESERVE
004C C3FFFF         119         JMP     LEER            ;RESERVE
004F C3FFFF         120         JMP     LEER            ;RESERVE
                    121
0052 C3B60B         122         JMP     WCHAR           ;1 ZEICHEN AUSGEBEN
0055 C3F70B         123         JMP     WCHARI          ;1 ZEICHEN NACH CALL AUSGEBEN
0058 C36F0B         124         JMP     WAHEX           ;2 HEX.STELLEN ASCII AUSGEBEN (A)
005B C31C0C         125         JMP     WHLHEX          ;4 HEX.STELLEN ASCII AUSGEBEN (HL)
005E C3040B         126         JMP     WABIN           ;8 BIN.WERTE ASCII AUSGEBEN (A)
0061 C3190B         127         JMP     WADEZ           ;3 DEZ.STELLEN ASCII AUSGEBEN (A)
0064 C34F0B         128         JMP     WAFOR           ;(ASCII/BIN/DEZ/HEX)-STELLEN ASCII AUSGEBEN (A)
0067 C38E0B         129         JMP     WBLANK          ;SPACE AUSGEBEN
006A C3A10B         130         JMP     WBUF            ;TEXT AUSGEBEN
006D C3B00B         131         JMP     WBUFI           ;TEXT NACH CALL AUSGEBEN
0070 C3010C         132         JMP     WCRLF           ;CR+LF AUSGEBEN
0073 C3130C         133         JMP     WCRLFI          ;CR+LF UND TEXT NACH CALL AUSGEBEN
0076 C3FFFF         134         JMP     LEER
0079 C3FFFF         135         JMP     LEER
007C C3FFFF         136         JMP     LEER
                    137
                    138 ;   COPYRIGHT NOTICE
007F 28432920       139         DB      '(C) COPYRIGHT 1982 BFZ/MFA, ESSEN, W. GERMANY
0083 434F5059
0087 52494748
008B 54203139
008F 38322042
0093 465A2F4D
0097 46412C20
009B 45535345
009F 4E2C2057
00A3 2E204745
00A7 524D414E
00AB 59202020
00AF 20202020
00B3 202020
                    140
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                   141
                   142 ;KOMMANDO-TABELLE
                   143
00B6 41535345      144 CMDTAB: DB      'ASSEMBLER',0
00BA 4D424C45
00BE 52
00BF 00
00C0 A202          145         DW      ASSEMBLER       ;INKREMENTAL-ASSEMBLER
                   146
00C2 42524541      147         DB      'BREAKPOINT',0
00C6 4B504F49
00CA 4E54
00CC 00
00CD CB02          148         DW      BREAKPOINT      ;BREAKPOINTS EIN-/AUSSCHALTEN
                   149
00CF 44495341      150         DB      'DISASSEMBLER',0
00D3 5353454D
00D7 424C4552
00DB 00
00DC 4A03          151         DW      DISASSEMBLER    ;MASCHINENCODE DISASSEMBLIEREN
                   152
00DE 474F          153         DB      'GO',0
00E0 00
00E1 7403          154         DW      GO              ;ANWENDERPROGRAMM AUSFUEHREN
                   155
00E3 48454C50      156         DB      'HELP',0
00E7 00
00E8 B503          157         DW      HELP            ;HILFE: ALLE KOMMANDOS AUSDRUCKEN
                   158
00EA 494E          159         DB      'IN',0
00EC 00
00ED D103          160         DW      INPORT          ;DATEN VON EINGABE-PORT LESEN
                   161
00EF 4C4F4144      162         DB      'LOAD TAPE',0
00F3 20544150
00F7 45
00F8 00
00F9 F403          163         DW      LOAD            ;DATEN VON BANDKASSETTE LADEN
                   164
                   165
00FB 4D454D4F      166         DB      'MEMORY',0
00FF 5259
0101 00
0102 9904          167         DW      MEMORI          ;SPEICHER DRUCKEN UND AENDERN
                   168
0104 4E455854      169         DB      'NEXT INSTRUCTION',0
0108 20494E53
010C 54525543
0110 54494F4E
0114 00
0115 0105          170         DW      NEXTINSTRUCTION ;ANWENDERPROGRAMM SCHRITTWEISE AUSFUEHREN
                   171
0117 4F5554        172         DB      'OUT',0
011A 00
011B 1505          173         DW      OUTPORT         ;DATEN ZUM AUSGABE-PORT AUSGEBEN
```

2. KOMMANDOTABELLE

```
 LOC  OBJ          LINE      SOURCE STATEMENT
                    174
011D 5052494E       175       DB      'PRINT',0
0121 54
0122 00
0123 4105           176       DW      PRINT           ;SPEICHERBEREICH AUSGEBEN
                    177
0125 52454749       178       DB      'REGISTER',0
0129 53544552
012D 00
012E 8405           179       DW      REGISTER        ;REGISTER DRUCKEN UND AENDERN
                    180
0130 53415645       181       DB      'SAVE',0
0134 00
0135 7F06           182       DW      SAVE            ;DATEN AUF BANDCASSETTE SPEICHERN TAPE
                    183
0137 54524143       184       DB      'TRACE INTERVAL',0
013B 4520494E
013F 54455256
0143 414C
0145 00
0146 1607           185       DW      TRACEINTERVAL   ;INTERVALL-TRACE EIN-/AUSSCHALTEN
                    186
0148 00             187       DB      0                       ;ENDE KOMMANDO-TABELLE
                    188
                    189
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                   190
0149 3180FC        191 RESET:  LXI     SP,MONSTK       ;STACK INITIALISIEREN
014C CD0C0E        192         CALL    BPTREM
014F CDF907        193         CALL    CASINIT         ;CASSETTEN I/O INITIALISIEREN
0152 3EC3          194         MVI     A,0C3H
0154 3280FC        195         STA     SERIN           ;ADRESSEN DER SERIELLEN ROUTINEN VORBESETZEN
0157 3283FC        196         STA     SEROUT          ;MIT DEN NORMALEN ROUTINEN
015A 21D008        197         LXI     H,SERI
015D 2281FC        198         SHLD    SERIN+1
0160 21F908        199         LXI     H,SERO
0163 2284FC        200         SHLD    SEROUT+1
                   201                                 ;POWER UP ODER WARMSTART ?
0166 21DEFC        202         LXI     H,RESBUF
0169 3E5A          203         MVI     A,5AH
016B BE            204         CMP     M
016C CA6502        205         JZ      TASTERRESET     ;WARMSTART, RAM IST SCHON INITIALISIERT
016F 77            206         MOV     M,A             ;POWER UP, BZW. KALTSTART
0170 21A1FC        207         LXI     H,BRAM          ;RAM INITIALISIEREN (=0 SETZEN)
0173 0E3D          208         MVI     C,ERAM-BRAM
0175 3600          209 RESET2: MVI     M,0
0177 23            210         INX     H
0178 0D            211         DCR     C
0179 C27501        212         JNZ     RESET2
017C 2132FC        213         LXI     H,USRSTK        ;ANWENDER-STACK INITIALISIEREN
017F 22D9FC        214         SHLD    SPWERT
0182 3E48          215         MVI     A,'H'
0184 32A1FC        216         STA     FORMAT          ;FORMAT = HEX
0187 CD600B        217         CALL    SERINIT         ;SERIELLEN I/O INITIALISIEREN
018A CD8818        218         CALL    LBINIT          ;LABELTABELLE LOESCHEN
018D CD130C        219         CALL    WCRLFI
                   220 ;
0190 42465A2D      221         DB      'BFZ-MONITOR VERSION 1.8',LF,0
0194 4D4F4E49
0198 544F5220
019C 56455253
01A0 494F4E20
01A4 312E38
01A7 0A
01A8 00
                   222
01A9 CD8503        223         CALL    HELP            ;ALLE KOMMANDOS AUSDRUCKEN
                   224 ;
                   225 ;KOMMANDO-DEKODIERER
                   226
01AC 3180FC        227 CMD:    LXI     SP,MONSTK       ;STACK INITIALISIEREN
01AF CDE410        228         CALL    TRINIT          ;TRACE INITIALISIEREN
01B2 CDC608        229         CALL    LCLR            ;ZEILENZAEHLER AUSSCHALTEN
01B5 AF            230         XRA     A               ;AKKU = 0
01B6 32C7FC        231         STA     BCKFLG          ;"+,-" ABSCHALTEN
01B9 32CAFC        232         STA     RUBFLG          ;RUBOUT EINSCHALTEN
01BC 32C9FC        233         STA     GROFLG          ;NUR GROSSBUCHSTABEN
01BF CDAA0C        234         CALL    BCLR            ;EINGABEPUFFER LOESCHEN
01C2 CD130C        235         CALL    WCRLFI
01C5 0A            236         DB      LF,0
01C6 00
```

3. RESET, PROGRAMM-ABORTS, KOMMANDO-DEKODIERER

```
LOC  OBJ          LINE        SOURCE STATEMENT
01C7 CD800B         237 CMD1:   CALL    WBUFI
01CA 0D             238         DB      CR,'        '
01CB 20202020
01CF 202020
01D2 0D             239         DB      CR,'KMD > ',0
01D3 4B4D4420
01D7 3E20
01D9 00
01DA 0601           240         MVI     B,1             ;ANZAHL DER ZEICHEN IN
01DC 0E00           241         MVI     C,0             ;  ASCII-FORMAT
01DE CD0A0D         242         CALL    BREAD           ;ZEICHEN EINLESEN UND PUFFERN
01E1 CDCD0C         243         CALL    BGETF           ;ZEICHEN HOLEN; WELCHE DA ?
01E4 DAF001         244         JC      CMDEX2          ; NEIN
01E7 21B600         245         LXI     H,CMDTAB        ; JA, KOMMANDO AUSFUEHREN
01EA 4F             246         MOV     C,A
01EB 7E             247 CMDEX1: MOV     A,M
01EC B7             248         ORA     A
01ED C20502         249         JNZ     CMDEX3
01F0 210020         250 CMDEX2: LXI     H,2000H         ;BEI NICHT VORGESEHENEM KOMMANDO
01F3 3EC3           251         MVI     A,0C3H          ;  WIRD BEI ADR. 2000HEX GEPRUEFT,
01F5 BE             252         CMP     M               ;  OB DORT EIN JMP-BEFFEHL (C3) STEHT;
01F6 CC740E         253         CZ      CALLHL          ;  WENN  J A: SPRUNG DORTHIN
01F9 CD890B         254         CALL    WBELL           ;FALSCHES KOMMANDO
01FC CDD20E         255         CALL    CRTTST          ;TTY ?
01FF D2C701         256         JNC     CMD1            ; NEIN, KDO IN GLEICHER ZEILE
0202 C3AC01         257         JMP     CMD             ; JA,   IN NAECHSTER ZEILE
0205 23             258 CMDEX3: INX     H
0206 B9             259         CMP     C               ;ZEICHEN ERKANNT ?
0207 CA1502         260         JZ      CMDEX5          ; JA, ERKANNT
020A 7E             261 CMDEX4: MOV     A,M             ; NEIN, RESTTEXT UEBERGEHEN
020B 23             262         INX     H
020C B7             263         ORA     A               ;ENDE RESTTEXT ?
020D C20A02         264         JNZ     CMDEX4          ; NEIN
0210 23             265         INX     H               ; JA, ADRESSE UEBERGEHEN
0211 23             266         INX     H
0212 C3EB01         267         JMP     CMDEX1
0215 CDA10B         268 CMDEX5: CALL    WBUF            ;RESTTEXT AUSGEBEN
0218 5E             269         MOV     E,M             ;KOMMANDOADRESSE --) (DE)
0219 23             270         INX     H
021A 56             271         MOV     D,M
021B EB             272         XCHG
021C CD740E         273         CALL    CALLHL          ;KOMMANDO AUSFUEHREN
021F DC890B         274         CC      WBELL
0222 C3AC01         275         JMP     CMD
                    276 ;
                    277
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                   278 ;
                   279 ;*********************************************************************
                   280 ;* CMDABORT -    GRUND DES PROGRAMMABBRUCHS AUSDRUCKEN
                   281 ;*       CALL     CMDABORT
                   282 ;*       DB       'GRUND DES ABBRUCHS',0
                   283 ;*********************************************************************
                   284 CMDABORT:
                   285
0225 CD0C0E        286          CALL     BPTREM
0228 CD130C        287          CALL     WCRLFI
022B 202A2A2A      288          DB       ' *** ',0
022F 20
0230 00
0231 E3            289          XTHL
0232 CDA10B        290          CALL     WBUF
0235 E3            291          XTHL
0236 CDB00B        292          CALL     WBUFI
0239 202A2A2A      293          DB       ' ***',0
023D 00
023E C9            294          RET
                   295 ;*********************************************************************
                   296 ;* CMDUSER -     ANWENDERPROGRAMM KEHRT MIT RST 1 ODER CALL 8 ZURUECK
                   297 ;*********************************************************************
                   298 CMDUSER:                                  ;*** RST 1 ***
023F CD1610        299          CALL     REGSAV
0242 CD2502        300          CALL     CMDABORT
0245 55534552      301          DB       'USER',0
0249 00
024A C3AC01        302          JMP      CMD
                   303 ;*********************************************************************
                   304 ;* ABORT -       PROGRAMM-ABSTURZ DURCH FF ALS OPCODE
                   305 ;*********************************************************************
                   306 ABORT:                                    ;*** RST 7 ***
024D CD1610        307          CALL     REGSAV
0250 CD2502        308          CALL     CMDABORT
0253 50524F47      309          DB       'PROGRAMM ABORT',0
0257 52414D4D
025B 2041424F
025F 5254
0261 00
0262 C3AC01        310          JMP      CMD
                   311 ;*********************************************************************
                   312 ;* TASTERRESET -         RESET-TASTER BETAETIGT
                   313 ;*********************************************************************
                   314 TASTERRESET:                              ;*** TASTE RESET ***
0265 3180FC        315          LXI      SP,MONSTK
0268 CDAA0C        316          CALL     BCLR
026B CD2502        317          CALL     CMDABORT
026E 52455345      318          DB       'RESET',0
0272 54
0273 00
0274 2ADFFC        319          LHLD     STIME          ; BESTIMME OB TTY ODER CRT BAUD RATE
0277 3EE8          320          MVI      A,LOW B300
0279 95            321          SUB      L
027A 3E03          322          MVI      A,HIGH B300
```

4. ROUTINEN BEI PROGRAMMABBRUCH (TRAP, RST 1, RST 7 ...)

```
LOC  OBJ          LINE        SOURCE STATEMENT
027C 9C           323         SBB     H               ; KLEINER GLEICH 300 BAUD BEDEUTET TTY
027D 3E00         324         MVI     A,0
027F 8F           325         ADC     A               ; =1 TTY, =0 CRT
0280 32C8FC       326         STA     CRTFLG
0283 C3AC01       327         JMP     CMD
                  328 ;*****************************************************************
                  329 ;* TRAP -       NICHT-MASKIERBARER INTERRUPT (TRAP)
                  330 ;*****************************************************************
                  331 TRAP:                                   ;*** TRAP ***
0286 CD1610       332         CALL    REGSAV
0289 CD6008       333         CALL    SERINIT
028C CD2502       334         CALL    CMDABORT
028F 4D4F4E49     335         DB      'MONITOR RESTART',0
0293 544F5220
0297 52455354
029B 415254
029E 00
029F C3AC01       336         JMP     CMD
                  337
                  338
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT

                    339 ;
                    340 ; ASSEMBLER -   INKREMENTALASSEMBLER
                    341 ;*************************************************************************
                    342 ;*  KMD ) "A"SSEMBLER"          A-CR ODER A-SPACE TIPPEN,
                    343 ;*                              "SSEMBLER" WIRD ERGAENZT
                    344 ;*  "START-ADR =XXXX" YYYY      XXXX=ALT, NEU: YYYY-CR ODER YYYY-SPACE
                    345 ;*                                        ALT: CR ODER SPACE
                    346 ;*       (PROGRAMM EINTIPPEN, SIEHE ASSEMBLERBESCHREIBUNG)
                    347 ;*       END CR
                    348 ;*       "KMD ) "NAECHSTES KOMMANDO
                    349 ;*************************************************************************
                    350 ASSEMBLER:
02A2 2AB4FC         351          LHLD    ABADR
02A5 CD740A         352          CALL    HSTARD          ;STARTADR. HOLEN
02A8 22B4FC         353          SHLD    ABADR
02AB CD7D12         354          CALL    INKLB           ;(HL)=ANF.ADRESSE
02AE EB             355          XCHG                    ;(HL)=LWA+1
02AF CD130C         356          CALL    WCRLFI
02B2 202A2A2A       357          DB      ' *** ',0
02B6 20
02B7 00
02B8 CD0C0A         358          CALL    HJANEIN
02BB 52455354       359          DB      'RESTART ?',0
02BF 41525420
02C3 3F
02C4 00
02C5 37             360          STC
02C6 3F             361          CMC
02C7 C0             362          RNZ                     ;NICHT "JA"
02C8 36CF           363          MVI     M,0CFH          ;"JA": RST 1 DANACH
02CA C9             364          RET
                    365
                    366 ;*************************************************************************
                    367 ;* BREAKPOINT - BREAKPOINTS AUSDRUCKEN UND EIN-/AUSSCHALTEN
                    368 ;* "KMD ) "B"REAKPOINT" B-CR ODER B-SPACE EINTIPPEN,
                    369 ;*          "REAKPOINT" WIRD ERGAENZT
                    370 ;*
                    371 ;*       "BREAK-ADR1=X1X1"      X1X1=BREAKPOINTADRESSE 1
                    372 ;*       "BREAK-ADR2=X2X2"      X2X2=BREAKPOINTADRESSE 2
                    373 ;*       "EIN/AUS   =X" Y       X=ALT, EIN: Y=E-CR ODER E-SPACE
                    374 ;*                                    AUS: Y=A-CR ODER E-SPACE
                    375 ;*                                    Y=CR ODER SPACE: UNVERAENDERT
                    376 ;*************************************************************************
                    377 BREAKPOINT:
02CB 0600           378          MVI     B,0             ;ALLE BREAKPOINTADR. AUSDRUCKEN
02CD CD090E         379          CALL    BPTNUM          ;(C) <-- ANZAHL BREAKPOINTS
02D0 CD1409         380 BREAK1:  CALL    HBPTD1
02D3 04             381          INR     B
02D4 0D             382          DCR     C
02D5 C2D002         383          JNZ     BREAK1
02D8 21C6FC         384          LXI     H,BPTFLG        ;BREAKPOINTS EIN-/AUSSCHALTEN
02DB CD7B09         385          CALL    HEINAUS         ;E=EIN, A=AUS
02DE C9             386          RET
                    387
                    388
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT

                    389
                    390
                    391 ;*************************************************************************
                    392 ;* BREAK -      BREAKPOINT-AUSFUEHRUNGSROUTINE
                    393 ;*       DIE ROUTINE WIRD AUSGEFUEHRT, WENN
                    394 ;*       (1) BREAKPOINTS EINGESCHALTET SIND  U N D
                    395 ;*           BREAKPOINTS NICHT ALLE =0 SIND  U N D
                    396 ;*           DAS PROGRAMM EINE ADRESSE ERREICHT, AUF DIE EIN
                    397 ;*             BREAKPOINT GESETZT IST:
                    398 ;*           *** BREAKPOINT ***
                    399 ;*            REGISTERAUSDRUCK
                    400 ;*       (2) EIN RST 4 IM DURCHLAUFENEN PROGRAMMCODE STEHT:
                    401 ;*           *** BREAKPOINT ERROR ***
                    402 ;*           REGISTERAUSDRUCK
                    403 ;*       DIE INSTRUKTION, BEI DER SICH EIN BREAKPOINT EREIGNET HAT,
                    404 ;*         IST NOCH NICHT AUSGEFUEHRT, D.H., DASS DER REGISTERAUSDRUCK
                    405 ;*         DAS RESULTAT DER VORANGEHENDEN INSTRUKTION ANZEIGT.
                    406 ;*************************************************************************
                    407 BREAK:
02DF CD1610         408         CALL    REGSAV          ;REGISTER RETTEN;  (HL)=PCWERT
02E2 CDC608         409         CALL    LCLR            ;ZEILENZAEHLER AUSSCHALTEN
02E5 2B             410         DCX     H               ;(PC)-1, DA INSTRUKTION NICHT AUSGEFUEHRT
02E6 22D7FC         411         SHLD    PCWERT
02E9 3AC6FC         412         LDA     BPTFLG
02EC B7             413         ORA     A               ;BREAKPOINTS EINGESCHALTET ?
02ED CA1603         414         JZ      B0              ; NEIN, RST 4 IM CODE
02F0 32C5FC         415         STA     BPTACT          ; JA
02F3 06FF           416         MVI     B,0FFH
02F5 CD370E         417         CALL    BPTSET          ;(PC)-1=BREAK-ADR ?
02F8 D21603         418         JNC     B0              ; NEIN, BREAKPOINT FEHLER
02FB CD130C         419         CALL    WCRLFI          ; JA
02FE 0A             420         DB      LF
02FF 202A2A2A       421         DB      ' *** BREAKPOINT ***',0
0303 20425245
0307 414B504F
030B 494E5420
030F 2A2A2A
0312 00
0313 C33B03         422         JMP     B1
0316 2AD7FC         423 B0:     LHLD    PCWERT          ;BREAKPOINT FEHLER:
0319 23             424         INX     H               ;  (PC)-1 UNGLEICH BREAK-ADR
031A 22D7FC         425         SHLD    PCWERT          ;  ODER BREAKPOINTS NICHT EINGESCHALTET
031D CD130C         426         CALL    WCRLFI
0320 0A             427         DB      LF
0321 202A2A2A       428         DB      ' *** BREAKPOINT ERROR ***',0
0325 20425245
0329 414B504F
032D 494E5420
0331 4552524F
0335 52202A2A
0339 2A
033A 00
033B CD0C0E         429 B1:     CALL    BPTREM          ;BREAKPOINTS ENTFERNEN
033E CDAE0F         430         CALL    PREGT           ;REGISTERINHALTE AUSDRUCKEN
0341 CD0A0F         431         CALL    PREGP
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT
 0344 CD120F        432          CALL    PREGS
 0347 CDAC01        433          CALL    CMD             ;ACHTUNG! KOMMT NIE ZURUECK
                    434
                    435 ;*****************************************************************************
                    436 ;* DISASSEMBLER -       DISASSEMBLIER MASCHINENCODE
                    437 ;*       "KMD ) "D"ISASSEMBLER"          D-CR ODER D-SPACE EINTIPPEN,
                    438 ;*                                       "ISASSEMBLER" WIRD ERGAENZT
                    439 ;*        "START-ADR =X1X1 "Y1Y1         X1X1=ALT, NEU: Y1Y1-CR ODER Y1Y1-SPACE
                    440 ;*                                                 ALT: CR ODER SPACE
                    441 ;*        "STOP -ADR =X2X2 "Y2Y2         X2X2=ALT, NEU: Y2Y2-CR ODER Y2Y2-SPACE
                    442 ;*                                                 ALT: CR ODER SPACE
                    443 ;*****************************************************************************
                    444
                    445 DISASSEMBLER:
 034A 2AB6FC        446          LHLD    DBADR
 034D CD740A        447          CALL    HSTARD          ;START-ADR HOLEN
 0350 22B6FC        448          SHLD    DBADR
 0353 EB            449          XCHG
 0354 2AB8FC        450          LHLD    DEADR
 0357 CD9C0A        451          CALL    HSTOPD          ;STOP-ADR HOLEN
 035A 22B8FC        452          SHLD    DEADR
 035D EB            453          XCHG
 035E CDCD08        454          CALL    LINIT           ;ZEILENZAEHLER EINSCHALTEN
 0361 CDAE19        455          CALL    DISASM          ;DISASSEMBLIEREN: (HL)=START, (DE)=STOP
 0364 CD010C        456          CALL    WCRLF
 0367 CD930B        457          CALL    WBLNKI
 036A 15            458          DB      21
 036B CDB00B        459          CALL    WBUFI
 036E 454E44        460          DB      'END',0
 0371 00
 0372 B7            461          ORA     A
 0373 C9            462          RET
                    463
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT

                    464 ;*****************************************************************************
                    465 ;* GO -          PROGRAMM AUSFUEHREN, ZUVOR GGF. BREAKPOINTS EINSETZEN
                    466 ;*       "KMD ) "G"O"              G-CR ODER G-SPACE EINTIPPEN,
                    467 ;*                                 "O" WIRD ERGAENZT
                    468 ;*        "START-ADR =XXXX "YYYY          XXXX=ALT, NEU: YYYY-CR ODER YYYY-SPACE
                    469 ;*                                                  ALT: CR ODER SPACE
                    470 ;*        "BREAK-ADR1=X1X1" Y1Y1          X1X1=ALT, NEU: Y1Y1-SPACE
                    471 ;*                                                  ALT: SPACE
                    472 ;*        "BREAK-ADR2=X2X2" Y2Y2          X2X2=ALT, NEU: Y2Y2-CR ODER Y2Y2-SPACE
                    473 ;*                                                  ALT: CR ODER SPACE
                    474 ;*       O D E R
                    475 ;*       "KMD ) "G"O"
                    476 ;*        "START-ADR =XXXX" YYYY
                    477 ;*        "BREAK-ADR1=X1X1" Y1Y1          X1X1=ALT, NEU: Y1Y1-CR
                    478 ;*                                                  ALT: CR
                    479 ;*       O D E R  (NUR, WENN BREAKPOINTS AUSGESCHALTET !)
                    480 ;*       "KMD ) "G"O"
                    481 ;*        "START-ADR =XXXX "YYYY
                    482 ;*****************************************************************************
                    483 GO:
0374 2AD7FC         484         LHLD    PCWERT
0377 CD740A         485         CALL    HSTARD          ;PROGRAMMSTARTADR HOLEN
037A 22D7FC         486         SHLD    PCWERT
037D 3AC6FC         487         LDA     BPTFLG
0380 B7             488         ORA     A               ;BREAKPOINTS EINGESCHALTET ?
0381 CA9403         489         JZ      GO2             ; NEIN
0384 CD090E         490         CALL    BPTNUM          ; JA, BREAK-ADRESSEN ANFORDERN
0387 0600           491         MVI     B,0
0389 CD0809         492 GO1:    CALL    HBPTD           ;BREAK-ADR HOLEN
038C C29403         493         JNZ     GO2             ; NZ: KEINE WEITEREN BREAK-ADR.S
038F 04             494         INR     B
0390 0D             495         DCR     C
0391 C28903         496         JNZ     GO1
0394 CD010C         497 GO2:    CALL    WCRLF
0397 3AC4FC         498         LDA     TRCFLG
039A B7             499         ORA     A               ;TRACE EINGESCHALTET ?
039B C2AE03         500         JNZ     GO3             ; JA
039E 32C5FC         501         STA     BPTACT          ; NEIN
03A1 CDF610         502         CALL    TRACE           ;EINE INSTRUKTION TRACEN,
03A4 DCCA07         503         CC      TRHALT
03A7 CDC50D         504         CALL    BPTINS          ; DANN BREAKPOINTS EINFUEGEN,
03AA CDF20F         505         CALL    REGRES          ; REGISTER WIEDERHERSTELLEN
03AD C9             506         RET                     ; UND KONTROLLE AN ANWENDERPROGRAMM
03AE CDCD08         507 GO3:    CALL    LINIT           ;TRACE: ZEILENZAEHLER EINSCHALTEN
03B1 CD5607         508         CALL    TRINT           ;TRACE MODUS
03B4 C9             509         RET
                    510
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                   511 ;*****************************************************************************
                   512 ;* HELP -       ALLE KOMMANDOS AUSDRUCKEN
                   513 ;*****************************************************************************
                   514 HELP:
03B5 21B600        515          LXI     H,CMDTAB          ;(HL) = ANF.ADR DER KOMMANDOTABELLE
03B8 7E            516 HELP1:   MOV     A,M
03B9 B7            517          ORA     A                 ;TABELLENENDE ?
03BA C8            518          RZ                        ; JA, EXIT
03BB CD130C        519          CALL    WCRLFI            ; NEIN, TEXT IN NEUER ZEILE AUSGEBEN
03BE 20            520          DB      ' ',0
03BF 00
03C0 7E            521 HELP2:   MOV     A,M
03C1 23            522          INX     H
03C2 B7            523          ORA     A                 ;TEXT ENDE ?
03C3 CACC03        524          JZ      HELP3             ; JA
03C6 CDB60B        525          CALL    WCHAR             ; NEIN, ZEICHEN AUSGEBEN
03C9 C3C003        526          JMP     HELP2
03CC 23            527 HELP3:   INX     H                 ;KOMMANDOADR UEBERGEHEN
03CD 23            528          INX     H
03CE C3B803        529          JMP     HELP1
                   530
                   531 ;*****************************************************************************
                   532 ;* INPORT -     EINGABEPORT LESEN
                   533 ;*      "KMD ) "I"N"              I-CR ODER I-SPACE EINTIPPEN,
                   534 ;*                                "N" WIRD ERGAENZT
                   535 ;*       "PORT-NR=X1 "Y1          X1=ALT, ALT-1: "-"
                   536 ;*                                        ALT+1: "+"
                   537 ;*                                        NEU  : Y1-CR ODER Y1-SPACE
                   538 ;*                                        ALT  : CR ODER SPACE
                   539 ;*       "DATEN  =X2 "Y2          X2=GELESENER WERT, Y2=CR    : FERTIG
                   540 ;*                                                   Y2=SPACE: NOCHMAL LESEN
                   541 ;*                                                   Y2=+/-  : NEUE PORT-ADRESSE
                   542 ;*****************************************************************************
                   543 INPORT:
03D1 3E01          544 MVI      A,1
03D3 32C7FC        545          STA     BCKFLG            ;"+"/"-" ZUSAETZLICH ALS ABSCHLUSS
03D6 CD2C0A        546 I1:      CALL    HPORTD            ;HOLE PORTADR.
03D9 C5            547 I2:      PUSH    B
03DA 0EDB          548          MVI     C,0DBH            ;"IN"-BEFEHL
03DC CD3205        549          CALL    EXPORT            ;  AUSFUEHREN (S. ROUT. EXPORT)
03DF CDA30A        550          CALL    PDATA
03E2 0600          551          MVI     B,0               ;0 ZEICHEN (NUR SCHLUSSZEICHEN VON INTEREESE)
03E4 0E03          552          MVI     C,FHEX            ;HEX.FORMAT
03E6 CD0A0D        553          CALL    BREAD
03E9 C1            554          POP     B
03EA FE20          555          CPI     ' '               ;SCHLUSSZEICHEN IST SPACE ?
03EC CAD903        556          JZ      I2                ; JA, NOCHMAL DATEN LESEN UND AUSGEBEN
03EF D2D603        557          JNC     I1                ; NEIN; "+" ODER "-": NEUE PORTADR HOLEN
03F2 B7            558          ORA     A                 ;CR: EXIT MIT CY=0
03F3 C9            559          RET
                   560
                   561
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                    562 ;*****************************************************************************
                    563 ;* LOAD -       PROGRAMM ODER DATEN VON CASSETTE LADEN
                    564 ;*       "KMD = "L"OAD"                  L-CR ODER L-SPACE EINTIPPEN,
                    565 ;*                                       "OAD" WIRD ERGAENZT
                    566 ;*        "START-ADR ="YYYY              NEU: YYYY-CR ODER YYYY-SPACE MIT
                    567 ;*                                            YYYY NICHT 0 !
                    568 ;*                                       YYYY=0: START-ADR VON CASSETTE
                    569 ;*        "SPACE, DANN BAND EINSCHALTEN" LOS GEHT'S, WENN SPACE EINGETIPPT !
                    570 ;*        "ZZZZZ"                        ZZZZZ=READY: FEHLERFREI EINGELESEN
                    571 ;*                                       ZZZZZ=ERROR: FEHLER BEIM EINLESEN
                    572 ;*                                                    CHECKSUM ODER TIME-OUT
                    573 ;*****************************************************************************
                    574
                    575 LOAD:
03F4 210000         576         LXI     H,0
03F7 CD6D0A         577         CALL    HSTART          ;LADEADR HOLEN
03FA CD130C         578         CALL    WCRLFI
03FD 20535041       579         DB      ' SPACE, DANN BAND EINSCHALTEN',0
0401 43452C20
0405 44414E4E
0409 2042414E
040D 44204549
0411 4E534348
0415 414C5445
0419 4E
041A 00
041B CDE00A         580 LSPACE: CALL    RCHAR           ;AUF SPACE WARTEN
041E FE20           581         CPI     ' '
0420 C21B04         582         JNZ     LSPACE
0423 CD010C         583         CALL    WCRLF
0426 7C             584         MOV     A,H
0427 B5             585         ORA     L               ;LADEADR = 0 ?
0428 F5             586         PUSH    PSW             ; (FLAGS RETTEN)
0429 CA2F04         587         JZ      L1              ; JA, LADEADR VON CASSETTE LESEN
042C 22BEFC         588         SHLD    LBADR           ; NEIN, LADEADR. ABSPEICHERN
042F CD2E0C         589 L1:     CALL    RCAS            ;RECORD MARK SUCHEN
0432 DA7604         590         JC      L5              ; TIME-OUT
0435 FE3A           591         CPI     ':'             ;RECORD MARK ?
0437 C22F04         592         JNZ     L1              ; NEIN, WEITER SUCHEN
043A 0600           593         MVI     B,0             ; JA;  CHECKSUM=0 ANFANGS
043C CD340C         594         CALL    RCASAHEX        ;ANZAHL BYTES LESEN
043F CA6A04         595         JZ      L3              ;LAENGE=0: ENDE-RECORD
0442 4F             596         MOV     C,A             ;ANZAHL DATENBYTES
0443 CD340C         597         CALL    RCASAHEX        ;LADEADR LESEN (2 BYTES)
0446 57             598         MOV     D,A
0447 CD340C         599         CALL    RCASAHEX
044A 5F             600         MOV     E,A             ;(DE)=LADEADR
044B F1             601         POP     PSW             ;LADEADR VON CASSETTE VERWENDEN ?
044C F5             602         PUSH    PSW
044D C25104         603         JNZ     L11             ; NEIN
0450 EB             604         XCHG                    ; JA, ADR. VON CASSETTE NACH (HL)
0451 CD340C         605 L11:    CALL    RCASAHEX        ;RECORD TYP EINLESEN
0454 B7             606         ORA     A
0455 C27604         607         JNZ     L5              ;NZ: TYP UNGLEICH 0
0458 CD340C         608 L2:     CALL    RCASAHEX        ;DATENBYTES EINLESEN
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
045B 77           609         MOV     M,A             ;  UND AB LADEADR ABSPEICHERN
045C 23           610         INX     H               ;  SOLANGE, BIS
045D 0D           611         DCR     C               ;  ALLE IM RECORD ERFASST
045E C25804       612         JNZ     L2
0461 CD340C       613         CALL    RCASAHEX        ;CHECKSUM EINLESEN
0464 D22F04       614         JNC     L1              ; NC: CHECKSUM O.K., NAECHSTER RECORD
0467 C37604       615         JMP     L5              ; CHECKSUM FEHLER !
046A 0E04         616 L3:     MVI     C,4             ;ENDE RECORD OHNE WEITERE
046C CD340C       617 L4:     CALL    RCASAHEX        ;  KONTROLLE EINLESEN (BIS AUF CHECKSUM!)
046F 0D           618         DCR     C
0470 C26C04       619         JNZ     L4
0473 D28C04       620         JNC     L6              ; NC: CHECKSUM O.K.
0476 CD130C       621 L5:     CALL    WCRLFI          ; FEHLERNACHRICHT
0479 20434845     622         DB      ' CHECKSUM ERROR',0
047D 43484553
0481 4D204552
0485 524F52
0488 00
0489 F1           623         POP     PSW
048A 37           624         STC
048B C9           625         RET
048C CD130C       626 L6:     CALL    WCRLFI          ;ALLES KORREKT EINGELESEN
048F 20524541     627         DB      ' READY',0
0493 4459
0495 00
0496 F1           628         POP     PSW
0497 B7           629         ORA     A
0498 C9           630         RET
                  631
```

```
LOC  OBJ          LINE        SOURCE STATEMENT

                   632 ;*****************************************************************************
                   633 ;* MEMORI -     SPEICHERINHALTE ANZEIGEN UND AENDERN
                   634 ;*       "KMD ) "M"EMORY"                ;M-CR ODER M-SPACE EINTIPPEN
                   635 ;*                                       "EMORY" WIRD ERGAENZT
                   636 ;*        "START-ADR =XXXX "YYYY         XXXX=ALT, NEU: YYYY-CR ODER YYYY-SPACE
                   637 ;*                                              ALT: CR ODER SPACE
                   638 ;*        "FORMAT    =X "Y               X=ALT, NEU: Y-CR ODER Y-SPACE
                   639 ;*                                              ALT: CR ODER SPACE
                   640 ;*                                       Y=A: ASCII (DRUCKBARE ZEICHEN)
                   641 ;*                                        =B: BINAER (0,1)
                   642 ;*                                        =D: DEZIMAL (0...9)
                   643 ;*                                        =H: HEXADEZIMAL (0...9,A...F)
                   644 ;*              BEISPIELHAFT WEITER FUER YYYY=1000
                   645 ;*        "1000 C3 "            SPACE:          UNVERAENDERT, ADR+1
                   646 ;*        "1001 20 "22          22-SPACE:       VERAENDERT,   ADR+1
                   647 ;*        "1002 44              SPACE:          UNVERAENDERT, ADR+1
                   648 ;*        "1003 55 "-           -:              UNVERAENDERT, ADR-1
                   649 ;*        "1002 44 "54-         54-:            VERAENDERT,   ADR-1
                   650 ;*        "1001 22 "            +:              UNVRAENDERT, ADR+1
                   651 ;*        "1002 44 "            CR:             UNVERAENDERT, FERTIG
                   652 ;*       "KMD ) "NAECHSTES KOMMANDO
                   653 ;*****************************************************************************
                   654
                   655 MEMORI:
0499 2AAEFC        656          LHLD    MBADR
049C CD740A        657          CALL    HSTARD          ;STARTADR HOLEN
049F CDC509        658          CALL    HFORMD          ;DATENFORMAT (A,B,D,H) HOLEN
04A2 3E01          659          MVI     A,1
04A4 32C7FC        660          STA     BCKFLG          ;ADR VORWAERTS/RUECKWAERTS DURCH "+"/"-"
04A7 32C9FC        661          STA     GROFLG          ;KLEINBUCHSTABEN BEI ASCII-FORMAT ZULASSEN
04AA 22AEFC        662 M0:      SHLD    MBADR
04AD CD130C        663          CALL    WCRLFI
04B0 20            664          DB      ' ',0
04B1 00
04B2 CD270C        665          CALL    WHLHXB
04B5 7E            666          MOV     A,M             ;SPEICHERWERT ENTSPRECHEND
04B6 CD680B        667          CALL    WAFORB          ;  DATENFORMAT AUSDRUCKEN
04B9 E5            668          PUSH    H
04BA 59            669          MOV     E,C
04BB 213218        670          LXI     H,1832H         ;ANZAHL ZIFFERN: 1=A, 8=B, 3=D, 2=H
04BE 29            671 M1:      DAD     H
04BF 17            672          RAL
04C0 29            673          DAD     H
04C1 17            674          RAL
04C2 29            675          DAD     H
04C3 17            676          RAL
04C4 29            677          DAD     H
04C5 17            678          RAL
04C6 1D            679          DCR     E
04C7 F2BE04        680          JP      M1
04CA E60F          681          ANI     0FH
04CC 47            682          MOV     B,A             ;(B)=ANZAHL DER ZIFFERN ENTSPRECHEND FORMAT
04CD E1            683          POP     H
04CE CD0A0D        684          CALL    BREAD           ;ZIFFERN EINLESEN
04D1 F5            685          PUSH    PSW
```

```
LOC  OBJ          LINE        SOURCE STATEMENT

04D2 CDDB0C        686         CALL    BGETL
04D5 DAEB04        687         JC      M4              ; C: NUR SCHLUSSZEICHEN, SPEICHERINHALT BLEIBT
04D8 110000        688         LXI     D,0             ; SPEICHERINHALT AENDERN
04DB CD7110        689         CALL    ZUFOR           ;ZIFFERN ENTSPR. FORMAT WANDELN
04DE CDDB0C        690 M2:     CALL    BGETL           ;  ENSPRECHEND FORMAT UMWANDELN
04E1 DAEA04        691         JC      M3              ;  UND ABSPEICHERN
04E4 CD7110        692         CALL    ZUFOR
04E7 C3DE04        693         JMP     M2
04EA 72            694 M3:     MOV     M,D
04EB F1            695 M4:     POP     PSW
04EC FE0D          696         CPI     CR              ;SCHLUSSZEICHEN = CR ?
04EE CAFF04        697         JZ      M5              ; JA, FERTIG
04F1 FE2C          698         CPI     '+'+1           ; NEIN; ZEICHEN = "-" ?
04F3 23            699         INX     H
04F4 FAAA04        700         JM      M0              ;  NEIN, ADR = ADR+1
04F7 2B            701         DCX     H               ;  JA,   ADR = ADR-1
04F8 2B            702         DCX     H
04F9 CDB60B        703         CALL    WCHAR           ;"-" MARKIERT ADR. RUECKWAERTS
04FC C3AA04        704         JMP     M0
04FF B7            705 M5:     ORA     A
0500 C9            706         RET
                   707
                   708 ;*************************************************************************
                   709 ;* NEXTINSTRUCTION -    ANZAHL VON INSTRUKTIONEN AUSFUEHREN
                   710 ;* "KMD > "N"EXT INSTRUCTIONS"  N-CR ODER N-SPACE EINTIPPEN,
                   711 ;*                                      "EXT INSTRUCTIONS" WIRD ERGAENZT
                   712 ;*       "START-ADR =XXXX "YYYY         XXXX=ALT, NEU: YYYY-CR ODER YYYY-SPACE
                   713 ;*                                            ALT: CR ODER SPACE
                   714 ;*       "STEPS     =XX "YY             XX=ALT, NEU: YY-CR ODER YY-SPACE
                   715 ;*                                          ALT: CR ODER SPACE
                   716 ;*             DANACH WERDEN DIE XX BZW. YY NACHFOLGENDEN INSTRUKTIONEN
                   717 ;*             AUSGEFUEHRT. NACH JEDER INSTRUKTION WERDEN ALLE REGISTER
                   718 ;*             AUSGEDRUCKT. DANACH:
                   719 ;*      "KMD > "NAECHSTES KOMMANDO
                   720 ;*************************************************************************
                   721
                   722 NEXTINSTRUCTION:
0501 2AD7FC        723         LHLD    PCWERT
0504 CD740A        724         CALL    HSTARD          ;START-ADR HOLEN
0507 22D7FC        725         SHLD    PCWERT
050A CD7B0A        726         CALL    HSTEPD          ;ANZAHL DER SCHRITTE HOLEN
050D CDCD08        727         CALL    LINIT           ;ZEILENZAEHLER EINSCHALTEN
0510 CD3207        728         CALL    TRSTEP          ;ANGEGEBENE ZAHL VON INSTRUKTIONEN AUSFUEHREN
0513 B7            729         ORA     A
0514 C9            730         RET
                   731
```

```
LOC  OBJ          LINE        SOURCE STATEMENT

                   732 ;*****************************************************************************
                   733 ;* OUTPORT -    DATEN UEBER I/O-PORT AUSGEBEN
                   734 ;*       "KMD ) "O"UT"                    ;O-CR ODER O-SPACE EINTIPPEN,
                   735 ;*                                        "UT" WIRD ERGAENZT
                   736 ;*        "PORT-NR=X1 "Y1                 X1=ALT, ALT-1: "-"
                   737 ;*                                                ALT+1: "+"
                   738 ;*                                                NEU  : Y1-CR ODER Y1-SPACE
                   739 ;*                                        ALT  : CR ODER SPACE
                   740 ;*        "DATEN  =X2 "Y2                 X2=ALT, NEU: Y2-CR (FERTIG)
                   741 ;*                                                    Y2-SPACE (NOCHMAL)
                   742 ;*                                                ALT: SPACE
                   743 ;*                                        Y2=CR   : FERTIG
                   744 ;*                                        Y2=+/-  : NEUE PORT-ADR
                   745 ;*                                        Y2=SPACE: NOCHMAL SCHREIBEN
                   746 ;*****************************************************************************
                   747 OUTPORT:
0515 3E01          748         MVI     A,1
0517 32C7FC        749         STA     BCKFLG          ;ADR VORWAERTS/RUECKWAERTS DURCH "+"/"-"
051A CD2C0A        750 O1:     CALL    HPORTD          ;PORT-ADR HOLEN
051D CD6E09        751 O2:     CALL    HDATAD          ;DATEN HOLEN
0520 F5            752         PUSH    PSW             ;ENDEZEICHEN RETTEN
0521 7A            753         MOV     A,D
0522 0ED3          754         MVI     C,0D3H          ;"OUT"-BEFEHL
0524 CD3205        755         CALL    EXPORT
0527 F1            756         POP     PSW
0528 FE20          757         CPI     ' '             ;SPACE ?
052A CA1D05        758         JZ      O2              ; JA, NEUE DATEN
052D D21A05        759         JNC     O1              ; NEIN, "+" ODER "-", NEUE PORT-ADR
0530 B7            760         ORA     A               ; NEIN, CR, FERTIG
0531 C9            761         RET
                   762
                   763 EXPORT:                         ;PORT-BEFEHL AUSFUEHREN
0532 E5            764         PUSH    H               ;PORT-BEFEHL (IN/OUT ADR) AUF STACK
0533 21C900        765         LXI     H,00C9H
0536 E5            766         PUSH    H               ;RET-BEFEHL AUF STACK
0537 C5            767         PUSH    B
0538 6C            768         MOV     L,H             ;(L) = 0
0539 39            769         DAD     SP
053A CD740E        770         CALL    CALLHL          ;IN/OUT-BEFEHL WIRD AUF STACK AUSGEFUEHRT !
053D E1            771         POP     H
053E E1            772         POP     H
053F E1            773         POP     H
0540 C9            774         RET
                   775
```

```
LOC  OBJ          LINE        SOURCE STATEMENT

                   776 ;*****************************************************************************
                   777 ;* PRINT -      SPEICHERBEREICH AUSDRUCKEN
                   778 ;*       "KMD ) "P"RINT"                  P-CR ODER P-SPACE EINTIPPEN,
                   779 ;*                                        "RINT" WIRD ERGAENZT
                   780 ;* "START-ADR =X1X1 "Y1Y1                 X1X1=ALT, NEU: Y1Y1-CR ODER Y1Y1-SPACE
                   781 ;*                                                  ALT: CR ODER SPACE
                   782 ;*       "STOP -ADR =X2X2 "Y2Y2           X2X2=ALT, NEU: Y2Y2-CR ODER Y2Y2-SPACE
                   783 ;*                                                  ALT: CR ODER SPACE
                   784 ;*       "FORMAT    =X "Y                 X=ALT, NEU: Y-CR ODER Y-SPAE
                   785 ;*                                                 ALT: CR ODER SPACE
                   786 ;*                                        Y=A: ASCII (DRUCKBARE ZEICHEN)
                   787 ;*                                         =B: BINAER (0,1)
                   788 ;*                                         =D: DEZIMAL (0...9)
                   789 ;*                                         =H: HEXADEZIMAL (0...9,A...F)
                   790 ;*              (AUSDRUCK ASCII/BIN/DEZ/HEX)
                   791 ;*****************************************************************************
                   792 PRINT:
0541 2A80FC        793         LHLD    PBADR           ;ALTE ADRESSE
0544 CD740A        794         CALL    HSTARD          ;STARTADR. HOLEN
0547 2280FC        795         SHLD    PBADR           ;NEUE ADR. ODER UNVERAENDERT
054A EB            796         XCHG
054B 2A82FC        797         LHLD    PEADR
054E CD9C0A        798         CALL    HSTOPD          ;STOPADR. HOLEN
0551 2282FC        799         SHLD    PEADR
0554 CD3910        800         CALL    SUB2            ;(HL) = ANZAHL DER AUSZUDRUCKENDEN
0557 D8            801         RC                      ;       SPEICHERINHALTE
0558 EB            802         XCHG
0559 13            803         INX     D
055A CDC509        804         CALL    HFORMD          ;ZAHLENFORMAT HOLEN
055D CDCD08        805         CALL    LINIT
0560 0607          806         MVI     B,00000111B     ;MASKE FUER NEUE ZEILE
0562 79            807         MOV     A,C             ;   8 WERTE PRO ZEILE BEI ASCII, DEZ, HEX
0563 3D            808         DCR     A               ;BINAER ?
0564 C26905        809         JNZ     P1              ; NEIN
0567 0600          810         MVI     B,00000000B     ; JA, EIN BIN.WERT PRO ZEILE
0569 CD010C        811 P1:     CALL    WCRLF
056C CD8E0B        812         CALL    WBLANK
056F CD270C        813         CALL    WHLHXB          ;SPEICHERADR. AUSDRUCKEN
0572 7E            814 P2:     MOV     A,M             ;SPEICHERWERT HOLEN
0573 CD680B        815         CALL    WAFORB          ;WERT WANDELN UND AUSGEBEN
0576 1B            816         DCX     D
0577 7A            817         MOV     A,D
0578 B3            818         ORA     E               ;ALLES AUSGEDRUCKT ?
0579 C8            819         RZ                      ; JA
057A 23            820         INX     H               ; NEIN, NAECHSTEN WERT
057B 7D            821         MOV     A,L
057C A0            822         ANA     B               ;NEUE ZEILE ?
057D C27205        823         JNZ     P2              ; NEIN
0580 C36905        824         JMP     P1              ; JA
0583 C9            825         RET
                   826
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT

                    827 ;*************************************************************************
                    828 ;* REGISTER -   REGISTERINHALTE DRUCKEN UND AENDERN
                    829 ;*       "KMD ) "R"EGISTER"              R-CR ODER R-SPACE EINTIPPEN,
                    830 ;*                                       "EGISTER WIRD ERGAENZT
                    831 ;*        "PC   LABEL:  OP   ADR.FELD  A  NZHPC  B  C  D  E  H  L  I  SP
                    832 ;*         1022 L1:     LXI  H,WERT    12 10110  22 FF 23 AA 12 34 00 F000
                    833 ;*
                    834 ;*              AENDERUNGEN DER REGISTER WERDEN SO VORGENOMMEN:
                    835 ;*              (1) SPACE BEENDET AENDERUNG EINES REGISTERS
                    836 ;*              (2) CR    BEENDET ALLE EINGABEN: NAECHSTES KOMMANDO FOLGT
                    837 ;*              (3) EINGEGEBENE HEZ.ZIFFERN AENDERN NUR DIE STELLE, AN DER
                    838 ;*                  DIE ZIFFER EINGEGEBEN WURDE.
                    839 ;*       BEISPIEL:
                    840 ;*         PC   LABEL:  OP   ADR.FELD  A  NZHPC  B  C  D  E  H  L  I  SP
                    841 ;*         1022 L1:     LXI  H,WERT    12 10110  22 FF 23 AA 12 34 00 F000
                    842 ;*         2000                        0  0001      1  55 B           F1
                    843 ;*       KMD ) REGISTER
                    844 ;*         PC   LABEL:  OP   ADR.FELD  A  NZHPC  B  C  D  E  H  L  I  SP
                    845 ;*         2000 L1:     LXI  H,WERT    02 00010  22 1F 55 BA 12 34 00 F100
                    846 ;*************************************************************************
                    847 REGISTER:
0584 3E01           848         MVI     A,1
0586 32CAFC         849         STA     RUBFLG          ;RUBOUT FUER TTY AUSSCHALTEN
0589 CDAE0F         850         CALL    PREGT           ;TITEL DRUCKEN
058C 217306         851         LXI     H,REGTAB        ;REGISTER-TABELLE
058F E5             852         PUSH    H
0590 CD1B0F         853         CALL    PREGS0          ;ALLE REGISTER AUSDRUCKEN
0593 E1             854         POP     H
0594 7E             855 REG1:   MOV     A,M             ;(A) = REGISTER TYP (S. REGTAB)
0595 23             856         INX     H
0596 B7             857         ORA     A               ;REGISTER TYP = 0 ?
0597 C8             858         RZ                      ; JA,FERTIG
0598 E5             859         PUSH    H               ; NEIN
0599 21CFFC         860         LXI     H,REGW          ;(HL)=ADR. DER AKTUELLEN REGISTERLISTE
059C 5F             861         MOV     E,A             ;TABELLENINDEX ERRECHNEN FUER
059D E60F           862         ANI     0FH             ;  DIE FAELLE:
059F 4F             863         MOV     C,A             ;  (1) 8BIT REGISTER (A,B,C,...)
05A0 0600           864         MVI     B,0             ;  (2) FLAG REGISTER 5BIT BINAER
05A2 09             865         DAD     B               ;  (3) 16BIT REGISTER
05A3 7B             866         MOV     A,E             ;  (4) 16BIT REGISTER AM ZEILENANFANG
05A4 EB             867         XCHG
05A5 07             868         RLC
05A6 07             869         RLC
05A7 E603           870         ANI     3
05A9 21BA05         871         LXI     H,RTAB          ;ENTSPRECHEND REG.TYP UEBER TABELLE SPRINGEN
05AC 4F             872         MOV     C,A
05AD CD750E         873         CALL    CALLTB
05B0 E1             874         POP     H
05B1 FE0D           875         CPI     CR              ;CR = SCHLUSSZEICHEN ?
05B3 C8             876         RZ                      ; JA, FERTIG
05B4 CDBE0B         877         CALL    WBLANK          ; NEIN
05B7 C39405         878         JMP     REG1
05BA C205           879 RTAB:   DW      R2HEX           ; 8BIT HEX
05BC D005           880         DW      R5CC            ; 5BIT BIN; FLAGS
05BE 1A06           881         DW      R4HEX           ;16BIT HEX
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
05C0 6606          882          DW      R4HEXPC         ;16BIT HEX PC
                   883
                   884 R?HEX:                           ;ZWEI HEX.BYTES EINLESEN
05C2 0602          885          MVI     B,2             ;2 BYTES
05C4 0E03          886          MVI     C,FHEX          ;HEX.FORMAT
05C6 CD2D06        887          CALL    R4HEXO          ;NEUEN WERT EINLESEN
05C9 F5            888          PUSH    PSW
05CA 0A            889          LDAX    B               ;NUR DIE EINGEGEBENEN STELLEN
05CB A3            890          ANA     E               ;  AENDERN, UND ZWAR LINKSBUENDIG
05CC B5            891          ORA     L
05CD 02            892          STAX    B
05CE F1            893          POP     PSW
05CF C9            894          RET
                   895
                   896 R5CC:                            ;FUENF BIN.BYTES EINLESEN
05D0 0605          897          MVI     B,5             ;5 BYTES
05D2 0E01          898          MVI     C,FBIN          ;BIN.FORMAT
05D4 CD0A0D        899          CALL    BREAD
05D7 F5            900          PUSH    PSW
05D8 CD0B0C        901          CALL    BGETL           ;ZIFFER EINGEGEBEN ?
05DB 3F            902          CMC
05DC D21406        903          JNC     R5CC4           ; NEIN, FERTIG (WERT UNVERAENDERT)
05DF E601          904          ANI     1               ; JA
05E1 6F            905          MOV     L,A             ;(L) = ZIFFERN WERT (0 ODER 1)
05E2 06AB          906          MVI     B,10101011B     ;FLAG-REG. MASKE
05E4 0E04          907          MVI     C,4             ;NOCH MAX. 4 EINGEGEBENE BYTES
05E6 CD0B0C        908 R5CC1:   CALL    BGETL           ;ZIFFER AUS PUFFER HOLEN; WAR NOCH WAS DA ?
05E9 D2F405        909          JNC     R5CC2           ; JA
05EC CD0E0B        910          CALL    WBLANK          ; NEIN, SPACE AUSGEBEN, DAMIT
05EF 78            911          MOV     A,B             ;       CURSOR RICHTIG STEHT
05F0 E67F          912          ANI     7FH             ;OBERSTES MASKENBIT IN (B) LOESCHEN
05F2 47            913          MOV     B,A
05F3 AF            914          XRA     A               ;BIT WERT = 0
05F4 E601          915 R5CC2:   ANI     1
05F6 29            916          DAD     H               ;BITWERT (=0/1) DURCH LINKSSCHIFT
05F7 CAFB05        917          JZ      R5CC3           ;  IN (L) REINSCHIEBEN
05FA 2C            918          INR     L
05FB 29            919 R5CC3:   DAD     H
05FC 78            920          MOV     A,B
05FD 07            921          RLC
05FE 07            922          RLC
05FF 47            923          MOV     B,A
0600 0D            924          DCR     C               ;FERTIG ?
0601 C2E605        925          JNZ     R6CC1           ; NEIN, WEITERE STELLEN
0604 0F            926          RRC                     ; JA, BIT-MASKEN ARRANGIEREN
0605 47            927          MOV     B,A
0606 7C            928          MOV     A,H
0607 0F            929          RRC
0608 7D            930          MOV     A,L
0609 1F            931          RAR
060A EB            932          XCHG
060B A0            933          ANA     B
060C 4F            934          MOV     C,A
060D 78            935          MOV     A,B
060E 2F            936          CMA
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
060F A6            937          ANA     M              ;VON LINKS BEGINNEND WERDEN NUR SO VIELE
0610 B1            938          ORA     C              ;  FLAGS GEAENDERT, WIE BINAERE STELLEN
0611 77            939          MOV     M,A            ;  EINGEGEBEN WURDEN
0612 F1            940          POP     PSW
0613 C9            941          RET
0614 CD930B        942 R5CC4:   CALL    WBLNKI         ;CURSOR RICHTIG POSITIONIEREN
0617 05            943          DB      5
0618 F1            944          POP     PSW
0619 C9            945          RET
                   946 R4HEX:                          ;4 HEX.BYTES EINLESEN
061A 0604          947          MVI     B,4            ;4 BYTES
061C 0E03          948          MVI     C,FHEX         ;HEX.FORMAT
061E CD2D06        949          CALL    R4HEX0         ;HEX.WERT EINLESEN
0621 F5            950          PUSH    PSW            ;  (HL)=HEX.WERT, (DE)=MASKE FUER
0622 0A            951          LDAX    B              ;  GEAENDERTE STELLEN
0623 A3            952          ANA     E
0624 B5            953          ORA     L              ;ALTEN WERT HOLEN, ZU AENDERNDE
0625 02            954          STAX    B              ;  STELLEN ZU 0 MASKIEREN,
0626 03            955          INX     B              ;  NEUEN WERT DRAUF ODERN UND
0627 0A            956          LDAX    B              ;  DIESEN WERT WEGSPEICHERN
0628 A2            957          ANA     D
0629 B4            958          ORA     H
062A 02            959          STAX    B
062B F1            960          POP     PSW
062C C9            961          RET
                   962
062D CD0A0D        963 R4HEX0:  CALL    BREAD          ;NEUEN WERT EINLESEN
0630 F5            964          PUSH    PSW
0631 D5            965          PUSH    D
0632 210000        966          LXI     H,0
0635 54            967          MOV     D,H
0636 5C            968          MOV     E,H
0637 CD0B0C        969 R4HEX1:  CALL    BGETL          ;ENTSPRECHEND DER EINGEGEBENEN
063A D24C06        970          JNC     R4HEX3         ;  STELLEN WIRD DER NEUE WERT
063D CD8E0B        971          CALL    WBLANK         ;  LINKSBUENDIG IN (HL) GEBILDET;
0640 EB            972          XCHG                   ;  IN (DE) WIRD PARALLEL EINE
0641 0E04          973          MVI     C,4            ;  MASKE GEBILDET:
0643 29            974 R4HEX2:  DAD     H              ;  1111 = GEAENDERTE STELLE
0644 2C            975          INR     L              ;  0000 = UNGEAENDERTE STELLE
0645 0D            976          DCR     C
0646 C24306        977          JNZ     R4HEX2
0649 EB            978          XCHG
064A 3E30          979          MVI     A,'0'
064C CD8410        980 R4HEX3:  CALL    ZUHEX
064F 87            981          ADD     A
0650 87            982          ADD     A
0651 87            983          ADD     A
0652 87            984          ADD     A
0653 0E04          985          MVI     C,4
0655 29            986 R4HEX4:  DAD     H
0656 87            987          ADD     A
0657 D25B06        988          JNC     R4HEX5
065A 2C            989          INR     L
065B 0D            990 R4HEX5:  DCR     C
065C C25506        991          JNZ     R4HEX4
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
065F 05            992          DCR     B
0660 C23706        993          JNZ     R4HEX1
0663 C1            994          POP     B              ;(HL)=NEUER WERT LINKSBUENDIG
0664 F1            995          POP     PSW            ;(DE)=MASKE GEAENDERTER STELLEN
0665 C9            996          RET
                   997
                   998 R4HEXPC:                              ;VIER HEX.BYTES LESEN; VORHER CR+LF
0666 CD130C        999          CALL    WCRLFI
0669 20           1000          DB      ' ',0
066A 00
066B CD1A06       1001          CALL    R4HEX          ;SONST WIE R3-ROUTINE
066E CD930B       1002          CALL    WBLNKI         ;OPCODE UEBERSPRINGEN
0671 17           1003          DB      23
0672 C9           1004          RET
                  1005
0001              1006 AREG     EQU     1              ;ZAHLEN GEBEN POSITION IN REGISTER-
0000              1007 FREG     EQU     0              ;  SPEICHER AN
0003              1008 BREG     EQU     3
0002              1009 CREG     EQU     2
0005              1010 DREG     EQU     5
0004              1011 EREG     EQU     4
0007              1012 HREG     EQU     7
0006              1013 LREG     EQU     6
000C              1014 IREG     EQU     12
0008              1015 PREG     EQU     8
000A              1016 SREG     EQU     10
                  1017
0673 D8           1018 REGTAB:  DB      PREG+0D0H      ;HEX, 4 BYTES (PC); CR+LF+SPACE VORHER
0674 01           1019          DB      AREG+0         ;HEX, 2 BYTES
0675 40           1020          DB      FREG+040H      ;BINAER, 5BIT
0676 03           1021          DB      BREG+0         ;HEX, 2 BYTES
0677 02           1022          DB      CREG+0         ;HEX, 2 BYTES
0678 05           1023          DB      DREG+0         ;HEX, 2 BYTES
0679 04           1024          DB      EREG+0         ;HEX, 2 BYTES
067A 07           1025          DB      HREG+0         ;HEX, 2 BYTES
067B 06           1026          DB      LREG+0         ;HEX, 2 BYTES
067C 0C           1027          DB      IREG+0         ;HEX, 2 BYTES
067D 8A           1028          DB      SREG+080H      ;HEX, 4 BYTES
067E 00           1029          DB      0              ;***TABELLENENDE
                  1030
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT
                   1031 ;*************************************************************************;*
                   1032 ;*       SAVE -  SPEICHERINHALT AUF CASSETTE AUSGEBEN
                   1033 ;*       "KMD ) "S"AVE"                  S-CR ODER S-SPACE EINTIPPEN,
                   1034 ;*                                       "AVE" WIRD ERGAENZT
                   1035 ;*        "START-ADR = X1X1" Y1Y1        X1X1=ALT, NEU: Y1Y1-CR ODER Y1Y1-SPACE
                   1036 ;*                                                 ALT: CR ODER SPACE
                   1037 ;*        "STOP -ADR = X2X2" Y2Y2        X2X2=ALT,NEU: Y2Y2-CR ODER Y2Y2-SPACE
                   1038 ;*                                                 ALT: CR ODER SPACE
                   1039 ;*        "BAND EINSCHALTEN, DANN SPACE" NACH SPACE GEHTS LOS
                   1040 ;*************************************************************************
                   1041
                   1042 SAVE:
067F 2AC0FC        1043          LHLD    SBADR
0682 CD740A        1044          CALL    HSTARD          ;START-ADR HOLEN
0685 22C0FC        1045          SHLD    SBADR
0688 EB            1046          XCHG
0689 2AC2FC        1047          LHLD    SEADR
068C CD9C0A        1048          CALL    HSTOPD          ;STOP-ADR HOLEN
068F 22C2FC        1049          SHLD    SEADR
0692 CD010C        1050          CALL    WCRLF
0695 CD3910        1051          CALL    SUB2            ;ANZAHL DATENBYTES = STOP-START+1
0698 D8            1052          RC                      ;C: START < STOP ! (FEHLER)
0699 CD130C        1053          CALL    WCRLFI
069C 2042414E      1054          DB      ' BAND EINSCHALTEN, DANN SPACE',0
06A0 44204549
06A4 4E534348
06A8 414C5445
06AC 4E2C2044
06B0 414E4E20
06B4 53504143
06B8 45
06B9 00
06BA CDE00A        1055 SSPACE:  CALL    RCHAR           ;AUF SPACE WARTEN
06BD FE20          1056          CPI     ' '
06BF C2BA06        1057          JNZ     SSPACE
06C2 CD010C        1058          CALL    WCRLF
06C5 EB            1059          XCHG
06C6 13            1060          INX     D               ;(DE)=ANZAHL DATENBYTES
06C7 CD910C        1061          CALL    WCASBUFI
06CA 0A            1062          DB      LF,CR,0
06CB 0D
06CC 00
06CD 7B            1063          MOV     A,E
06CE E60F          1064          ANI     0FH             ;ANZAHL DATENBYTES = 16 ?
06D0 C2D506        1065          JNZ     S11             ; NEIN, KLEINER 16
06D3 3E10          1066 S1:      MVI     A,16            ; JA, =16 !
06D5 CD910C        1067 S11:     CALL    WCASBUFI        ;RECORD MARK AUSGEBEN
06D8 3A            1068          DB      ':',0           ;RECORD MARK ":"
06D9 00
06DA CD7C0C        1069          CALL    WCASAHEX        ;ANZAHL DER RECORDS
06DD 4F            1070          MOV     C,A
06DE CD9F0C        1071          CALL    WCASHLHEX       ;ANF.ADR. DER FOLGENDEN BYTES
06E1 85            1072          ADD     L               ;CHECKSUM BILDEN
06E2 84            1073          ADD     H
06E3 47            1074          MOV     B,A
```

5. MONITOR-KOMMANDOS (ALPHABETISCH)

```
LOC  OBJ        LINE        SOURCE STATEMENT
06E4 AF         1075        XRA     A
06E5 CD7C0C     1076        CALL    WCASAHEX        ;RECORD TYP AUSGEBEN
06E8 7E         1077 S2:    MOV     A,M             ;SPEICHERWERTE AUSGEBEN
06E9 CD7C0C     1078        CALL    WCASAHEX
06EC 80         1079        ADD     B
06ED 47         1080        MOV     B,A             ;CHECKSUM BILDEN
06EE 23         1081        INX     H               ;SPEICHERADR + 1
06EF 1B         1082        DCX     D               ;ANZAHL DATENBYTES - 1
06F0 0D         1083        DCR     C               ;RECORD AUSGEGEBEN ?
06F1 C2E806     1084        JNZ     S2              ; NEIN, WEITER
06F4 AF         1085        XRA     A               ; JA
06F5 90         1086        SUB     B
06F6 CD7C0C     1087        CALL    WCASAHEX        ;CHECKSUM AUSGEBEN
06F9 CD910C     1088        CALL    WCASBUFI
06FC 0D         1089        DB      CR,LF,0
06FD 0A
06FE 00
06FF 7A         1090        MOV     A,D
0700 B3         1091        ORA     E               ;WEITERE DATENBYTES ?
0701 C2D306     1092        JNZ     S1              ; JA, NAECHSTER RECORD
0704 CD910C     1093        CALL    WCASBUFI        ; NEIN, ENDE-RECORD AUSGEBEN
0707 3A         1094        DB      ':','00','0000','01','FF',0AH,0DH,0
0708 3030
070A 30303030
070E 3031
0710 4646
0712 0A
0713 0D
0714 00
0715 C9         1095        RET
                1096
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT
                    1097 ;*************************************************************************
                    1098 ;* TRACEINTERVAL -      INTERVALL-TRACE EIN-/AUSSCHALTEN
                    1099 ;*       "KMD ) "T"RACE INTERVAL"          T-CR ODER T-SPACE EINTIPPEN,
                    1100 ;*                                         "RACE INTERVAL" WIRD ERGAENZT
                    1101 ;*        "EIN/AUS    =X "Y                X=ALT, EIN: Y=E-CR ODER Y=E-SPACE
                    1102 ;*                                                AUS: Y=A-CR ODER Y=A-SPACE
                    1103 ;*                                                Y=CR ODER Y=SPACE: UNVERAENDERT
                    1104 ;*              (NUR, WENN TRACE EINGESCHALTET)
                    1105 ;*        "START-ADR =X1X1 "Y1Y1           X1X1=ALT, NEU: Y1Y1-CR ODER Y1Y1-SPACE
                    1106 ;*                                                ALT: CR ODER SPACE
                    1107 ;*        "STOP -ADR =X2X2 "Y2Y2           X2X2=ALT, NEU: Y2Y2-CR ODER Y2Y2-SPACE
                    1108 ;*                                                ALT: CR ODER SPACE
                    1109 ;=========================================================================
                    1110 ;*       ALGORITHMUS:
                    1111 ;*       NACH EINSCHALTEN DES TRACE-MODUS UND NACH AUSFUEHREN
                    1112 ;*       DES GO-KOMMANDOS WIRD EIN ANWENDERPROGRAMM
                    1113 ;*       UNTER KONTROLLE DES TRACEPROGRAMMES INSTRUKTION FUER
                    1114 ;*       INSTRUKTION DURCHLAUFEN. DAS ANWENDERPROGRAMM WIRD ALSO
                    1115 ;*       NICHT MEHR IN REALZEIT DURCHLAUFEN, D.H. UNTER ANDEREM,
                    1116 ;*       DASS DIE BEARBEITUNGSZEIT VERLAENGERT WIRD.
                    1117 ;*       (1) SOLANGE SICH DAS PROGRAMM IN BEREICHEN AUSSERHALB DES
                    1118 ;*           ANGEGEBENEN TRACE-INTERVALLS AUFHAELT, GESCHIEHT (AUSSER
                    1119 ;*           EINER VERLANGSAMUNG DES ABLAUFS) NICHTS.
                    1120 ;*       (2) GERAET DAS PROGRAMM IN DAS TRACE-INTERVALL (EINSCHLIESSLICH
                    1121 ;*           START-/STOP-ADRESSE), SO WERDEN NACH AUSFUEHRUNG JEDER
                    1122 ;*           INSTRUKTION IN DIESEM ADRESSBEREICH ALLE REGISTERINHALTE
                    1123 ;*           AUSGEDRUCKT. DER AUSDRUCK IST IDENTISCH MIT DEM DES
                    1124 ;*           REGISTER-KOMMANDOS.
                    1125 ;*       (3) NACH VERLASSEN DES TRACE-INTERVALLS GEHTS WEITER WIE
                    1126 ;*           UNTER (1) BESCHRIEBEN, BIS DAS PROGRAMM WIEDER EINE
                    1127 ;*           ADRESSE IM TRACE-INTERVALL ERREICHT.
                    1128 ;*       (4) BREAKPOINTS KOENNEN OHNE EINSCHRAENKUNG IM GESAMTEN
                    1129 ;*           ADRESSBEREICH WEITERHIN VERWENDET WERDEN.
                    1130 ;*       (5) IM TRACE-MODUS WERDEN DIE HALT-INSTRUKTIONEN (76HEX)
                    1131 ;*           SOWIE ILLEGALE OPCODES ERKANNT UND FUEHREN ZUM
                    1132 ;*           PROGRAMMABBRUCH. (OHNE TRACE-MODUS BLEIBT EIN PROGRAMM
                    1133 ;*           BEI AUSFUEHRUNG EINER HALT-INSTRUKTION OHNE WEITERE
                    1134 ;*           NACHRICHT EINFACH HAENGEN!)
                    1135 ;*************************************************************************
                    1136 TRACEINTERVAL:
0716 21C4FC         1137         LXI     H,TRCFLG
0719 CD7B09         1138         CALL    HEINAUS          ;TRACE EIN-/AUSSCHLALTEN
071C 05             1139         DCR     D                ;EINGESCHALTET ?
071D F8             1140         RM                       ;NEIN, FERTIG
071E 2ABAFC         1141         LHLD    TBADR            ;JA
0721 CD740A         1142         CALL    HSTARD           ;START-ADR HOLEN (UNTERE GRENZE)
0724 22BAFC         1143         SHLD    TBADR
0727 EB             1144         XCHG
0728 2ABCFC         1145         LHLD    TEADR
072B CD9C0A         1146         CALL    HSTOPD           ;STOP-ADR HOLEN (OBERE GRENZE)
072E 22BCFC         1147         SHLD    TEADR
0731 C9             1148         RET
                    1149
```

```
LOC  OBJ        LINE        SOURCE STATEMENT

                1150 ;*************************************************************************
                1151 ;* TRSTEP -      ANZAHL INSTRUCTIONEN TRACEN (STEPPEN)
                1152 ;*       MVI     D,ANZAHL TRACE-SCHRITTE
                1153 ;*       CALL    TRSTEP
                1154 ;*************************************************************************
                1155
0732 7A         1156 TRSTEP: MOV     A,D
0733 B7         1157         ORA     A              ;ANZAHL SCHRITTE = 0 ?
0734 C8         1158         RZ                     ; JA, FERTIG
0735 2AD7FC     1159 TRSTP1: LHLD    PCWERT         ; NEIN
0738 CDFE0E     1160         CALL    PREGF          ;ERSTEN REG.AUSDRUCK MIT TITEL
073B CDF610     1161 TRSTP2: CALL    TRACE          ;EINE INSTRUKTION TRACEN
073E DCCA07     1162         CC      TRHALT
0741 D8         1163         RC                     ;CY=1, halt oder illegale Instruktion
0742 CD120F     1164         CALL    PREGS          ;REGISTERINHALTE AUSDRUCKEN
0745 15         1165         DCR     D              ;WEITERE SCHRITTE ?
0746 C8         1166         RZ                     ; NEIN, FERTIG
0747 CDD20E     1167         CALL    CRTTST
074A DA3B07     1168         JC      TRSTP2         ; tty, keine Ueberschriften
074D CDE508     1169         CALL    LTST           ; JA, ZEILENZAHL KONTROLLIEREN
0750 CA3507     1170         JZ      TRSTP1         ;Z: MAX.ZEILENZAHL WAR ERREICHT, NEUER TITEL
0753 C33B07     1171         JMP     TRSTP2         ;   NOCH NICHT ERREICHT, KEIN NEUER TITEL
                1172
                1173 ;*************************************************************************
                1174 ;* TRINT -       INTERVALL TRACEN
                1175 ;*               REGISTERAUSDRUCKE, WENN PROGRAMM IM INTERVALL INCL. GRENZEN
                1176 ;*               KEINE AUSDRUCKE AUSSERHALB DES INTERVALLS
                1177 ;*************************************************************************
0756 CDCD08     1178 TRINT:  CALL    LINIT          ;ZEILENZAEHLER EINSCHALTEN
0759 0EFF       1179         MVI     C,0FFH         ;(C)=DRUCKKONTROLLE: OFFH=ABGESCHALTET
075B 3AC5FC     1180         LDA     BPTACT
075E B7         1181         ORA     A              ;BREAKPOINTS AKTIVIERT?
075F CA6A07     1182         JZ      TRINT1         ; NEIN
0762 79         1183         MOV     A,C            ; JA
0763 41         1184         MOV     B,C
0764 CD370E     1185         CALL    BPTSET         ;BREAKPOINT AUF NAECHSTEM OPCODE?
0767 DA7107     1186         JC      TRINT3         ; JA, BREAKPOINTS DANN NICHT EINFUEGEN
076A CDC50D     1187 TRINT1: CALL    BPTINS         ; NEIN
076D AF         1188         XRA     A
076E 32C5FC     1189         STA     BPTACT
0771 47         1190 TRINT3: MOV     B,A
0772 2AD7FC     1191         LHLD    PCWERT
0775 EB         1192         XCHG
0776 2ABAFC     1193         LHLD    TBADR
0779 7B         1194         MOV     A,E            ;UNTERHALB INTERVALL?
077A 95         1195         SUB     L
077B 7A         1196         MOV     A,D
077C 9C         1197         SBB     H
077D DA9D07     1198         JC      TRINT4         ;JA, KEIN AUSDRUCK
0780 2ABCFC     1199         LHLD    TEADR
0783 7D         1200         MOV     A,L            ;OBERHALB INTERVALL?
0784 93         1201         SUB     E
0785 7C         1202         MOV     A,H
0786 9A         1203         SBB     D
0787 DA9D07     1204         JC      TRINT4         ; JA, KEIN AUSDRUCK
```

```
LOC  OBJ          LINE        SOURCE STATEMENT

078A 0C           1205          INR     C               ; NEIN; VORHER BEREITS IM INTERVALL ?
078B C29D07       1206          JNZ     TRINT4          ;  JA
078E 04           1207          INR     B               ;  NEIN, NEUE TITELZEILE
078F C40C0E       1208          CNZ     BPTREM
0792 2AD7FC       1209          LHLD    PCWERT
0795 CDFE0E       1210          CALL    PREGF
0798 05           1211          DCR     B
0799 F4C50D       1212          CP      BPTINS
079C B7           1213          ORA     A
079D 3E00         1214 TRINT4:  MVI     A,0
079F DE00         1215          SBI     0
07A1 4F           1216          MOV     C,A
07A2 CDF610       1217          CALL    TRACE           ;EINE INSTRUKTION TRACEN
07A5 DCCA07       1218          CC      TRHALT
07A8 D8           1219          RC                      ;C: HALT ODER ILLEGALE INSTRUKTION
07A9 79           1220          MOV     A,C
07AA B7           1221          ORA     A               ;IM INTERVALL?
07AB 78           1222          MOV     A,B
07AC FA6A07       1223          JM      TRINT1          ; NEIN, KEIN AUSDRUCK
07AF 04           1224          INR     B               ; JA, REGISTERINHALTE AUSDRUCKEN
07B0 0600         1225          MVI     B,0
07B2 C40C0E       1226          CNZ     BPTREM          ;BREAKPOINTS MUESSEN VOR AUSDRUCK
07B5 CD120F       1227          CALL    PREGS           ;  DER REGISTERINHALTE ENTFERNT WERDEN,
07B8 CDC50D       1228          CALL    BPTINS          ;  DA SONST RST 4 ALS OPCODE DORT ERSCHEINT
07BB CDD20E       1229          CALL    CRTTST          ;CRT ?
07BE DA6A07       1230          JC      TRINT1          ; nein, tty
07C1 CDD508       1231          CALL    LTST            ;ZEILENKONTROLLE
07C4 CA5607       1232          JZ      TRINT           ; Z: MAX. ZAHL WAR ERREICHT, NEUER TITEL
07C7 C36A07       1233          JMP     TRINT1          ;   NICHT ERREICHT, KEIN NEUER TITEL
                  1234 ;*********************************************************************
                  1235 ;* TRHALT -     HALT ODER ILLEGALE INSTRUKTION
                  1236 ;*********************************************************************
07CA F5           1237 TRHALT:  PUSH    PSW
07CB 2AD7FC       1238          LHLD    PCWERT          ;REGISTERINHALTE MIT PC AN DER STELLE .
07CE 2B           1239          DCX     H               ;  DER HLT- BZW. DER ILLEGEALEN INSTRUKTION
07CF CD2502       1240          CALL    CMDABORT
07D2 48414C54     1241          DB      'HALT ODER ILLEGALER OPCODE',0
07D6 204F4445
07DA 5220494C
07DE 4C454741
07E2 4C455220
07E6 4F50434F
07EA 4445
07EC 00
07ED F1           1242          POP     PSW
07EE C9           1243          RET
                  1244 ;
                  1245
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                  1246 ;
                  1247 ;*****************************************************************************
                  1248 ;*       I/O-TREIBER (ALPHABETISCH)
                  1249 ;*
                  1250 ;*       CASI     - CASSETTEN-EINGABE UEBER UART (8251)
                  1251 ;*       CASINIT  - CASSETTEN-I/O INITIALISIEREN (8251)
                  1252 ;*       CASO     - CASSETTEN-AUSGABE UEBER UART (8251)
                  1253 ;*       SERI     - SERIELLE EINGABE (8085 SID)
                  1254 ;*       SERINIT  - SERIELLEN I/O INITIALISIEREN;
                  1255 ;*                  MIT BAUD-RATENBESTIMMUNG (8085 SID)
                  1256 ;*       SERO     - SERIELLE AUSGABE (8085 SOD)
                  1257 ;*****************************************************************************
                  1258
03E8              1259 B300     EQU      1000            ;GRENZE TTY((300 BAUD)/CRT()300 BAUD)
00FF              1260 UARTST   EQU      0FFH     ;ADRESSE STEUERREGISTER
00FE              1261 UARTD    EQU      0FEH     ;ADRESSE DATENREGISTER
                  1262
                  1263
                  1264 ;*****************************************************************************
                  1265 ;* CASI -        CASSETTEN-EINGABE, EIN 8BIT WERT
                  1266 ;*       CALL     CASI
                  1267 ;*                (A) = EINGELESENES ZEICHEN
                  1268 ;*****************************************************************
07EF DBFF         1269 CASI:    IN       UARTST          ;STATUS LESEN
07F1 E602         1270          ANI      2               ;RECEIVER READY ?
07F3 CAEF07       1271          JZ       CASI            ; NEIN
07F6 DBFE         1272          IN       UARTD           ; JA, DATEN LESEN
07F8 C9           1273          RET
                  1274 ;*****************************************************************
                  1275 ;* CASINIT -     CASSETTEN I/O INITTIALISIEREN
                  1276 ;*        CALL    CASINIT
                  1277 ;*****************************************************************
                  1278 CASINIT:
07F9 3EC3         1279          MVI      A,0C3H          ;RAM-SPRUNGADRESSEN VORBESETZEN
07FB 3286FC       1280          STA      CASIN
07FE 3289FC       1281          STA      CASOUT
0801 21EF07       1282          LXI      H,CASI          ;CASSETTEN INPUT
0804 2287FC       1283          SHLD     CASIN+1
0807 212108       1284          LXI      H,CASO          ;CASSETTEN OUTPUT
080A 228AFC       1285          SHLD     CASOUT+1
080D AF           1286          XRA A
080E D3FF         1287          OUT UARTST
0810 D3FF         1288          OUT UARTST
0812 D3FF         1289          OUT UARTST
0814 3E40         1290          MVI A,01000000B         ;RESET UART
0816 D3FF         1291          OUT UARTST
0818 3ECF         1292          MVI A,11001111B         ;8 BITS/OHNE PARITAET/
081A D3FF         1293          OUT UARTST              ;  2STOP-BITS/TEILER 64X
081C 3E25         1294          MVI A,00100101B         ;KOMMANDOWORT
081E D3FF         1295          OUT UARTST
0820 C9           1296          RET
                  1297
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                  1298 ;*****************************************************************************
                  1299 ;* CASO -       CASSETTEN-AUSGABE, EIN 8BIT WERT
                  1300 ;*****************************************************************************
                  1301
                  1302 ;       MVI     A,ZEICHEN
                  1303 ;       CALL    CASO
0821 F5           1304 CASO:   PUSH    PSW
0822 DBFF         1305 CASO1:  IN      UARTST          ;STATUS LESEN
0824 E601         1306         ANI     1               ;TRANSMITTER READY ?
0826 CA2208       1307         JZ      CASO1           ; NEIN, WARTEN
0829 F1           1308         POP     PSW             ; JA
082A D3FE         1309         OUT     UARTD           ;DATENBYTE AUSGEBEN
082C C9           1310         RET
                  1311
                  1312 ;*****************************************************************************
                  1313 ;*      SERIELLE EINGABE
                  1314 ;*      CALL    SERI
                  1315 ;*              (A) = EINGEGEBENES ZEICHEN
                  1316 ;*****************************************************************************
                  1317
082D E5           1318 SERI:   PUSH    H               ;REGISTER RETTEN
082E D5           1319         PUSH    D
082F C5           1320         PUSH    B
0830 11FAFF       1321         LXI     D,-6
0833 2ADFFC       1322         LHLD    STIME           ;BITZEIT*6 HOLEN
0836 B7           1323         ORA     A               ;  UND HALBIEREN, DA
0837 7C           1324         MOV     A,H             ;  BEI DER SERIELLEN
0838 1F           1325         RAR                     ;  EINGABE JEDES AN-
0839 67           1326         MOV     H,A             ;  KOMMENDES BIT IN DER
083A 7D           1327         MOV     A,L             ;  M I T T E  GELESEN
083B 1F           1328         RAR                     ;  WERDEN MUSS !!
083C 6F           1329         MOV     L,A
083D 0E0A         1330         MVI     C,10            ;10 BIT, 2. STOPBIT WIRD IGNORIERT
083F 0600         1331         MVI     B,0
0841 20           1332 SERI1:  RIM                     ;
0842 B7           1333         ORA     A               ;WARTEN AUF START BIT
0843 FA4108       1334         JM      SERI1           ;
                  1335                                 ;*AEUSSERE SCHLEIFE FUER ZEICHEN
0846 20           1336 SERI2:  RIM                     ;***INNERE SCHLEIFE: BIT EINLESEN
0847 B7           1337         ORA     A               ;***  1. BIT WIRD NACH HALBER BITZEIT*6 GELESEN
0848 19           1338         DAD     D               ;***
0849 DA4608       1339         JC      SERI2           ;***ENDE DER BITSCHLEIFE
084C 2600         1340         MVI     H,0             ;*    (ZEITAUSGLEICH)
084E 23           1341         INX     H               ;*    (ZEITAUSGLEICH)
084F 17           1342         RAL                     ;*  ERGEBNISBIT VON RECHTS
0850 78           1343         MOV     A,B             ;*  IN (B) HINEINROTIEREN
0851 1F           1344         RAR                     ;*
0852 47           1345         MOV     B,A             ;*
0853 2ADFFC       1346         LHLD    STIME           ;*  BITZEIT*6 HOLEN (BEIM 1.MAL WAR'S NUR
0856 0D           1347         DCR     C               ;*                  DIE HAELFTE !)
0857 C24608       1348         JNZ     SERI2           ;*NZ: NAECHSTES BIT EINLESEN
085A 78           1349         MOV     A,B             ;ERGEBNIS NACH (A) UND
085B 17           1350         RAL                     ;  RECHTS JUSTIEREN
085C C1           1351         POP     B               ;REGISTER WIEDERHERSTELLEN
085D D1           1352         POP     D
```

```
LOC  OBJ        LINE        SOURCE STATEMENT
085E E1         1353        POP     H
085F C9         1354        RET
                1355
                1356 ;*****************************************************************
                1357 ;* SERINIT -    SERIELLEN I/O INITIALISIEREN
                1358 ;*      CALL    SERINIT
                1359 ;*              ALLE REGISTER GEAENDERT
                1360 ;*****************************************************************
                1361
                1362 SERINIT:
0860 3E40       1363        MVI     A,40H
0862 30         1364        SIM                     ;AUSGANG AUF HIGH
0863 110100     1365        LXI     D,1
0866 210000     1366        LXI     H,0
0869 20         1367 SERIT1: RIM                    ;AUF START BIT WARTEN
086A B7         1368        ORA     A               ;STARTBIT ?
086B FA6908     1369        JM      SERIT1          ; NEIN
086E 19         1370 SERIT2: DAD    D               ;***ZENTRALE BITSSCHLEIFE:
086F 20         1371        RIM                     ;***  BITZEIT WIRD DURCH DIE EINGABE
0870 B7         1372        ORA     A               ;***  VON SPACE BESTIMMT, D.H. AUS DER ZEIT
0871 F26E08     1373        JP      SERIT2          ;***  VON 6 BIT
0874 11F4FF     1374        LXI     D,-2*6          ;ANPASSUNG DER BITZEIT AN DIE ZEIT,
0877 19         1375        DAD     D               ;  DIE BEI EIN-/AUSGABE IN DEN AUESSEREN
0878 22D0FC     1376        SHLD    STIME           ;  SCHLEIFEN HINZUKOMMT
087B 3EE8       1377        MVI     A,LOW B300      ;PRUEFEN, OB TTY (<= 300 BAUD)
087D 95         1378        SUB     L               ;   ODER  OB CRT ( ) 300 BAUD)
087E 3E03       1379        MVI     A,HIGH B300
0880 9C         1380        SBB     H
0881 3E00       1381        MVI     A,0             ;0 = CRT
0883 CE00       1382        ACI     0               ;1 = TTY
0885 32C8FC     1383        STA     CRTFLG
0888 11FFFF     1384 SERIT3: LXI    D,-1
088B 19         1385        DAD     D
088C DA8808     1386        JC      SERIT3
088F C0         1387        RNZ                     ; JA, FERTIG
0890 3E0C       1388        MVI     A,0CH           ; NEIN, SCHIRM LOESCHEN
0892 CD83FC     1389        CALL    SEROUT
0895 11204E     1390        LXI     D,20000         ;CA. 210 MS WARTEN
0898 1B         1391 SERIT4: DCX    D               ; (BEI 4 MHZ-TAKT)
0899 7A         1392        MOV     A,D
089A B3         1393        ORA     E
089B C29808     1394        JNZ     SERIT4
089E C9         1395        RET
                1396 ;*****************************************************************
                1397 ;* SERO -       SERIELLE AUSGABE
                1398 ;*      MVI     A,AUSGABE-ZEICHEN
                1399 ;*      CALL    SERO
                1400 ;*****************************************************************
089F E5         1401 SERO:  PUSH    H
08A0 D5         1402        PUSH    D
08A1 C5         1403        PUSH    B
08A2 F5         1404        PUSH    PSW
08A3 2F         1405        CMA                     ;AUSGANG INVERTIERT !
08A4 F680       1406        ORI     80H             ;STARTBIT ERZEUGEN
08A6 11FAFF     1407        LXI     D,-6
```

```
LOC  OBJ        LINE        SOURCE STATEMENT
08A9 0E0B       1408        MVI     C,11        ;11 BIT (1 START, 7 DATEN, 1 PAR., 2STOP)
08AB 2AIFFC     1409 SER01: LHLD    STIME       ;****  (--- AEUSSERE SCHLEIFE; BITZEIT*6 HOLEN
08AE 47         1410        MOV     B,A         ;   *
08AF E680       1411        ANI     80H         ;   *  HOECHSTES DATENBIT AUSMASKIEREN
08B1 F640       1412        ORI     40H         ;   *  SOD SCHARFMACHEN
08B3 30         1413 SER02: SIM                 ;   ***(--- INNERE SCHLEIFE
08B4 B7         1414        ORA     A           ;   * *
08B5 19         1415        DAD     D           ;   * *
08B6 DAB308     1416        JC      SER02       ;   ***
08B9 78         1417        MOV     A,B         ;   *
08BA E67F       1418        ANI     7FH         ;   *  STOPBIT ERZEUGEN (INVERTIERTE LOGIK !)
08BC 0F         1419        RRC                 ;   *
08BD 0D         1420        DCR     C           ;   *ALLE BITS AUSGEGEBEN ?
08BE C2A808     1421        JNZ     SER01       ;**** NEIN, WEITER
08C1 F1         1422        POP     PSW
08C2 C1         1423        POP     B
08C3 D1         1424        POP     D
08C4 E1         1425        POP     H
08C5 C9         1426        RET
                1427
                1428
                1429 ;
                1430
0010            1431 BUFLG  EQU     16          ;MAX. ANZAHL GEPUFFERTER ZEICHEN
000F            1432 NLINE  EQU     15          ;MAX. ANZAHL ZEICHEN AUF BILDSCHIRM
                1433
                1434
```

```
 LOC  OBJ        LINE        SOURCE STATEMENT

                  1435
                  1436 ;*************************************************************************
                  1437 ;*      ALLGEMEINE EIN-/AUSGABEROUTINEN
                  1438 ;*        (ALPHABETISCH GEORDNET AUSSER L...)
                  1439 ;*
                  1440 ;*      LCLR    - ZEILENZAEHLER AUSSCHALTEN
                  1441 ;*      LINIT   - ZEILENZAEHLER MIT MAX. ZEILENZAHL EINSCHALTEN
                  1442 ;*      LTST    - ZEILENZAEHLER PRUEFEN, OB MAX. ZEILENZAHL ERREICHT
                  1443 ;*
                  1444 ;*      HADR    - HOLE ADRESSE 16BIT
                  1445 ;*      HBPTD   - HOLE BREAKPOINTADRESSEN
                  1446 ;*      HDATA   - HOLE DATEN 8BIT (FORMAT BIN, DEZ, HEX)
                  1447 ;*      HDATAD  - HOLE DATEN 8BIT HEX (ALTWERT WIRD AUSGEDRUCKT)
                  1448 ;*      HEINAUS - HOLE "EIN/AUS"
                  1449 ;*      HFORMD  - HOLE DATENFORMAT (ALTWERT WIRD AUSGEDRUCKT)
                  1450 ;*      HJANEIN - HOLE "JA/NEIN"
                  1451 ;*      HPORTD  - HOLE I/O-PORT-ADRESSE (ALTWERT WIRD AUSGEDRUCKT)
                  1452 ;*      HSTART  - HOLE STARTADRESSE
                  1453 ;*      HSTARD  - "HSTART" + AUSDRUCK ALTWERT
                  1454 ;*      HSTOPD  - HOLE STOPADRESSE (ALTWERT WIRD AUSGEDRUCKT)
                  1455 ;*      P.....  - HILFSROUTINEN ZU H.....("HOLE")-ROUTINEN
                  1456 ;*
                  1457 ;*      RCHAR   - EIN ZEICHEN LESEN
                  1458 ;*      WAASC   - 8BIT WERT IN ASCII AUSGEBEN
                  1459 ;*      WABIN   - 8BIT WERT BINAER AUSGEBEN
                  1460 ;*      WADEZ   - 8BIT WERT DEZIMAL AUSGEBEN
                  1461 ;*      WAFOR   - 8BIT WERT MIT WAEHLBAREM FORMAT AUSGEBEN
                  1462 ;*                (ASCII, BINAER, DEZIMAL, HEXADEZIMAL)
                  1463 ;*      WAFORB  - WIE "WAFOR," DANACH EIN LEERZEICHEN
                  1464 ;*      WAHEX   - 8BIT WERT HEXADEZIMAL AUSGEBEN
                  1465 ;*      WAHEXB  - 8BIT WERT HEX. UND DANACH LEERZEICHEN AUSGEBEN
                  1466 ;*      WBELL   - KLINGELZEICHEN
                  1467 ;*      WBLANK  - LEERZEICHEN AUSGEBEN
                  1468 ;*      WBLNKI  - ANZAHL LEERZEICHEN NACH CALL AUSGEBEN
                  1469 ;*      WBUF    - PUFFER NACH (HL) AUSGEBEN
                  1470 ;*      WBUFI   - PUFFER NACH CALL AUSGEBEN
                  1471 ;*      WCHAR   - EIN ZEICHEN AUSGEBEN
                  1472 ;*      WCHARI  - EIN ZEICHEN NACH CALL AUSGGEBEN
                  1473 ;*      WCRLF   - ZEICHEN FUER WAGENRUECKLAUF (CR) UND ZEILENVORSCHUB (LF)
                  1474 ;*                AUSGEBEN
                  1475 ;*      WCRLFI  - WIE "WCRLF", DANACH TEXT NACH CALL AUSGEBEN
                  1476 ;*      WHLHEX  - 16BIT WERT HEXADEZIMAL AUSGEBEN
                  1477 ;*      WHLHXB  - WIE "WHLHEX", DANACH EIN LEERZEICHEN AUSGEBEN
                  1478 ;*=======================================================================*
                  1479 ;*      SPEZIELLE EIN-/AUSGABEROUTINEN
                  1480 ;*      (BANDCASSETTE, ALPHABETISCH GEORDNET)
                  1481 ;*
                  1482 ;*      RCAS     - ZEICHEN VON CASSETTE LESEN
                  1483 ;*      RCASAHEX - 8BIT WERT HEXADEZIMAL VON CASSETTE LESEN (MIT CHECKSUM)
                  1484 ;*      WCAS     - ZEICHEN AUF CASSETTE AUSGEBEN
                  1485 ;*      WCASAHEX - 8BIT WERT ALS ZWEI HEX.ZEICHEN AUF CASSETTE AUSGEBEN
                  1486 ;*      WCASBUFI - ZEICHEN NACH CALL AUF CASSETTE AUSGEBEN
                  1487 ;*      WCASHLHEX- 16BIT WERT IN (HL) HEXADEZIMAL AUSGEBEN
                  1488 ;*      WCASTRAIL- "TRAILER" AUSGEBEN (D.I. 60NULLEN)
                  1489 ;*=======================================================================
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT

                   1490 ;*      ALLGEMEINE HILFSROUTINEN
                   1491 ;*        (ALPHABETISCH GEORDNET)
                   1492 ;*
                   1493 ;*      *** EINGABEPUFFERUNG
                   1494 ;*      BCLR-   EINGABEPUFFER LOESCHEN
                   1495 ;*      BGETF-  ERSTES ZEICHEN AUS PUFFER HOLEN
                   1496 ;*      ABGETL- LETZTES ZEICHEN AUS PUFFER HOLEN
                   1497 ;*      BPUT-   EIN ZEICHEN IM PUFFER ABLEGEN
                   1498 ;*      BREAD-  ZEICHEN LESEN, PRUEFEN UND PUFFERN
                   1499 ;*
                   1500 ;*      *** BREAKPOINT-ROUTINEN
                   1501 ;*      BPTINS- BREAKPOINTS INS PROGRAMM EINFUEGEN
                   1502 ;*      BPTNUM- ANZAHL DER BREAKPOINTS ZURUECKGEBEN
                   1503 ;*      BPTREM- BREAKPOINTS AUS PROGRAMM HERAUSNEHMEN
                   1504 ;*      BPTSET- BREAKPOINTADRESSE PRUEFEN UND GGF. ABSPEICHERN
                   1505 ;*
                   1506 ;*      CALLHL- SUBROUTINENSPRUNG UEBER (HL)
                   1507 ;*      CALLTB- SUBROUTINENSPRUNG INDIZIERT UEBER ADRESSTABELLE
                   1508 ;*      CHTST-  TESTEN, OB ASCII-ZEICHEN ANGEGEBENEM FORMAT ENTSPRICHT
                   1509 ;*      CMP2 -  16BIT VERGLEICH
                   1510 ;*      CMPL -  EIN ZEICHEN MIT ZEICHENLISTE VERGLEICHEN
                   1511 ;*      CMPLI - WIE CMPL, ABER ZEICHENLISTE NACH CALL
                   1512 ;*      CRTTST- TEST, OB CRT ODER TTY
                   1513 ;*      DISLINE-DISASSEMBLIERE EINE INSTRUKTION BEI GEG. ADRESSE
                   1514 ;*      GROSS-  KLEINE BUCHSTABEN IN GROSSE WANDELN
                   1515 ;*      HEXASC- HEX.ZAHL ZU ASCII WANDELN
                   1516 ;*      PREGF-  TITELZEILE, (PC) UND OPCODE AUSGEBEN
                   1517 ;*      PREGP-  REGISTERAUSDRUCK POSITIONIEREN
                   1518 ;*      PREGS-  ALLE REGISTER AUSDRUCKEN
                   1519 ;*      PREGSO- WIE PREGS, ABER WAEHLBARE REGISTERFOLGE (FUER R-KDO)
                   1520 ;*      PREGT-  REGISTERUEBERSCHRIFT AUSDRUCKEN
                   1521 ;*      REGRES- ALLE REGISTER WIEDERHERSTELLEN ("RESTORE")
                   1522 ;*      REGSAV- ALLE REGISTER SICHERN ("SAVE")
                   1523 ;*      SUB2-   ZWEI 16BIT-WERTE SUBTRAHIEREN
                   1524 ;*      ZUASC-  "ASCII-ZU-ASCII" MIT TEST AUF KONTROLLZEICHEN
                   1525 ;*      ZUBIN-  ASCII-ZU-BINAER WANDELN; MIT TEST, OB 0 ODER 1
                   1526 ;*      ZUBIN8- "ZUBIN" + ERGEBNIS AUF (D)*2 ADDIEREN
                   1527 ;*      ZUDEZ-  ASCII-ZU-DEZ. WANDELN; MIT TEST, OB ZW. 0 UND 9
                   1528 ;*      ZUDEZ3- "ZUDEZ" + ERGEBNIS AUF (D)*10 ADDIEREN
                   1529 ;*      ZUFOR-  ASCII-ZU-WAEHLBARES-FORMAT WANDELN; ERGEBNIS AUF
                   1530 ;*              (D)*RADIX(2,10,16) ADDIEREN
                   1531 ;*      ZUHEX-  ASCII-ZU-HEX. WANDELN; MIT TEST, OB ZW. 0..9,A..F
                   1532 ;*      ZUHEX2- "ZUHEX" + ERGEBNIS AUF (D)*16 ADDIEREN
                   1533 ;*************************************************************************
                   1534 ;
                   1535
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                  1536 ;*************************************************************************
                  1537 ;* LCLR -        ZEILENZAEHLER ABSCHALTEN
                  1538 ;*       CALL    LCLR
                  1539 ;*************************************************************************
08C6 F5           1540 LCLR:    PUSH    PSW
08C7 AF           1541          XRA     A               ;ZEILENZAEHLER LOESCHEN UND
08C8 32CCFC       1542          STA     LINES           ;  ABSCHALTEN
08CB F1           1543          POP     PSW
08CC C9           1544          RET
                  1545
                  1546
                  1547 ;*************************************************************************
                  1548 ;* LINIT -       ZEILENZAEHLER INITIALISIEREN
                  1549 ;*       CALL    LINIT
                  1550 ;*************************************************************************
                  1551
08CD F5           1552 LINIT:   PUSH    PSW
08CE 3E0F         1553          MVI     A,NLINE         ;ZEILENZAEHLER MIT MAXIMALER
08D0 32CCFC       1554          STA     LINES           ;  ZEILENANZAHL EINSCHALTEN
08D3 F1           1555          POP     PSW
08D4 C9           1556          RET
                  1557
                  1558 ;*************************************************************************
                  1559 ;*
                  1560 ;* LTST -        STAND DES ZEILENZAEHLERS TESTEN (FUER REGISTER-TITEL)
                  1561 ;*       CALL    LTST
                  1562 ;*       JZ      MAXIMALE ZEILENZAHL
                  1563 ;*       JNZ     ZEILENZAHL KLEINER ALS MAX.
                  1564 ;*************************************************************************
08D5 C5           1565 LTST:    PUSH    B
08D6 47           1566          MOV     B,A
08D7 3ACCFC       1567          LDA     LINES           ;PRUEFE, OB MAXIMALE
08DA FE0F         1568          CPI     NLINE           ;  ZEILENANZAHL GESETZT
08DC 78           1569          MOV     A,B
08DD C1           1570          POP     B
08DE C9           1571          RET
                  1572
                  1573 ;*************************************************************************
                  1574 ;* HADR -        HOLE ADRESSE (MAX. 4 ZEICHEN HEX)
                  1575 ;*       LHLD    ALTWERT
                  1576 ;*       CALL    HADR
                  1577 ;*               (HL)= ALTWERT/NEUWERT (CY=1/0)
                  1578 ;*               (A) = ABSCHLUSSZEICHEN (CR, SPACE, ETC.)
                  1579 ;*               CY=1: ALTWERT
                  1580 ;*                  =0: NEUWERT
                  1581 ;*************************************************************************
08DF C5           1582 HADR:    PUSH    B
08E0 010304       1583          LXI     B,0403H         ;4 ZEICHEN, HEX
08E3 CD0A0D       1584          CALL    BREAD           ;EINLESEN
08E6 F5           1585          PUSH    PSW
08E7 CD0B0C       1586          CALL    BGETL           ;EIN ZEICHEN AUS EINGABEPUFFER HOLEN
08EA DA0509       1587          JC      HADR2           ;C: PUFFER LEER
08ED CD8410       1588          CALL    ZUHEX           ;HEX.WANDLUNG
08F0 6F           1589          MOV     L,A             ;RESULTAT IN (HL) SPEICHERN
08F1 2600         1590          MVI     H,0
```

8. EIN-/AUSGABEROUTINEN

```
LOC  OBJ          LINE        SOURCE STATEMENT
08F3 CDDB0C       1591 HADR1: CALL    BGETL           ;NAECHSTES ZEICHEN AUS PUFFER HOLEN
08F6 DA0509       1592        JC      HADR2           ;C: PUFFER LEER
08F9 CD8410       1593        CALL    ZUHEX
08FC 29           1594        DAD     H               ;WERT IN (HL) AUFBEREITEN
08FD 29           1595        DAD     H
08FE 29           1596        DAD     H
08FF 29           1597        DAD     H
0900 B5           1598        ORA     L
0901 6F           1599        MOV     L,A
0902 C3F308       1600        JMP     HADR1           ;NAECHSTES ZEICHEN
0905 F1           1601 HADR2: POP     PSW             ;(A) = ABSCHLUSSZEICHEN
0906 C1           1602        POP     B
0907 C9           1603        RET
                  1604
                  1605 ;*****************************************************************************
                  1606 ;* HBPTD -     HOLE BREAKPOINTS
                  1607 ;*       MVI   B,BREAKPOINTNUMMER (0...BPTANZ-1)
                  1608 ;*       CALL  HBPTD
                  1609 ;*****************************************************************************
0908 CD1409       1610 HBPTD: CALL    HBPTD1          ;ALTE BREAKPOINTADRESSEN AUSDRUCKEN
090B CDDF08       1611        CALL    HADR            ;NEUE BREAKPOINTADRESSE LESEN
090E CD370E       1612        CALL    BPTSET          ;  ABSPEICHERN
0911 FE20         1613        CPI     ' '
0913 C9           1614        RET
0914 C5           1615 HBPTD1: PUSH   B               ;TEXT 'BREAK-ADRX=' ALTWERT AUSGEBEN
0915 21A2FC       1616        LXI     H,BPTADR
0918 78           1617        MOV     A,B             ;BREAKPOINTNUMMER (0...BPTANZ-1)
0919 87           1618        ADD     A               ;  MAL 3
091A 80           1619        ADD     B
091B 4F           1620        MOV     C,A
091C 78           1621        MOV     A,B
091D 0600         1622        MVI     B,0
091F 09           1623        DAD     B               ;(HL) = SPEICHERADR. MIT BREAKPOINTS
0920 5E           1624        MOV     E,M
0921 23           1625        INX     H
0922 56           1626        MOV     D,M
0923 EB           1627        XCHG                    ;(HL) = BREAKPOINTADRESSE
0924 CD130C       1628        CALL    WCRLFI
0927 20425245     1629        DB      ' BREAK-ADR',0
092B 414B2D41
092F 4452
0931 00
0932 3C           1630        INR     A
0933 F630         1631        ORI     '0'             ;BREAKPOINTNUMMER = 1,2,...BPTANZ
0935 CDB60B       1632        CALL    WCHAR           ;  AUSDRUCKEN
0938 CDF70B       1633        CALL    WCHARI
093B 3D           1634        DB      '='
093C CD270C       1635        CALL    WHLHXB          ;BREAKPOINTADRESSE AUSDRUCKEN
093F C1           1636        POP     B
0940 C9           1637        RET
                  1638;
```

```
LOC  OBJ        LINE        SOURCE STATEMENT
                1639 ;*****************************************************************************
                1640 ;* HDATA -       HOLE 8BIT DATENBYTE
                1641 ;*       MVI     C,FORMAT (0=ASCII, 1=BIN, 2=DEZ, 3=HEX)
                1642 ;*       LXI     H,ADRESSE ALTWERT(/NEUWERT)
                1643 ;*       CALL    HDATA
                1644 ;*               (D)=NEUER/ALTER DATENWERT
                1645 ;*                  GLEICHER WERT BEI ADRESSE IN (HL)
                1646 ;*****************************************************************************
                1647
0941 C5         1648 HDATA:  PUSH    B
0942 0602       1649         MVI     B,2             ;2 ZEICHEN ENTSPR. FORMAT IN (C)
0944 CD0A0D     1650         CALL    BREAD           ;ZEICHEN EINLESEN
0947 C1         1651         POP     B
0948 F5         1652         PUSH    PSW
0949 E5         1653         PUSH    H
094A 66         1654         MOV     H,M             ;(H)=ALTWERT
094B CDDB0C     1655         CALL    BGETL           ;ERSTES ZEICHEN AUS PUFFER HOLEN
094E D25909     1656         JNC     HDATA1
0951 CD930B     1657         CALL    WBLNKI          ;PUFFER LEER,
0954 02         1658         DB      2               ;  ZWEIMAL SPACE AUSGEBEN
0955 54         1659         MOV     D,H             ;(D)=ALTWERT
0956 C36A09     1660         JMP     HDATA2          ; FERTIG
0959 1600       1661 HDATA1: MVI     D,0
095B CD7110     1662         CALL    ZUFOR           ;ZEICHEN ENTSPR. FORMAT IN (C) WANDELN
095E CDDB0C     1663         CALL    BGETL           ;NAECHSTES ZEICHEN AUS PUFFER HOLEN
0961 DCBE0B     1664         CC      WBLANK          ;PUFFER LEER, SPACE AUSGEBEN
0964 DA6A09     1665         JC      HDATA2
0967 CD7110     1666         CALL    ZUFOR           ;WEITERES ZEICHEN WANDELN
096A E1         1667 HDATA2: POP     H
096B 72         1668         MOV     M,D             ;NEUWERT/ALTWERT ABSPEICHERN
096C F1         1669         POP     PSW
096D C9         1670         RET
                1671
                1672 ;*****************************************************************************
                1673 ;* HDATAD -      HOLE DATENBYTE; DEFAULT WIRD ZUVOR AUSGEDRUCKT
                1674 ;*       CALL    HDATAD
                1675 ;*               (D) = 8BIT NEUWERT/ALTWERT
                1676 ;*                  GLEICHER WERT BEI "DATA" ABGESPEICHERT
                1677 ;*****************************************************************************
                1678
096E 21CDFC     1679 HDATAD: LXI     H,DATA
0971 7E         1680         MOV     A,M             ;ALTWERT HOLEN UND
0972 CDA30A     1681         CALL    PDATA           ;  ZUSAMMEN MIT ' DATEN =' AUSDRUCKEN
0975 0E03       1682         MVI     C,3             ;HEX. DATEN
0977 CD4109     1683         CALL    HDATA           ;DATENBYTE HOLEN
097A C9         1684         RET
                1685 ;*****************************************************************************
                1686 ;* HEINAUS -     HOLE "EIN/AUS"
                1687 ;*       LXI     H,ADRESSE UNTER DER EIN/AUS-ZUSTAND STEHT
                1688 ;*       CALL    HEINAUS
                1689 ;*       (B)=ALTER/NEUER EIN(=1)/AUS(=0)-ZUSTAND
                1690 ;*                  GLEICHER WERT UNTER ADRESSE IN (HL)
                1691 ;*****************************************************************************
                1692
                1693 HEINAUS:
```

8. EIN-/AUSGABEROUTINEN

```
 LOC  OBJ          LINE        SOURCE STATEMENT
097B CD010C       1694          CALL    WCRLF
097E CD800B       1695 HEIN0:   CALL    WBUFI
0981 0D           1696          DB      0DH,' EIN/AUS  =',0
0982 2045494E
0986 2F415553
098A 20203D
098D 00
098E 7E           1697          MOV     A,M             ;(A)=EIN/AUS-ZUSTAND
098F B7           1698          ORA     A               ;AUS=1 ?
0990 3E41         1699          MVI     A,'A'
0992 CA9709       1700          JZ      HEIN1           ;JA,  AUS = 'A'
0995 3E45         1701          MVI     A,'E'           ;NEIN,EIN = 'E'
0997 CD860B       1702 HEIN1:   CALL    WCHAR           ;ZUSTAND AUSDRUCKEN
099A CD8E0B       1703          CALL    WBLANK
099D 010001       1704          LXI     B,0100H         ;1 ZEICHEN, ASCII
09A0 CD0A0D       1705          CALL    BREAD           ;ZEICHEN EINLESEN
09A3 CD0B0C       1706          CALL    BGETL           ;ZEICHEN AUS PUFFER HOLEN
09A6 46           1707          MOV     B,M             ;  (B)=ALTER ZUSTAND
09A7 3F           1708          CMC
09A8 D0           1709          RNC                     ;NC: ALTER ZUSTAND BLEIBT (PUFFER LEER)
09A9 0600         1710          MVI     B,0             ;GUELTIGKEIT DES EINGELESENEN
09AB FE41         1711          CPI     'A'             ;  ZEICHENS PRUEFEN;
09AD CAC309       1712          JZ      HEIN2           ;  GUELTIG IST 'E' FUER EIN=1
09B0 04           1713          INR     B               ;  UND         'A' FUER AUS=0
09B1 FE45         1714          CPI     'E'
09B3 CAC309       1715          JZ      HEIN2
09B6 CD890B       1716          CALL    WBELL           ;UNGUELTIGES ZEICHEN, NOCHMAL LESEN
09B9 CDD20E       1717          CALL    CRTTST
09BC DA7B09       1718          JC      HEINAUS
09BF C37E09       1719          JMP     HEIN0
09C2 C9           1720          RET
09C3 70           1721 HEIN2:   MOV     M,B             ;ZUSTAND ABSPEICHERN
09C4 C9           1722          RET
                  1723
                  1724 ;*****************************************************************
                  1725 ;* HFORMD -      HOLE ZAHLENFORMAT (MIT AUSDRUCK 'FORMAT=ALTWERT ')
                  1726 ;*       CALL    HFORMD
                  1727 ;*               FORMAT = NEUES/ALTES FORMAT
                  1728 ;*****************************************************************
                  1729
09C5 CD010C       1730 HFORMD:  CALL    WCRLF
09C8 CD800B       1731 HFORD0:  CALL    WBUFI
09CB 0D           1732          DB      0DH,' FORMAT    =',0
09CC 20464F52
09D0 4D415420
09D4 2020203D
09D8 00
09D9 3AA1FC       1733          LDA     FORMAT
09DC CD860B       1734          CALL    WCHAR
09DF CD8E0B       1735          CALL    WBLANK
09E2 010001       1736          LXI     B,0100H         ;1 ZEICHEN, ASCII
09E5 CD0A0D       1737          CALL    BREAD           ;IN EINGABEPUFFER LESEN
09E8 CD0B0C       1738          CALL    BGETL           ;ZEICHEN AUS PUFFER HOLEN
09EB D2F109       1739          JNC     HFORD1
09EE 3AA1FC       1740          LDA     FORMAT
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
09F1 CDCC0E       1741 HFORD1: CALL    CMPLI          ;LEGALES ZEICHEN ?
09F4 41           1742         DB      'A'
09F5 42           1743         DB      'B'
09F6 44           1744         DB      'D'
09F7 48           1745         DB      'H'
09F8 00           1746         DB      0
09F9 D20B0A       1747         JNC     HFORD2         ;JA, FERTIG
09FC CD890B       1748         CALL    WBELL          ;NEIN, KLINGELN
09FF CDD20E       1749         CALL    CRTTST         ;  UND NEUE EINGABE
0A02 DAC509       1750         JC      HFORMD         ;  ANFORDERN
0A05 C3C809       1751         JMP     HFORD0
0A08 32A1FC       1752 HFORD2: STA     FORMAT
0A0B C9           1753         RET
                  1754
                  1755 ;*********************************************************************
                  1756 ;* HJANEIN -    HOLE "JA/NEIN"
                  1757 ;*      CALL    HJANEIN
                  1758 ;*      JZ      JA
                  1759 ;*      JNZ     NICHT JA
                  1760 ;*********************************************************************
                  1761
                  1762 HJANEIN:
0A0C E3           1763         XTHL
0A0D CDA10B       1764         CALL    WBUF           ;ZEICHEN NACH CALL AUSGEBEN
0A10 E3           1765         XTHL                   ;RETURN-ADR. AUF STACK
0A11 CDB00B       1766         CALL    WBUFI
0A14 20284A41     1767         DB      ' (JA/NEIN) ',0
0A18 2F4E4549
0A1C 4E2920
0A1F 00
0A20 010001       1768         LXI     B,0100H        ;1 ZEICHEN; ASCII
0A23 CD0A0D       1769         CALL    BREAD
0A26 CDDB0C       1770         CALL    BGETL
0A29 FE4A         1771         CPI     'J'            ;Z=JA / NZ=NICHT JA
0A2B C9           1772         RET
                  1773
                  1774 ;*********************************************************************
                  1775 ;* HPORTD -     HOLE EIN-/AUSGABE-PORT; DEFAULT WIRD VORHER AUSGEDRUCKT
                  1776 ;*              EINGEGEBEN WIRD ENTWEDER: PORT-ADR ODER
                  1777 ;*                                        '+' FUER ADR+1 ODER
                  1778 ;*                                        '-' FUER ADR-1
                  1779 ;*      CALL    HPORTD
                  1780 ;*              PORT = ALTE/NEUE PORT-ADRESSE
                  1781 ;*********************************************************************
                  1782
0A2C CD130C       1783 HPORTD: CALL    WCRLFI
0A2F 20504F52     1784         DB      ' PORT-NR=',0
0A33 543D4E52
0A37 3D
0A38 00
0A39 21CEFC       1785         LXI     H,PORT
0A3C 7E           1786         MOV     A,M
0A3D CD020B       1787         CALL    WAHEXB         ;ALTWERT AUSDRUCKEN
0A40 010302       1788         LXI     B,0203H        ;2 ZEICHEN, HEX
0A43 CD0A0D       1789         CALL    BREAD          ;ZEICHEN EINLESEN
```

```
LOC  OBJ        LINE        SOURCE STATEMENT
0A46 F5         1790        PUSH    PSW             ;ABSCHLUSSZEICHEN FUER SPAETER RETTEN
0A47 56         1791        MOV     D,M
0A48 CDDB0C     1792        CALL    BGETL           ;ERSTES ZEICHEN AUS PUFFER HOLEN
0A4B DA5B0A     1793        JC      HPORD1          ;PUFFER LEER, ABSCHLUSSZEICHEN PRUEFEN
0A4E CD8410     1794        CALL    ZUHEX           ;ZEICHEN ZU HEX. WANDELN
0A51 57         1795        MOV     D,A
0A52 CDDB0C     1796        CALL    BGETL           ;ZWEITES ZEICHEN AUS PUFFER HOLEN
0A55 DA5B0A     1797        JC      HPORD1          ;PUFFER LEER
0A58 CD9610     1798        CALL    ZUHEX2          ;WANDELN
0A5B 42         1799 HPORD1: MOV    B,D
0A5C 70         1800        MOV     M,B
0A5D F1         1801        POP     PSW             ;ABSCHLUSSZEICHEN AUSWERTEN:
0A5E FE21       1802        CPI     ' '+1           ;  CR ODER SPACE ?
0A60 3F         1803        CMC
0A61 D0         1804        RNC                     ;JA,  FERTIG
0A62 34         1805        INR     M               ;NEIN
0A63 FE2D       1806        CPI     '-'             ;ADR-1 ?
0A65 C22C0A     1807        JNZ     HPORTD          ;NEIN, ADR+1, WEITER
0A68 35         1808        DCR     M               ;JA,   ADR-1
0A69 35         1809        DCR     M
0A6A C32C0A     1810        JMP     HPORTD          ;              WEITER
                1811
                1812 ;*************************************************************************
                1813 ;* HSTART -    HOLE STARTADRESSE (4 HEX.ZEICHEN)
                1814 ;*       LHLD  ALTER WERT
                1815 ;*       CALL  HSTART
                1816 ;*             (HL)=NEUWERT/ALWERT
                1817 ;*************************************************************************
                1818
0A6D CD840A     1819 HSTART: CALL   PSTART          ;'START-ADR=' AUSDRUCKEN
0A70 CDDF08     1820        CALL    HADR            ;STARTADRESSE HOLEN
0A73 C9         1821        RET
                1822
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT
                    1823 ;*********************************************************************
                    1824 ;* HSTARD -      HOLE STARTADRESSE; DEFAULT ZUVOR AUSDRUCKEN
                    1825 ;*       LHLD    ALTER WERT
                    1826 ;*       CALL    HSTARD
                    1827 ;*               (HL)=NEUWERT/ALTWERT
                    1828 ;*********************************************************************
                    1829
0A74 CDC50A         1830 HSTARD: CALL     PSTARD          ;'START-ADR=' MIT ALTWERT AUSDRUCKEN
0A77 CDDF08         1831         CALL     HADR            ;STARTADRESSE HOLEN
0A7A C9             1832         RET
                    1833
                    1834 ;*********************************************************************
                    1835 ;* HSTEPD -      HOLE ANZAHL EINZELINSTRUKTIONEN; DEFAULT ZUVOR AUSDRUCKEN
                    1836 ;*       CALL    HSTEPD
                    1837 ;*               (C)=SCHRITTZAHL
                    1838 ;*                   DESGLEICHEN IN "STEPS"
                    1839 ;*********************************************************************
                    1840
0A7B CD130C         1841 HSTEPD: CALL     WCRLFI
0A7E 20535445       1842         DB       ' STEPS     =',0
0A82 50533020
0A86 2020203D
0A8A 00
0A8B 21CBFC         1843         LXI      H,STEPS
0A8E 7E             1844         MOV      A,M
0A8F CD190B         1845         CALL     WADEZ           ;ALTWERT DEZ. AUSDRUCKEN
0A92 CD8E0B         1846         CALL     WBLANK
0A95 0E02           1847         MVI      C,2             ;DATEN DEZ. (0...99)
0A97 CD4109         1848         CALL     HDATA           ;DEZ. SCHRITTZAHL HOLEN UND
0A9A 48             1849         MOV      C,B             ;  (C) ZURUECKGEBEN
0A9B C9             1850         RET
                    1851 ;*********************************************************************
                    1852 ;* HSTOPD -      HOLE STOPADRESSE; ZUVOR DEFAULT AUSGEBEN
                    1853 ;*       LHLD    ALTWERT
                    1854 ;*       CALL    HSTOPD
                    1855 ;*********************************************************************
0A9C CDCC0A         1856 HSTOPD: CALL     PSTOPD          ;'STOP-ADR=' MIT ALTWERT AUSGEBEN
0A9F CDDF08         1857         CALL     HADR            ;STOPADRESSE HOLEN
0AA2 C9             1858         RET
                    1859
                    1860 ;*********************************************************************
                    1861 ;* PDATA -       'DATEN=' UND ALTWERT AUSDRUCKEN
                    1862 ;*       MVI     A,ALTWERT
                    1863 ;*       CALL    PDATA
                    1864 ;*********************************************************************
0AA3 CD130C         1865 PDATA:  CALL     WCRLFI
0AA6 20444154       1866         DB       ' DATEN  =',0
0AAA 454E2020
0AAE 3D
0AAF 00
0AB0 CD820B         1867         CALL     WAHEXB          ;ALTWERT AUSDRUCKEN
0AB3 C9             1868         RET
                    1869
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT
                   1870 ;*************************************************************************
                   1871 ;* PSTART -      'START-ADR=' IN NEUER ZEILE AUSDRUCKEN
                   1872 ;*       LHLD    ALTWERT
                   1873 ;*       CALL    PSTART
                   1874 ;*************************************************************************
 0AB4 CD130C       1875 PSTART: CALL     WCRLFI
 0AB7 20535441     1876         DB       ' START-ADR =',0
 0ABB 52542D41
 0ABF 4452203D
 0AC3 00
 0AC4 C9           1877         RET
                   1878 ;
                   1879 ;*************************************************************************
                   1880 ;* PSTARD -      'START-ADR=' UND ALTWERT AUSDRUCKEN
                   1881 ;*       LHLD    ALTWERT
                   1882 ;*       CALL    PSTARD
                   1883 ;*************************************************************************
 0AC5 CDB40A       1884 PSTARD: CALL     PSTART
 0AC8 CD270C       1885         CALL     WHLHXB          ;ALTWERT IN (HL) AUSDRUCKEN
 0ACB C9           1886         RET
                   1887
                   1888 ;*************************************************************************
                   1889 ;* PSTOPD -      'STOP -ADR=' UND ALTE STOPADRESSE AUSDRUCKEN
                   1890 ;*       LHLD    ALTE STOPADRESSE
                   1891 ;*       CALL    PSTOPD
                   1892 ;*************************************************************************
 0ACC CD130C       1893 PSTOPD: CALL     WCRLFI
 0ACF 2053544F     1894         DB       ' STOP -ADR =',0
 0AD3 50202D41
 0AD7 4452203D
 0ADB 00
 0ADC CD270C       1895         CALL     WHLHXB          ;ALTE STOPADRESSE AUSDRUCKEN
 0ADF C9           1896         RET
                   1897
                   1898 ;*************************************************************************
                   1899 ;* RCHAR         -       EIN ZEICHEN EINLESEN
                   1900 ;*       CALL    RCHAR
                   1901 ;*               (A) = ZEICHEN
                   1902 ;*************************************************************************
                   1903 RCHAR:
                   1904
 0AE0 CD80FC       1905         CALL     SERIN           ;MDS-INPUT
 0AE3 E67F         1906         ANI      7FH             ;BIT7 WEGMASKIEREN
 0AE5 FE1B         1907         CPI      ESC             ;EINGABE ABBRECHEN ?
 0AE7 C0           1908         RNZ                      ; NEIN, EXIT
 0AE8 CD890B       1909         CALL     WBELL           ; JA, KLINGELN UND
 0AEB CDAC01       1910         CALL     CMD             ;     ZURUECK ZU KOMMANDOROUTINE
                   1911                                  ;KOMMT NIEMALS VON "CMD" ZURUECK !!
                   1912
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                  1913 ;*************************************************************************
                  1914 ;* WAASC         -       8BIT WERT IN ASCII AUSGEBEN
                  1915 ;*       MVI     A,8BIT WERT
                  1916 ;*       CALL    WAASC
                  1917 ;*               Z.B. .X, .Z, .#, .+  DRUCKBARE ZEICHEN
                  1918 ;*                    10, B4, 1B, 7F  NICHT-DRUCHBARE ZEICHEN
                  1919 ;*************************************************************************
                  1920
0AEE FE20         1921 WAASC: CPI     20H             ;KONTROLLZEICHEN ?
0AF0 DA000B       1922        JC      WAASC1          ; JA
0AF3 FE7F         1923        CPI     RUBOUT          ; NEIN; RUBOUT ODER 80-FF HEX ?
0AF5 D2000B       1924        JNC     WAASC1          ;   JA
0AF8 CDF70B       1925        CALL    WCHARI          ;   NEIN; ZEICHEN MIT VORANGEHENDEM
0AFB 2E           1926        DB      '.'             ;         PUNKT AUSGABEN
0AFC CDB60B       1927        CALL    WCHAR
0AFF C9           1928        RET
0B00 CD6F0B       1929 WAASC1: CALL    WAHEX           ;NICHT DRUCKBARES ZEICHEN HEX AUSGEBEN
0B03 C9           1930        RET
                  1931 ;
                  1932 ;*************************************************************************
                  1933 ;* WABIN -       8BIT WERT BINAER AUSGEBEN
                  1934 ;*       MVI     A,8BIT WERT
                  1935 ;*       CALL    WABIN
                  1936 ;*************************************************************************
0B04 E5           1937 WABIN: PUSH    H               ;REGISTERINHALTE RETTEN
0B05 C5           1938        PUSH    B
0B06 F5           1939        PUSH    PSW
0B07 67           1940        MOV     H,A             ;BINAERSTELLEN AUS H HERAUSSCHIEBEN
0B08 0E08         1941        MVI     C,8             ;8 BINAERSTELLEN
0B0A AF           1942 WABIN1: XRA     A
0B0B 29           1943        DAD     H               ;BIT AUS H IN CARRY SCHIEBEN
0B0C CE30         1944        ACI     '0'             ; 0 -> 30='0',  1 -> 31='1'
0B0E CDB60B       1945        CALL    WCHAR
0B11 0D           1946        DCR     C               ;WEITERE BINAERSTELLEN ?
0B12 C20A0B       1947        JNZ     WABIN1          ; JA
0B15 F1           1948        POP     PSW             ; NEIN; REGISTER WIEDERHERSTELLEN UND ZURUECK
0B16 C1           1949        POP     B
0B17 E1           1950        POP     H
0B18 C9           1951        RET
                  1952
```

```
LOC  OBJ        LINE        SOURCE STATEMENT
                1953 ;*****************************************************************************
                1954 ;* WADEZ -      8BIT WERT DEZIMAL AUSGEBEN
                1955 ;*              (MIT UNTERDRUECKUNG DER FUEHRENDEN NULLEN)
                1956 ;*      MVI     A,8BIT WERT
                1957 ;*      CALL    WADEZ
                1958 ;*****************************************************************************
                1959
0B19 E5         1960 WADEZ:  PUSH    H              ;REGISTERINHALTE SICHERN
0B1A D5         1961         PUSH    D
0B1B F5         1962         PUSH    PSW
0B1C 0600       1963         MVI     B,0            ;NULLENUNTERDR. AN
0B1E 2E64       1964         MVI     L,100          ;100ER STELLE WANDELN
0B20 CD310B     1965         CALL    WADEZ1         ;      UND AUSGEBEN
0B23 2E0A       1966         MVI     L,10           ;10ER STELLE WANDELN
0B25 CD310B     1967         CALL    WADEZ1         ;    UND AUSGEBEN
0B28 F630       1968         ORI     '0'            ;1ER STELLE AUSGEBEN
0B2A CDB60B     1969         CALL    WCHAR
0B2D C1         1970         POP     B
0B2E D1         1971         POP     D
0B2F E1         1972         POP     H
0B30 C9         1973         RET
0B31 0E2F       1974 WADEZ1: MVI     C,'0'-1        ;STELLE MIT WERTIGKEIT IN C AUSGEBEN
0B33 0C         1975 WADEZ2: INR     C
0B34 95         1976         SUB     L              ;SOLANGE MIT STELLENWERTIGKEIT (100,10,1)
0B35 D2330B     1977         JNC     WADEZ2         ;  SUBTRAHIEREN, BIS UNTERLAUF
0B38 85         1978         ADD     L              ;  PASSIERT; C ENTHAELT STELLE
0B39 F5         1979         PUSH    PSW
0B3A 78         1980         MOV     A,B
0B3B B7         1981         ORA     A              ;NULLENUNTERDR. AN?
0B3C C2480B     1982         JNZ     WADEZ3         ; NEIN
0B3F 79         1983         MOV     A,C            ; JA
0B40 FE30       1984         CPI     '0'            ;STELLE = 0 ?
0B42 C2480B     1985         JNZ     WADEZ3         ; NEIN
0B45 0E20       1986         MVI     C,' '          ; JA, LEERZEICHEN STATT NULL
0B47 05         1987         DCR     B              ;NULLENUNTERDR. BLEIBT AN
0B48 04         1988 WADEZ3: INR     B
0B49 79         1989         MOV     A,C
0B4A CDB60B     1990         CALL    WCHAR          ;ZIFFER AUSGEBEN
0B4D F1         1991         POP     PSW
0B4E C9         1992         RET
                1993
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT
                   1994 ;*************************************************************************
                   1995 ;* WAFOR -       8BIT WERT ENTSPRECHEND FORMAT AUSGEBEN
                   1996 ;*       MVI     A,WERT
                   1997 ;*       MVI     C,FORMATINDEX (0=ASCII, 1=BIN, 2=DEZ, 3=HEX)
                   1998 ;*       CALL    WAFOR
                   1999 ;*************************************************************************
 0B4F E5           2000 WAFOR:   PUSH    H
 0B50 C5           2001          PUSH    B
 0B51 47           2002          MOV     B,A             ;(A) IN (B) ZWISCHENSPEICHERN
 0B52 79           2003          MOV     A,C             ;BEREICH VON 0...3 AUSBLENDEN
 0B53 E603         2004          ANI     3
 0B55 4F           2005          MOV     C,A
 0B56 78           2006          MOV     A,B             ;(A) WIEDERHERSTELLEN
 0B57 21600B       2007          LXI     H,WAFTAB
 0B5A CD750E       2008          CALL    CALLTB          ;MIT (C) ALS INDEX SPRINGEN
 0B5D C1           2009          POP     B
 0B5E E1           2010          POP     H
 0B5F C9           2011          RET
 0B60 EE0A         2012 WAFTAB:  DW      WAASC
 0B62 040B         2013          DW      WABIN
 0B64 190B         2014          DW      WADEZ
 0B66 6F0B         2015          DW      WAHEX
                   2016
                   2017 ;*************************************************************************
                   2018 ;* WAFORB        - WIE "WAFOR", DANACH EIN LEERZEICHEN
                   2019 ;*************************************************************************
                   2020
 0B68 CD4F0B       2021 WAFORB:  CALL    WAFOR           ;FORMATIERT AUSGEBEN
 0B6B CD8E0B       2022          CALL    WBLANK          ;1 SPACE
 0B6E C9           2023          RET
                   2024 ;*************************************************************************
                   2025 ;* WAHEX -       8BIT WERT ALS 2 HEX.ZEICHEN AAUSGEBEN
                   2026 ;*************************************************************************
                   2027
 0B6F F5           2028 WAHEX:   PUSH    PSW
 0B70 0F           2029          RRC
 0B71 0F           2030          RRC
 0B72 0F           2031          RRC
 0B73 0F           2032          RRC             ;LINKEN NIBBLE VON
 0B74 CDF20E       2033          CALL    HEXASC  ;  HEX. IN ASCII WANDELN
 0B77 CD860B       2034          CALL    WCHAR   ;  UND AUSGEBEN
 0B7A F1           2035          POP     PSW     ;RECHTEN NIBBLE VON
 0B7B CDF20E       2036          CALL    HEXASC  ;  HEX. IN ASCII WANDELN
 0B7E CD860B       2037          CALL    WCHAR   ;  UND AUSGEBEN
 0B81 C9           2038          RET
                   2039
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                  2040 ;*************************************************************************
                  2041 ;* WAHEXB -      8BIT WERT HEX AUSGEBEN, DANACH EIN LEERZEICHEN
                  2042 ;*************************************************************************
                  2043
0B82 CD6F0B       2044 WAHEXB: CALL     WAHEX             ;HEX. AUSGEBEN
0B85 CD8E0B       2045         CALL     WBLANK            ;1 SPACE HINTERHER
0B88 C9           2046         RET
                  2047
                  2048 ;*************************************************************************
                  2049 ;* WBELL -       BIMM !
                  2050 ;*       CALL    WBELL
                  2051 ;*************************************************************************
                  2052
0B89 CDF70B       2053 WBELL:  CALL     WCHARI            ;KLINGELZEICHEN BZW. PIEP AUSGEBEN
                  2054
0B8C 07           2055 DB       BEL
0B8D C9           2056         RET
                  2057 ;*************************************************************************
                  2058 ;* WBLANK -      EIN LEERZEICHEN AUSGEBEN
                  2059 ;*       CALL    WBLANK
                  2060 ;*************************************************************************
                  2061
0B8E CDF70B       2062 WBLANK: CALL     WCHARI
0B91 20           2063         DB       ' '
0B92 C9           2064         RET
                  2065
                  2066 ;*************************************************************************
                  2067 ;* WBLNKI -      ANZAHL LEERZEICHEN NACH CALL AUSGEBEN
                  2068 ;*       CALL    WBLNKI
                  2069 ;*       DB      12        ;12 LEERZEICHEN AUSGEBEN
                  2070 ;*************************************************************************
0B93 E3           2071 WBLNKI: XTHL                       ;(HL) DEUTET AUF ZAHL NACH CALL
0B94 F5           2072         PUSH     PSW
0B95 7E           2073         MOV      A,M               ;(A)=ANZAHL SPACE
0B96 CD8E0B       2074 WBLNK1: CALL     WBLANK
0B99 3D           2075         DCR      A                 ;WEITERE SPACE ?
0B9A C2960B       2076         JNZ      WBLNK1            ; JA
0B9D F1           2077         POP      PSW               ; NEIN, FERTIG
0B9E 23           2078         INX      H
0B9F E3           2079         XTHL
0BA0 C9           2080         RET
                  2081
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                  2082 ;*************************************************************************
                  2083 ;* WBUF -       PUFFER NACH (HL) AUSGEBEN
                  2084 ;*       LXI     H,ANFANGSADRESSE DES TEXTPUFFERS
                  2085 ;*       CALL    WBUF
                  2086 ;*               TEXTPUFFER = 'TEXT',0    (0 ALS ENDEZEICHEN)
                  2087 ;*               (HL) ZEIGT AUF WERT NACH DEM ENDEZEICHEN
                  2088 ;*************************************************************************
                  2089
OBA1 F5           2090 WBUF:   PUSH    PSW
OBA2 7E           2091 WBUF1:  MOV     A,M             ;ZEICHEN AUS SPEICHER HOLEN UND
OBA3 23           2092         INX     H               ;  ZEIGER UM 1 ERHOEHEN
OBA4 B7           2093         ORA     A               ;=ENDEZEICHEN ?
OBA5 CAAE0B       2094         JZ      WBUF2           ; JA, FERTIG
OBA8 CDB60B       2095         CALL    WCHAR           ; NEIN, AUSGEBEN UND
OBAB C3A20B       2096         JMP     WBUF1           ;       WEITER
OBAE F1           2097 WBUF2:  POP     PSW
OBAF C9           2098         RET
                  2099
                  2100 ;*************************************************************************
                  2101 ;* WBUFI -      TEXTPUFFER NACH CALL AUSGEBEN
                  2102 ;*       CALL    WBUFI
                  2103 ;*       DB      'TEXT',0
                  2104 ;*************************************************************************
OBB0 E3           2105 WBUFI:  XTHL                    ;(HL) ZEIGT AUF TEXTPUFFER NACH CALL
OBB1 CDA10B       2106         CALL    WBUF            ;AUSGEBEN; (HL) ZEIGT NUN AUF
OBB4 E3           2107         XTHL                    ;  INSTRUKTION NACH ENDEZEICHEN
OBB5 C9           2108         RET
                  2109
                  2110 ;*************************************************************************
                  2111 ;* WCHAR -      EIN ZEICHEN AUSGEBEN
                  2112 ;*       MVI     A,ZEICHEN
                  2113 ;*       CALL    WCHAR
                  2114 ;*************************************************************************
                  2115 WCHAR:
                  2116
OBB6 CD83FC       2117         CALL    SEROUT          ;ZEICHEN AUSGEBEN
OBB9 E67F         2118         ANI     7FH
OBBB FE0A         2119         CPI     LF              ;ZEILENVORSCHUB ?
OBBD C0           2120         RNZ                     ; NEIN, ZURUECK
OBBE F5           2121         PUSH    PSW             ; JA, ZEILENZAEHLER PRUEFEN U. MODIFIZIEREN
OBBF E5           2122         PUSH    H
OBC0 CDD20E       2123         CALL    CRTTST          ;CRT ?
OBC3 DAF40B       2124         JC      WCHAR3          ;NEIN
OBC6 21CCFC       2125         LXI     H,LINES         ;JA
OBC9 7E           2126         MOV     A,M
OBCA B7           2127         ORA     A               ;ZEILENZAEHLER ABGESCHALTET ?
OBCB CAF40B       2128         JZ      WCHAR3          ; JA, FERTIG
OBCE 35           2129         DCR     M               ; NEIN, ZEILENZAEHLER -1
OBCF C2F40B       2130         JNZ     WCHAR3          ;NZ: ZEILENZAEHLER NICHT 0
OBD2 360F         2131         MVI     M,NLINE         ;ZEILENZAEHLER NEU AUF MAX.
OBD4 CDB00B       2132         CALL    WBUFI           ;SAGEN, DASS SEITE VOLL
OBD7 20203D3D     2133         DB      '  ==) SPACE',CR,0
OBDB 3E20535O
OBDF 414345
OBE2 0D
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
0BE3 00
0BE4 CDE00A       2134 WCHAR2: CALL    RCHAR           ;WARTEN, BIS SPACE
0BE7 FE20         2135         CPI     ' '             ;  EINGEGEBEN WIRD
0BE9 C2E40B       2136         JNZ     WCHAR2          ;NZ: DAS WAR KEIN SPACE
0BEC CD930B       2137         CALL    WBLNKI          ;==> SPACE MIT SPACE UEBERSCHREIBEN
0BEF 14           2138         DB      20
0BF0 CDF70B       2139         CALL    WCHARI          ;ZURUECK ZUM ZEILENANFANG
0BF3 0D           2140         DB      CR
0BF4 E1           2141 WCHAR3: POP     H
0BF5 F1           2142         POP     PSW
0BF6 C9           2143         RET
                  2144
                  2145
                  2146 ;*****************************************************************************
                  2147 ;* WCHARI -     EIN ZEICHEN NACH CALL AUSGEBEN
                  2148 ;*      CALL    WCHARI
                  2149 ;*      DB      EIN ZEICHEN
                  2150 ;*****************************************************************************
0BF7 E3           2151 WCHARI: XTHL                    ;(HL) DEUTET AUF ZEICHEN NACH CALL
0BF8 F5           2152         PUSH    PSW
0BF9 7E           2153         MOV     A,M             ;ZEICHEN NACH CALL HOLEN
0BFA 23           2154         INX     H               ;(HL) AUF INSTRUKTION NACH ZEICHEN STELLEN
0BFB CDB60B       2155         CALL    WCHAR           ;ZEICHEN AUSGEBEN
0BFE F1           2156         POP     PSW
0BFF E3           2157         XTHL
0C00 C9           2158         RET
                  2159 ;*****************************************************************************
                  2160 ;* WCRLF -      NEUE ZEILE
                  2161 ;*      CALL    WCRLF
                  2162 ;*****************************************************************************
                  2163
0C01 CDB00B       2164 WCRLF:  CALL    WBUFI
0C04 0D           2165         DB      CR,LF,CR,0
0C05 0A
0C06 0D
0C07 00
0C08 CDD20E       2166         CALL    CRTTST          ;TTY ?
0C0B D8           2167         RC                      ; JA
0C0C CDB00B       2168         CALL    WBUFI           ; NEIN, ZEICHEN NACH CURSOR LOESCHEN
0C0F 20           2169         DB      ' ',BS,0
0C10 08
0C11 00
0C12 C9           2170         RET
                  2171
                  2172 ;*****************************************************************************
                  2173 ;* WCRLFI -     TEXT NACH CALL IN NEUE ZEILE AUSGEBEN
                  2174 ;*      CALL    WCRLFI
                  2175 ;*      DB      'TEXT',0
                  2176 ;*****************************************************************************
0C13 CD010C       2177 WCRLFI: CALL    WCRLF
0C16 E3           2178         XTHL                    ;(HL) DEUTET AUF 1. ZEICHEN NACH CALL
0C17 CDA10B       2179         CALL    WBUF
0C1A E3           2180         XTHL
0C1B C9           2181         RET
                  2182 ;
```

```
LOC  OBJ          LINE        SOURCE STATEMENT

                  2183 ;*************************************************************************
                  2184 ;* WHLHEX -      16BIT WERT ALS 4 HEX.ZEICHEN AUSGEBEN
                  2185 ;*************************************************************************
                  2186 ;        LXI     H,16BIT WERT
                  2187 ;        CALL    WHLHEX
0C1C F5           2188 WHLHEX: PUSH     PSW
0C1D 7C           2189         MOV      A,H             ;LINKE 8BIT ALS
0C1E CD6F0B       2190         CALL     WAHEX           ;  2 HEX.STELLEN AUSGEBEN
0C21 7D           2191         MOV      A,L             ;RECHTE 8BIT ALS
0C22 CD6F0B       2192         CALL     WAHEX           ;  2 HEX.STELLEN AUSGEBEN
0C25 F1           2193         POP      PSW
0C26 C9           2194         RET
                  2195
                  2196 ;*************************************************************************
                  2197 ;* WHLHXB -      (HL) IN HEX-ASCII AUSGEBEN, BLANK DANACH
                  2198 ;*       LXI     H,16BIT WERT
                  2199 ;*       CALL    WHLHXB
                  2200 ;*************************************************************************
0C27 CD1C0C       2201 WHLHXB: CALL     WHLHEX          ;16BIT HEX. AUSGEBEN, DANACH
0C2A CD8E0B       2202         CALL     WBLANK          ;  1 LEERZEICHEN
0C2D C9           2203         RET
                  2204
                  2205 ; RCAS -         EIN ZEICHEN VON CASSETTE LESEN
                  2206 ;        CALL    RCAS
                  2207 ;                (A) = ZEICHEN VON CASSETTE
0C2E CD86FC       2208 RCAS:   CALL     CASIN
0C31 E67F         2209         ANI      7FH
0C33 C9           2210         RET
                  2211
                  2212 ;*************************************************************************
                  2213 ;* RCASHEX -     ZEICHEN HEXADEZIMAL VON CASSETTE LESEN
                  2214 ;*       (MVI    B,0             ;CHECKSUM = 0)
                  2215 ;*       CALL    RCASHEX
                  2216 ;*               (A) = GEWANDELTER 8BIT WERT (AUS 2 HEX.ZEICHEN)
                  2217 ;*               (B) = (B)+(A) => CHECKSUM
                  2218 ;*                CY=1: FEHLER: CHECKSUM ODER ZEICHEN NICHT HEX ODER LESEFEHLER
                  2219 ;*                   0: CHECKSUM = 0, O.K.
                  2220 ;*************************************************************************
                  2221 RCASAHEX:
0C34 D5           2222         PUSH     D
0C35 CD2E0C       2223         CALL     RCAS            ;1. ZEICHEN VON CASSETTE LESEN
0C38 57           2224         MOV      D,A             ; rette Zeichen
0C39 CD8410       2225         CALL     ZUHEX           ;WANDELN
0C3C DA580C       2226         JC       RCASA1          ; C: ZEICHEN NICHT HEX
0C3F 07           2227         RLC
0C40 07           2228         RLC
0C41 07           2229         RLC
0C42 07           2230         RLC
0C43 5F           2231         MOV      E,A             ;IN E ALS LINKES NIBBLE
0C44 CD2E0C       2232         CALL     RCAS            ;2. ZEICHEN VON CASSETTE LESEN
0C47 57           2233         MOV      D,A             ; rette Zeichen, fuer Fehlerausdruck
0C48 CD8410       2234         CALL     ZUHEX           ;WANDELN
0C4B DA580C       2235         JC       RCASA1          ; C: ZEICHEN NICHT HEX
0C4E B3           2236         ORA      E
0C4F 5F           2237         MOV      E,A             ;4BIT NACH (E) ALS RECHTES NIBBLE
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
0C50 80           2238        ADD     B
0C51 47           2239        MOV     B,A             ;NEUE CHECKSUM
0C52 7B           2240        MOV     A,E             ;ERGEBNIS IN (A)
0C53 D1           2241        POP     D
0C54 37           2242        STC
0C55 C0           2243        RNZ                     ;CHECKSUM > 0
0C56 3F           2244        CMC
0C57 C9           2245        RET                     ;CHECKSUM = 0
                  2246 RCASA1:
0C58 7A           2247        MOV     A,D             ; falsches Zeichen
0C59 D1           2248        POP     D
0C5A CD130C       2249        CALL    WCRLFI
0C5D 202A2A2A     2250        DB      ' *** NICHT HEX = ',0
0C61 204E4943
0C65 48542048
0C69 45582030
0C6D 20
0C6E 00
0C6F CD6F0B       2251        CALL    WAHEX
0C72 37           2252        STC
0C73 C3AC01       2253        JMP     CMD
                  2254
                  2255 ;*****************************************************************************
                  2256 ;* WCAS -       EIN ZEICHEN AUF CASSETTE AUSGEBEN
                  2257 ;*       MVI     A,ZEICHEN
                  2258 ;*       CALL    WCAS
                  2259 ;*****************************************************************************
0C76 F5           2260 WCAS:   PUSH PSW
0C77 CD89FC       2261        CALL    CASOUT
0C7A F1           2262        POP PSW
0C7B C9           2263        RET
                  2264
                  2265 ;*****************************************************************************
                  2266 ;* WCASAHEX -   8BIT WERT HEX.ASCII AUF CASSETTE AUSGEBEN
                  2267 ;*       MVI     A,8BIT WERT
                  2268 ;*       CALL    WCASAHEX
                  2269 ;*****************************************************************************
                  2270 WCASAHEX:
0C7C F5           2271        PUSH    PSW
0C7D 0F           2272        RRC
0C7E 0F           2273        RRC
0C7F 0F           2274        RRC
0C80 0F           2275        RRC                     ;LINKES NIBBLE ZU
0C81 CDF20E       2276        CALL    HEXASC          ;  HEX.ASCII WANDELN UND
0C84 CD760C       2277        CALL    WCAS            ;  AUF CASSETTE AUSGEBEN
0C87 F1           2278        POP     PSW             ;RECHTES NIBBLE ZU
0C88 F5           2279        PUSH    PSW
0C89 CDF20E       2280        CALL    HEXASC          ;  HEX.ASCII WANDELN UND
0C8C CD760C       2281        CALL    WCAS            ;  AUF CASSETTE AUSGEBEN
0C8F F1           2282        POP     PSW
0C90 C9           2283        RET
                  2284
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT
                   2285 ;*****************************************************************************
                   2286 ;* WCASBUFI -   ZEICHENFOLGE NACH CALL AUSGEBEN
                   2287 ;*      CALL WQCASBUFI
                   2288 ;*      DB      TEXT,0
                   2289 ;*****************************************************************************
 0C91 E3           2290 WCASBUFI:       XTHL                    ;(HL) ZEIGT AUF TEXT NACH CALL
 0C92 F5           2291         PUSH    PSW
 0C93 7E           2292 WCASB1: MOV     A,M             ;ZEICHEN AUS PUFFER NACH CALL
 0C94 B7           2293         ORA     A               ;ZEICHEN = 0 (ENDE)
 0C95 C4760C       2294         CNZ     WCAS            ;NEIN, AUSGEBEN
 0C98 23           2295         INX      H              ;
 0C99 C2930C       2296         JNZ     WCASB1          ;UND WEITER
 0C9C F1           2297         POP     PSW
 0C9D E3           2298         XTHL                    ;(HL) DEUTET AUF INSTRUKTION NACH TEXT
 0C9E C9           2299         RET
                   2300
                   2301 ;*****************************************************************************
                   2302 ;* WCASHLHEX -  16BIT WEWRT IN (HL) HEXADEZIMALZAHL AUSGEBEN
                   2303 ;*      LXI     H,16BIT WERT
                   2304 ;*      CALL    WCASHLHEX
                   2305 ;*****************************************************************************
                   2306 WCASHLHEX:
 0C9F F5           2307         PUSH    PSW
 0CA0 7C           2308         MOV     A,H
 0CA1 CD7C0C       2309         CALL    WCASAHEX        ;LINKEN NIBBLE AUSGEBEN
 0CA4 7D           2310         MOV     A,L
 0CA5 CD7C0C       2311         CALL    WCASAHEX        ;RECHTER NIBBLE AUSGEBEN
 0CA8 F1           2312         POP     PSW
 0CA9 C9           2313         RET
                   2314
                   2315
                   2316
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT
                   2317 ;*************************************************************************
                   2318 ;* BCLR -        EINGABE-PUFFER LOESCHEN
                   2319 ;*************************************************************************
                   2320 ;        CALL BCLR
 0CAA E5           2321 BCLR:    PUSH    H
 0CAB F5           2322          PUSH    PSW
 0CAC 21E1FC       2323          LXI     H,BUFFER          ;PUFFERANFANG
 0CAF 36FF         2324          MVI     M,0FFH            ;0FFH ALS ANFANGSMARKE
 0CB1 3E10         2325          MVI     A,BUFLG           ;PUFFERLAENGE
 0CB3 23           2326 BCLR1:   INX     H
 0CB4 3680         2327          MVI     M,80H             ;80H = LOESCHWERT
 0CB6 3D           2328          DCR     A
 0CB7 C2B30C       2329          JNZ     BCLR1
 0CBA 23           2330          INX     H
 0CBB 36FF         2331          MVI     M,0FFH            ;0FFH ALS ENDMARKE
 0CBD F1           2332          POP     PSW
 0CBE E1           2333          POP     H
 0CBF C9           2334          RET
                   2335
                   2336 ;*************************************************************************
                   2337 ;* BGET -        ZEICHEN AUS EINGABE-PUFFER HOLEN
                   2338 ;*               (AELTESTES ODER JUENGSTES)
                   2339 ;*       LXI     H,PUFFER ANFANG/ENDE
                   2340 ;*       LXI     D,-1      ;AELTESTES, DANN BUFFER
                   2341 ;*                 +1      ;JUENGSTES, DANN BUFEND
                   2342 ;*       CALL    BGET
                   2343 ;*       JC      PUFFERLEER
                   2344 ;*       JNC     ZEICHEN IN (A)
                   2345 ;*************************************************************************
 0CC0 19           2346 BGET:    DAD     D
 0CC1 7E           2347          MOV     A,M
 0CC2 3C           2348          INR     A
 0CC3 37           2349          STC
 0CC4 C8           2350          RZ                        ;Z: PUFFER LEER
 0CC5 3D           2351          DCR     A
 0CC6 FAC00C       2352          JM      BGET
 0CC9 3680         2353          MVI     M,80H             ;GELESENES ZEICHEN DURCH 80H ERSETZEN
 0CCB B7           2354          ORA     A
 0CCC C9           2355          RET
                   2356
                   2357 ;*************************************************************************
                   2358 ;* BGETF -       ERSTES (JUENGSTES) ZEICHEN AUS EINGABE-PUFFER HOLEN
                   2359 ;*       CALL    BGETF
                   2360 ;*       JC      PUFFERLEER
                   2361 ;*       JNC     ZEICHEN IN (A)
                   2362 ;*************************************************************************
 0CCD E5           2363 BGETF:   PUSH    H
 0CCE D5           2364          PUSH    D
 0CCF 21E1FC       2365          LXI     H,BUFFER
 0CD2 110100       2366          LXI     D,1
 0CD5 CDC00C       2367          CALL    BGET
 0CD8 D1           2368          POP     D
 0CD9 E1           2369          POP     H
 0CDA C9           2370          RET
                   2371
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                  2372 ;*************************************************************************
                  2373 ;* BGETL -       LETZTES (AELTESTES) ZEICHEN AUS EINGABEPUFFER HOLEN
                  2374 ;*       CALL    BGETL
                  2375 ;*               SONST SIEHE BGETF
                  2376 ;*************************************************************************
0CDB E5           2377 BGETL:   PUSH    H
0CDC D5           2378          PUSH    D
0CDD 21F2FC       2379          LXI     H,BUFEND
0CE0 11FFFF       2380          LXI     D,-1
0CE3 CDC00C       2381          CALL    BGET
0CE6 D1           2382          POP     D
0CE7 E1           2383          POP     H
0CE8 C9           2384          RET
                  2385
                  2386 ;*************************************************************************
                  2387 ;* BPUT -        ZEICHEN IN EINGABE-PUFFER ABLEGEN
                  2388 ;*       MVI     A,ZEICHEN
                  2389 ;*       CALL    BPUT
                  2390 ;*       JC      PUFFER VOLL
                  2391 ;*       JNC     NOCH PLAETZE FREI IM PUFFER
                  2392 ;*************************************************************************
0CE9 E5           2393 BPUT:    PUSH    H
0CEA C5           2394          PUSH    B
0CEB 4F           2395          MOV     C,A             ;ZEICHEN IN (C) RETTEN
0CEC 21F2FC       2396          LXI     H,BUFEND
0CEF 2B           2397 BPUT1:   DCX     H
0CF0 7E           2398          MOV     A,M
0CF1 B7           2399          ORA     A
0CF2 F2EF0C       2400          JP      BPUT1
0CF5 3C           2401          INR     A               ;PUFFERENDE ?
0CF6 CA060D       2402          JZ      BPUT3           ; JA
0CF9 71           2403          MOV     M,C             ; NEIN, ABSPEICHERN
0CFA 2B           2404          DCX     H
0CFB 7E           2405          MOV     A,M
0CFC 3C           2406          INR     A               ;PUFFERENDE ?
0CFD CA020D       2407          JZ      BPUT2           ; JA
0D00 3680         2408          MVI     M,80H           ; NEIN, NAECHSTEN FREIEN PLATZ MARKIEREN
0D02 79           2409 BPUT2:   MOV     A,C             ;(A) RESTAURIEREN
0D03 C1           2410          POP     B
0D04 E1           2411          POP     H
0D05 C9           2412          RET
0D06 37           2413 BPUT3:   STC                     ;CARRY=1 ZEIGT AN, DASS PUFFER LEER WAR
0D07 C3020D       2414          JMP     BPUT2
                  2415
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT
                   2416 ;*************************************************************************
                   2417 ;* BREAD -       ZEICHEN EINLESEN, KONTROLLIEREN UND PUFFERN
                   2418 ;*               RUB OUT:          ZULETZT EINGEGEBENES ZEICHEN LOESCHEN
                   2419 ;*               CR ODER SPACE:    EINGABE NORMAL BEENDEN
                   2420 ;*               + ODER -:         EINGABE BEENDEN, WENN BCKFLG=1
                   2421 ;*               ESC:              EINGABE ABBRECHEN
                   2422 ;*               KONTR.ZEICHEN:    WERDEN IGNORIERT
                   2423 ;*               MVI     B,MAXIMALE ZAHL DER ZEICHEN
                   2424 ;*               MVI     C,TYP DER EINGEGEBENEN ZEICHEN:  0 = ASCII
                   2425 ;*                                                        1 = BINAER
                   2426 ;*                                                        2 = DEZIMAL
                   2427 ;*                                                        3 = HEXADEZIMAL
                   2428 ;*       CALL    BREAD
                   2429 ;*       JC      ABBRUCH
                   2430 ;*       JNC     CR ODER BLANK (ODER - ODER + , FALLS BCKFLG=1)
                   2431 ;*               (A) = ABSCHLUSSZEICHEN
                   2432 ;*************************************************************************
 0D0A E5           2433 BREAD:  PUSH    H
                   2434
 0D0B D5           2435 PUSH    D
 0D0C C5           2436         PUSH    B
 0D0D 1600         2437         MVI     D,0
 0D0F CDE00A       2438 BREAD0: CALL    RCHAR           ;ZEICHENEINGABE
 0D12 5F           2439         MOV     E,A
 0D13 3AC9FC       2440         LDA     GROFLG
 0D16 B7           2441         ORA     A               ;NUR GROSSBUCHSTABEN ?
 0D17 7B           2442         MOV     A,E
 0D18 CCE90E       2443         CZ      GROSS           ; JA
 0D1B 0C           2444         INR     C
 0D1C 0D           2445         DCR     C               ;ASCII-FORMAT ?
 0D1D C4E90E       2446         CNZ     GROSS           ; NEIN, GROSSBUCHSTABEN
 0D20 5F           2447         MOV     E,A             ;ZEICHEN IN E RETTEN
                   2448
 0D21 FE0D         2449         CPI     CR              ;==> EINGABE BEENDET DURCH CR ?
 0D23 CA2B0D       2450         JZ      BREAD2          ; JA
 0D26 FE20         2451 BREAD1: CPI     ' '             ; NEIN; ==> EINGABE BEENDET DURCH BLANK ?
 0D28 C23B0D       2452         JNZ     BREAD3          ;       NEIN
                   2453
 0D2B CDD20E       2454 BREAD2: CALL    CRTTST          ;***    ENDE DER EINGABE; CRT ?
 0D2E DA370D       2455         JC      BREA22          ;*       NEIN
 0D31 CDB00B       2456         CALL    UBUFI           ;*       JA, BLANK ALS ECHO
 0D34 20           2457         DB      ' '             ;*
 0D35 08           2458         DB      BS              ;*
 0D36 00           2459         DB      0               ;*
 0D37 B7           2460 BREA22: ORA     A               ;*      CARRY=0: NORMALES ENDE
 0D38 C3C10D       2461         JMP     BREAD9          ;***    EXIT
                   2462
 0D3B FE20         2463 BREAD3: CPI     20H             ;==> SONSTIGES KONTROLLZEICHEN ?
 0D3D D2460D       2464         JNC     BREAD5          ; NEIN, KEIN KONTROLLZEICHEN
 0D40 CD090B       2465 BREAD4: CALL    UBELL           ; JA,   BIMM !
 0D43 C30F0D       2466         JMP     BREAD0          ;NAECHSTES ZEICHEN EINLESEN
                   2467
 0D46 FE7F         2468 BREAD5: CPI     RUBOUT          ;==> RUB OUT ?
 0D48 C2810D       2469         JNZ     BREAD6          ; NEIN
                   2470
```

```
LOC  OBJ        LINE        SOURCE STATEMENT
0D4B CDD20E      2471          CALL    CRTTST          ;***    RUB-OUT; CRT ?
0D4E DA630D      2472          JC      BREA51          ;*       NEIN
0D51 CDCD0C      2473          CALL    BGETF           ;*       JA, ZEICHEN AUS PUFFER LOESCHEN
0D54 DA0F0D      2474          JC      BREAD0          ;*      PUFFER WAR LEER
0D57 CDB00B      2475          CALL    WBUFI           ;*
0D5A 08          2476          DB      BS              ;*
0D5B 20          2477          DB      ' '             ;*
0D5C 20          2478          DB      ' '             ;*
0D5D 08          2479          DB      BS              ;*
0D5E 08          2480          DB      BS              ;*
0D5F 00          2481          DB      0               ;*
0D60 C3740D      2482          JMP     BREA52          ;*      ENDE RUB-OUT FUER CRT
0D63 3ACAFC      2483 BREA51:  LDA     RUBFLG          ;*      TTY
0D66 B7          2484          ORA     A               ;*      RUB-OUT ERLAUBT ?
0D67 C2400D      2485          JNZ     BREAD4          ;*       NEIN, BIMM !
0D6A CDCD0C      2486          CALL    BGETF           ;*       JA,   ZEICHEN AUS PUFFER HOLEN
0D6D DA0F0D      2487          JC      BREAD0          ;*      PUFFER LEER, NAECHSTES ZEICHEN
0D70 CDF70B      2488          CALL    WCHARI          ;*      SCHRAEGSTRICH ALS ECHO
0D73 2F          2489          DB      '/'             ;*
0D74 7A          2490 BREA52:  MOV     A,D             ;*      (D) DURCH 10 DIVIDIEREN
0D75 16FF        2491          MVI     D,-1            ;*
0D77 C6F6        2492 BREA53:  ADI     -10             ;*         SOLANGE 10 SUBTRAHIEREN,
0D79 14          2493          INR     D               ;*         BIS (A) KLEINER ALS NULL; FERTIG ?
0D7A DA770D      2494          JC      BREA53          ;*           NEIN, WEITER -10 ADDIEREN
0D7D 04          2495          INR     B               ;*           JA  , FERTIG
0D7E C30F0D      2496          JMP     BREAD0          ;***    NAECHSTES ZEICHEN EINLESEN
                 2497
0D81 3AC7FC      2498 BREAD6:  LDA     BCKFLG          ;==> PRUEFEN, OB ENDE DER EINGABE DURCH '-'/'+'
0D84 B7          2499          ORA     A               ;'-'/'+' ERLAUBT ?
0D85 7B          2500          MOV     A,E             ;
0D86 CA960D      2501          JZ      BREAD7          ; NEIN
0D89 FE2B        2502          CPI     '+'             ; JA
0D8B CA2B0D      2503          JZ      BREAD2          ;ENDE DER EINGABE
0D8E FE2D        2504          CPI     '-'             ;
0D90 C2960D      2505          JNZ     BREAD7          ;WEITER PRUEFEN
0D93 C32B0D      2506          JMP     BREAD2          ;ENDE DER EINGABE
                 2507
0D96 CD840E      2508 BREAD7:  CALL    CHTST           ;==> ZEICHEN LEGAL ?
0D99 D2AE0D      2509          JNC     BREAD8          ; JA
0D9C CDD20E      2510 BREA71:  CALL    CRTTST          ; NEIN; CRT ?
0D9F DA400D      2511          JC      BREAD4          ;TTY
0DA2 7B          2512          MOV     A,E             ;CRT
0DA3 CDB60B      2513          CALL    WCHAR           ;ECHO DES ILLEGALEN ZEICHENS
0DA6 CDB00B      2514          CALL    WBUFI           ;
0DA9 08          2515          DB      BS,0            ;
0DAA 00
0DAB C3400D      2516          JMP     BREAD4          ;BIMM !
                 2517
0DAE 78          2518 BREAD8:  MOV     A,B             ;==> MAXIMALE ZEICHENZAHL ERREICHT ?
0DAF B7          2519          ORA     A               ;MAX. ERREICHT ?
0DB0 CA9C0D      2520          JZ      BREA71          ; JA, BIMMELN
                 2521
0DB3 7B          2522          MOV     A,E             ;***    ZEICHEN PUFFERN
0DB4 CDE90C      2523          CALL    BPUT            ;*      ZEICHEN REIN IN PUFFER
0DB7 DA400D      2524          JC      BREAD4          ;*      PUFFER WAR VOLL
```

```
LOC  OBJ        LINE        SOURCE STATEMENT
ODBA CDB60B     2525        CALL    WCHAR           ;*      ECHO (ZEICHEN IST GEPUFFERT)
ODBD 05         2526        DCR     B               ;*
ODBE C30F0D     2527        JMP     BREAD0          ;***    NAECHSTES ZEICHEN HOLEN
                2528
ODC1 C1         2529 BREAD9: POP    B               ;***    EXIT
ODC2 D1         2530        POP     D               ;*      CY=0: NORMALES ENDE
ODC3 E1         2531        POP     H               ;*         1: ABBRUCH DURCH ESC
ODC4 C9         2532        RET                     ;***    ZURUECK
                2533
                2534 ;*************************************************************************
                2535 ;* BPTINS -    BREAKPOINTS IM PROGRAMM EINFUEGEN
                2536 ;*             BREAKPOINTS WERDEN NICHT IM PROGRAMM EINGEFUEGT, WENN:
                2537 ;*             (1) BREAKPOINTS NICHT EINGESCHALTET
                2538 ;*             (2) WENN PROGRAMM-OPCODE = RST 4
                2539 ;*       CALL  BPTINS
                2540 ;*             CY=1: (PC)-1 UNGLEICH BREAK-ADRESSEN
                2541 ;*                =0: (PC)-1 GLEICH EINER DER BREAKADRESSEN
                2542 ;*************************************************************************
ODC5 E5         2543 BPTINS: PUSH   H               ;ALLE REGISTER RETTEN
ODC6 D5         2544        PUSH    D
ODC7 C5         2545        PUSH    B
ODC8 F5         2546        PUSH    PSW
ODC9 2AD7FC     2547        LHLD    PCWERT
ODCC 2B         2548        DCX     H               ;(PC)-1 AUF
ODCD E5         2549        PUSH    H               ; STACK SPEICHERN
ODCE 21A2FC     2550        LXI     H,BPTADR
ODD1 0600       2551        MVI     B,0             ;ANFANGS KEINE BREAKADR = (PC)-1
ODD3 0E04       2552        MVI     C,BPTANZ        ;ANZAHL DER BREAKPOINTS
ODD5 3AC6FC     2553        LDA     BPTFLG
ODD8 B7         2554        ORA     A               ;BREAKPOINTS EINGESCHALTET ?
ODD9 CAFE0D     2555        JZ      BPTIN4          ; NEIN, EXIT
ODDC 5E         2556 BPTIN1: MOV    E,M             ; JA, BREAK-ADR HOLEN
ODDD 23         2557        INX     H
ODDE 56         2558        MOV     D,M
ODDF 23         2559        INX     H
ODE0 7B         2560        MOV     A,E
ODE1 B2         2561        ORA     D               ;BREAK-ADR = 0 ?
ODE2 CAF90D     2562        JZ      BPTIN3          ; JA
ODE5 EB         2563        XCHG                    ; NEIN
ODE6 7E         2564        MOV     A,M             ;OPCODE NACH (A)
ODE7 36E7       2565        MVI     M,0E7H          ;DAFUER RST 4 EINFUEGEN
ODE9 EB         2566        XCHG
ODEA FEE7       2567        CPI     0E7H            ;OPCODE = RST 4 ?
ODEC CAF00D     2568        JZ      BPTIN2          ; NEIN, OPCODE NICHT RETTEN
ODEF 77         2569        MOV     M,A             ; JA, OPCODE RETTEN
ODF0 E3         2570 BPTIN2: XTHL                   ;(PC)-1 --> (HL)
ODF1 CDA80E     2571        CALL    CMP2            ;(PC)-1 = BREAK-ADR ?
ODF4 E3         2572        XTHL
ODF5 C2F90D     2573        JNZ     BPTIN3          ; NEIN
ODF8 04         2574        INR     B               ; JA, MERKE, DASS (PC)-1 = BREAK-ADR
ODF9 23         2575 BPTIN3: INX    H               ;OPCODE SKIPPEN
ODFA 0D         2576        DCR     C               ;ALLE BREAKPOINTS EINGESETZT ?
ODFB C2DC0D     2577        JNZ     BPTIN1          ; NEIN
ODFE 78         2578 BPTIN4: MOV    A,B             ; JA
ODFF D601       2579        SUI     1               ;CY=0, WENN (PC)-1 = BREAK-ADR, SONST 1
```

```
LOC  OBJ        LINE        SOURCE STATEMENT
0E01 3F         2580        CMC                 ;CARRY TOGGELN
0E02 E1         2581        POP     H
0E03 C1         2582        POP     B           ;(A) UEBER (B) RESTAURIEREN,
0E04 78         2583        MOV     A,B         ; DA POP PSW CARRY AENDERN WUERDE
0E05 C1         2584        POP     B
0E06 D1         2585        POP     D
0E07 E1         2586        POP     H
0E08 C9         2587        RET
                2588
                2589 ;*****************************************************************************
                2590 ;* BPTNUM -    ANZAHL DER BREAKPOINTS IN (C) ABLIEFERN
                2591 ;*       CALL    BPTNUM
                2592 ;*               (C) = ANZAHL BREAKPOINTS
                2593 ;*****************************************************************************
0E09 0E04       2594 BPTNUM: MVI     C,BPTANZ
0E0B C9         2595        RET
                2596
                2597 ;*****************************************************************************
                2598 ;* BPTREM -    BREAKPOINTS AUS PROGRAMM ENTFERNEN
                2599 ;*       CALL    BPTREM
                2600 ;*****************************************************************************
0E0C E5         2601 BPTREM: PUSH    H           ;ALLE REGISTER RETTEN
0E0D D5         2602        PUSH    D
0E0E C5         2603        PUSH    B
0E0F F5         2604        PUSH    PSW
0E10 3AC6FC     2605        LDA     BPTFLG
0E13 B7         2606        ORA     A           ;BREAKPOINTS EINGESCHALTET ?
0E14 CA320E     2607        JZ      BPTRE3      ; NEIN, EXIT
0E17 21A2FC     2608        LXI     H,BPTADR    ; JA, BREAKPOINTS ENTFERNEN
0E1A 0E04       2609        MVI     C,BPTANZ
0E1C 5E         2610 BPTRE1: MOV     E,M         ;BREAK-ADR HOLEN
0E1D 23         2611        INX     H
0E1E 56         2612        MOV     D,M
0E1F 23         2613        INX     H
0E20 7B         2614        MOV     A,E
0E21 B2         2615        ORA     D           ;BREAK-ADR = 0 ?
0E22 CA2D0E     2616        JZ      BPTRE2      ; JA
0E25 1A         2617        LDAX    D           ; NEIN, PROGRAMM-OPCODE HOLEN
0E26 FEE7       2618        CPI     0E7H        ;RST 4 ? (MUSS!)
0E28 C22D0E     2619        JNZ     BPTRE2      ; NEIN, NICHTS TUN
0E2B 7E         2620        MOV     A,M         ; JA, ALTEN OPCODE HOLEN
0E2C 12         2621        STAX    D           ;     UND WIEDER EINFUEGEN
0E2D 23         2622 BPTRE2: INX     H
0E2E 0D         2623        DCR     C
0E2F C21C0E     2624        JNZ     BPTRE1
0E32 F1         2625 BPTRE3: POP     PSW
0E33 C1         2626        POP     B
0E34 D1         2627        POP     D
0E35 E1         2628        POP     H
0E36 C9         2629        RET
                2630
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                  2631 ;*************************************************************************
                  2632 ;* BPTSET -     BREAKPOINTADRESSE PRUEFEN UND ABSPEICHERN
                  2633 ;*       LXI    H,BREAKPOINTADRESSE
                  2634 ;*       MVI    B,BREAKPOINTNUMMER (0,1,...)
                  2635 ;*                WENN (B) < 0: ADRESSE WIRD NUR GEPRUEFT
                  2636 ;*       CALL   BPTSET
                  2637 ;*              CY=1: ANGEGEBENE BREAK-ADR SCHON DA
                  2638 ;*                =0: O.K., WURDE ABGESPEICHERT
                  2639 ;*************************************************************************
0E37 D5           2640 BPTSET: PUSH    D
0E38 C5           2641         PUSH    B
0E39 F5           2642         PUSH    PSW
0E3A E5           2643         PUSH    H
0E3B 7C           2644         MOV     A,H
0E3C B5           2645         ORA     L              ;BREAK-ADR = 0 ?
0E3D C2460E       2646         JNZ     BPTSE0         ; NEIN
0E40 21A2FC       2647         LXI     H,BPTADR       ; JA, ABSPEICHERN, WENN (B)>0
0E43 C35D0E       2648         JMP     BPTSE2
0E46 21AEFC       2649 BPTSE0: LXI     H,BPTADR+3*BPTANZ
0E49 0E04         2650         MVI     C,BPTANZ
0E4B 2B           2651 BPTSE1: DCX     H              ;BEREITS GESETZTE BREAK-ADR HOLEN
0E4C 2B           2652         DCX     H
0E4D 56           2653         MOV     D,M
0E4E 2B           2654         DCX     H
0E4F 5E           2655         MOV     E,M
0E50 E3           2656         XTHL
0E51 CDA80E       2657         CALL    CMP2           ;GLEICH NEUER BREAK-ADR ?
0E54 E3           2658         XTHL
0E55 37           2659         STC
0E56 CA6E0E       2660         JZ      BPTSE4         ; JA
0E59 0D           2661         DCR     C              ; NEIN, MIT NAECHSTER BREAK-ADR VERGLEICHEN
0E5A C24B0E       2662         JNZ     BPTSE1
0E5D D1           2663 BPTSE2: POP     D
0E5E D5           2664         PUSH    D
0E5F 78           2665         MOV     A,B
0E60 87           2666         ADD     A              ;MAL 2: (B)<0 ?
0E61 DA6C0E       2667         JC      BPTSE3         ; JA, FERTIG
0E64 80           2668         ADD     B              ; NEIN, MAL 3
0E65 4F           2669         MOV     C,A
0E66 0600         2670         MVI     B,0            ;(BC) ZEIGT IN BREAKPOINT-SPEICHER
0E68 09           2671         DAD     B
0E69 73           2672         MOV     M,E            ;BREAK-ADR ABSPEICHERN
0E6A 23           2673         INX     H
0E6B 72           2674         MOV     M,D
0E6C EB           2675 BPTSE3: XCHG
0E6D B7           2676         ORA     A              ;CY=0
0E6E E1           2677 BPTSE4: POP     H
0E6F C1           2678         POP     B
0E70 78           2679         MOV     A,B
0E71 C1           2680         POP     B
0E72 D1           2681         POP     D
0E73 C9           2682         RET
                  2683
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                  2684 ;*****************************************************************
                  2685 ;* CALLHL -      UEBER (HL) SPRINGEN
                  2686 ;*       CALL    CALLHL
                  2687 ;*****************************************************************
0E74 E9           2688 CALLHL: PCHL                    ;RUECKKEHR DIREKT ZU RUFENDEM PROGRAMM
                  2689
                  2690 ; CALLTB -      UEBER ADRESSTABELLE SPRINGEN
                  2691 ;       MVI     C,TABELLENINDEX
                  2692 ;       LXI     H,ADRESSTABELLE
                  2693 ;       CALL    CALLTB
0E75 D5           2694 CALLTB: PUSH     D
0E76 F5           2695         PUSH     PSW
0E77 79           2696         MOV      A,C
0E78 87           2697         ADD      A
0E79 5F           2698         MOV      E,A
0E7A 1600         2699         MVI      D,0
0E7C 19           2700         DAD      D
0E7D 5E           2701         MOV      E,M
0E7E 23           2702         INX      H
0E7F 56           2703         MOV      D,M             ;(DE) = ADRESSE AUS TABELLE
0E80 EB           2704         XCHG
0E81 F1           2705         POP      PSW
0E82 D1           2706         POP      D
0E83 E9           2707         PCHL                     ;RUECKKEHR DIREKT ZU RUFENDEM PROGRAMM
                  2708
                  2709 ;*****************************************************************
                  2710 ;* CHTST -       ZEICHEN PRUEFEN
                  2711 ;*       MVI     A,ASCII-ZEICHEN
                  2712 ;*       MVI     C,TYP (0=ASCII, 1=BIN, 2=DEZ, 3=HEX)
                  2713 ;*       CALL    CHTST
                  2714 ;*               CY=1: FALSCH
                  2715 ;*                  0: O.K.
                  2716 ;*****************************************************************
0E84 E5           2717 CHTST:  PUSH     H
0E85 C5           2718         PUSH     B
0E86 D5           2719         PUSH     D
0E87 F5           2720         PUSH     PSW
0E88 21A00E       2721         LXI      H,CHTAB
0E8B 79           2722         MOV      A,C             ;TABELLENADR. BERECHNEN
0E8C 87           2723         ADD      A
0E8D 4F           2724         MOV      C,A
0E8E 0600         2725         MVI      B,0
0E90 09           2726         DAD      B
0E91 F1           2727         POP      PSW
0E92 5E           2728         MOV      E,M
0E93 23           2729         INX      H
0E94 56           2730         MOV      D,M
0E95 EB           2731         XCHG
0E96 D1           2732         POP      D
0E97 D5           2733         PUSH     D
0E98 CD740E       2734         CALL     CALLHL
0E9B C1           2735         POP      B
0E9C 59           2736         MOV      E,C             ;(E) RESTAURIEREN
0E9D C1           2737         POP      B
0E9E E1           2738         POP      H
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
0E9F C9           2739        RET
0EA0 4010         2740 CHTAB: DW      ZUASC,ZUBIN,ZUDEZ3,ZUHEX
0EA2 4510
0EA4 5E10
0EA6 8410
                  2741
                  2742 ;*****************************************************************
                  2743 ;* CMP2 -      VERGLEICHE (DE) MIT (HL)
                  2744 ;*       LXI   D,WERT-1
                  2745 ;*       LXI   H,WERT-2
                  2746 ;*       CALL  CMP2
                  2747 ;*             CY=1: (HL) ) (DE)
                  2748 ;*                0: SONST
                  2749 ;*****************************************************************
0EA8 7B           2750 CMP2:  MOV     A,E
0EA9 95           2751        SUB     L
0EAA 7A           2752        MOV     A,D
0EAB 9C           2753        SBB     H
0EAC C0           2754        RNZ
0EAD 7B           2755        MOV     A,E
0EAE 95           2756        SUB     L
0EAF C9           2757        RET
                  2758
                  2759 ;*****************************************************************
                  2760 ;* CMPL -      VERGLEICHE ZEICHEN MIT LISTE
                  2761 ;*       MVI   A,ASCII-ZEICHEN
                  2762 ;*       LXI   H,ZEICHENLISTE
                  2763 ;*       CALL  CMPL
                  2764 ;*             (C)=POSITION IN LISTE (0=1.ZEICHEN)
                  2765 ;*       JC    KEINESGLEICH
                  2766 ;*       JNC   EINESISTGLEICH
                  2767 ;*****************************************************************
0EB0 C5           2768 CMPL:  PUSH    B
0EB1 47           2769        MOV     B,A
0EB2 0EFF         2770        MVI     C,-1
0EB4 0C           2771 CMPL1: INR     C
0EB5 7E           2772        MOV     A,M
0EB6 23           2773        INX     H
0EB7 B7           2774        ORA     A               ;LISTENENDE ?
0EB8 CAC60E       2775        JZ      CMPL3           ;Z: JA
0EBB B8           2776        CMP     B               ;   NEIN; VERGLEICHEN
0EBC C2B40E       2777        JNZ     CMPL1           ;NZ: NICHT GLEICH, WEITER
0EBF 7E           2778 CMPL2: MOV     A,M
0EC0 23           2779        INX     H
0EC1 B7           2780        ORA     A
0EC2 C2BF0E       2781        JNZ     CMPL2
0EC5 37           2782        STC                     ;C: NICHTS GEFUNDEN
0EC6 78           2783 CMPL3: MOV     A,B
0EC7 E3           2784        XTHL
0EC8 44           2785        MOV     B,H
0EC9 E1           2786        POP     H
0ECA 3F           2787        CMC
0ECB C9           2788        RET
                  2789
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT
                   2790 ;*************************************************************************
                   2791 ;* CMPLI -      VERGLEICHE ZEICHEN MIT LISTE NACH CALL
                   2792 ;*       MVI     A,ZU VERGLEICHENDES ZEICHEN
                   2793 ;*       CALL    CMPLI
                   2794 ;*       DB      'ZEICHEN-1',...'ZEICHEN-N',0
                   2795 ;*               SONST WIE CMPL
                   2796 ;*************************************************************************
 0ECC E3           2797 CMPLI:  XTHL
 0ECD CD800E       2798         CALL    CMPL
 0ED0 E3           2799         XTHL
 0ED1 C9           2800         RET
                   2801 ;*************************************************************************
                   2802 ;* CRTTST -     TESTE, OB CRT
                   2803 ;*       CALL    CRTTST
                   2804 ;*       JC      TTY
                   2805 ;*       JNC     CRT
                   2806 ;*************************************************************************
 0ED2 C5           2807 CRTTST: PUSH    B
 0ED3 47           2808         MOV     B,A
 0ED4 3AC8FC       2809         LDA     CRTFLG
 0ED7 B7           2810         ORA     A               ;CRT ?
 0ED8 78           2811         MOV     A,B
 0ED9 C1           2812         POP     B
 0EDA C8           2813         RZ                      ;CY=0 crt
 0EDB 37           2814         STC                     ;CY=1 tty
 0EDC C9           2815         RET
                   2816
                   2817 ;*************************************************************************
                   2818 ;* DISLINE -    DISASSEMBLIERE EINE INSTRUKTION
                   2819 ;*       LXI     H,ADRESSE DER INSTRUKTION
                   2820 ;*       CALL    DISLINE         ;INSTRUKTION DISASSEMBLIEREN
                   2821 ;*************************************************************************
                   2822 DISLINE:
 0EDD E5           2823         PUSH    H
 0EDE D5           2824         PUSH    D
 0EDF CDF919       2825         CALL    DISA1           ;EINE INSTRUKTION: (HL)=PC
 0EE2 EB           2826         XCHG                    ;(HL) = ZEICHENPUFFER
 0EE3 CDA10B       2827         CALL    WBUF            ;DISASSEMBLIERTE INSTRUKTION DRUCKEN
 0EE6 D1           2828         POP     D
 0EE7 E1           2829         POP     H
 0EE8 C9           2830         RET
                   2831
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                  2832 ;*************************************************************************
                  2833 ;* GROSS -      KLEINE BUCHSTABEN IN GROSSE WANDELN
                  2834 ;*       LDA    ASCII-ZEICHEN
                  2835 ;*       CALL   GROSS
                  2836 ;*              (A) = GROSSBUCHSTABE, WENN VORHER KLEIN
                  2837 ;*************************************************************************
0EE9 FE61         2838 GROSS:  CPI    'A'+20H          ;(KLEIN-A)
0EEB F8           2839         RM
0EEC FE7B         2840         CPI    'Z'+20H+1        ;(KLEIN-Z + 1)
0EEE F0           2841         RP
0EEF E6DF         2842         ANI    NOT 20H          ;AUS KLEIN MACH' GROSS
0EF1 C9           2843         RET
                  2844
                  2845 ;*************************************************************************
                  2846 ;* HEXASC -     HEX.ZAHL IN ASCIIWANDELN
                  2847 ;*************************************************************************
0EF2 E60F         2848 HEXASC: ANI    0FH
0EF4 C690         2849         ADI    90H
0EF6 27           2850         DAA
0EF7 CE40         2851         ACI    40H
0EF9 27           2852         DAA
0EFA C9           2853         RET
                  2854
                  2855 ;*************************************************************************
                  2856 ;* PREGF -      TITELZEILE, PC UND DISASSEMBLIERTER BEFEHL
                  2857 ;*       LHLD   PCWERT
                  2858 ;*       CALL   PREGF
                  2859 ;*************************************************************************
0EFB CDAE0F       2860 PREGF:  CALL   PREGT            ;TITEL AUSDRUCKEN
0EFE CD130C       2861         CALL   WCRLFI
0F01 20           2862         DB     ' ',0
0F02 00
0F03 CD270C       2863         CALL   WHLHXB           ;PC DRUCKEN
0F06 CDDD0E       2864         CALL   DISLINE          ;OPCODE DISASSEMBLIEREN
0F09 C9           2865         RET
                  2866
                  2867 ;*************************************************************************
                  2868 ;* PREGP -      REGISTERAUSDRUCK POSITIONIEREN
                  2869 ;*       CALL   PREGP
                  2870 ;*************************************************************************
0F0A CD010C       2871 PREGP:  CALL   WCRLF
0F0D CD930B       2872         CALL   WBLNKI
0F10 1D           2873         DB     29
0F11 C9           2874         RET
                  2875
                  2876 ;*************************************************************************
                  2877 ;* PREGS -      ALLE REGISTER AUSDRUCKEN
                  2878 ;*       CALL   PREGS
                  2879 ;*************************************************************************
0F12 E5           2880 PREGS:  PUSH   H
0F13 216F0F       2881         LXI    H,REGTB
0F16 CD1B0F       2882         CALL   PREGSO
0F19 E1           2883         POP    H
0F1A C9           2884         RET
0F1B 7E           2885 PREGSO: MOV    A,M              ;ALLE REGISTER ENTSPR. LISTE NACH (HL)
```

9. ALLGEMEINE HILFSROUTINEN

```
LOC  OBJ       LINE        SOURCE STATEMENT
0F1C 23        2886        INX     H
0F1D B7        2887        ORA     A           ;ENDE DER REGISTERLISTE ?
0F1E C8        2888        RZ                  ; JA, FERTIG
0F1F F5        2889        PUSH    PSW         ; NEIN, AUSDRUCKEN
0F20 CD490F    2890        CALL    PREGS2
0F23 F1        2891        POP     PSW
0F24 F5        2892        PUSH    PSW
0F25 FE8A      2893        CPI     8AH
0F27 C48E0B    2894        CNZ     WBLANK      ; KEIN BLANK AM ENDE DER ZEILE
0F2A F1        2895        POP     PSW
0F2B E6F0      2896        ANI     0F0H
0F2D FEC0      2897        CPI     0C0H        ;(PC) OHNE INSTR. ?
0F2F C23D0F    2898        JNZ     PREGS1      ; NEIN
0F32 E5        2899        PUSH    H           ; JA
0F33 2AD7FC    2900        LHLD    PCWERT
0F36 CDDD0E    2901        CALL    DISLINE     ;INSTR. DISASSEMBLIEREN
0F39 E1        2902        POP     H
0F3A C31B0F    2903        JMP     PREGS0
0F3D FED0      2904 PREGS1: CPI     0D0H        ;(PC) OHNE INSTRUKTION ?
0F3F C21B0F    2905        JNZ     PREGS0      ; NEIN, NAECHSTES REGISTER
0F42 CD930B    2906        CALL    WBLNKI      ; JA,   INSTR.FELD UEBERGEHEN
0F45 17        2907        DB      23
0F46 C31B0F    2908        JMP     PREGS0
0F49 E5        2909 PREGS2: PUSH    H           ;REGISTERINHALT AUSDRUCKEN
0F4A C5        2910        PUSH    B
0F4B D5        2911        PUSH    D
0F4C 4F        2912        MOV     C,A
0F4D 21CFFC    2913        LXI     H,REGW      ;REGISTER HOLEN
0F50 E60F      2914        ANI     0FH
0F52 5F        2915        MOV     E,A
0F53 1600      2916        MVI     D,0
0F55 19        2917        DAD     D
0F56 79        2918        MOV     A,C
0F57 07        2919        RLC
0F58 07        2920        RLC
0F59 E603      2921        ANI     3
0F5B 4F        2922        MOV     C,A         ;(C) = REGISTERTYP
0F5C EB        2923        XCHG
0F5D 21670F    2924        LXI     H,PREGTB
0F60 CD750E    2925        CALL    CALLTB
0F63 D1        2926        POP     D
0F64 C1        2927        POP     B
0F65 E1        2928        POP     H
0F66 C9        2929        RET
0F67 7B0F      2930 PREGTB: DW      PREG1       ; 8BIT HEX
0F69 800F      2931        DW      PREG2       ; 5BIT BIN; CONDITION FLAGS
0F6B 9B0F      2932        DW      PREG3       ;16BIT HEX
0F6D A40F      2933        DW      PREG4       ;16BIT HEX; CR+LF+SPACE ZUVOR
               2934
0F6F 01        2935 REGTB:  DB      AREG+0      ;HEX, 2 BYTES
0F70 40        2936        DB      FREG+040H   ;BINAER, 5BIT
0F71 03        2937        DB      BREG+0      ;HEX, 2 BYTES
0F72 02        2938        DB      CREG+0      ;HEX, 2 BYTES
0F73 05        2939        DB      DREG+0      ;HEX, 2 BYTES
0F74 04        2940        DB      EREG+0      ;HEX, 2 BYTES
```

9. ALLGEMEINE HILFSROUTINEN

```
LOC  OBJ        LINE        SOURCE STATEMENT
0F75 07         2941        DB      HREG+0          ;HEX, 2 BYTES
0F76 06         2942        DB      LREG+0          ;HEX, 2 BYTES
0F77 0C         2943        DB      IREG+0          ;HEX, 2 BYTES
0F78 8A         2944        DB      SREG+080H       ;HEX, 4 BYTES
0F79 C8         2945        DB      PREG+0C0H       ;HEX, 4BYTES UND CR+LF ZUVOR
0F7A 00         2946        DB      0               ;***TABELLENENDE
                2947
0F7B 1A         2948 PREG1: LDAX    D               ;8BIT-REGISTER HEX AUSDRUCKEN
0F7C CD6F0B     2949        CALL    WAHEX
0F7F C9         2950        RET
                2951
0F80 1A         2952 PREG2: LDAX    D               ;5BIT-CONDITION FLAGS AUSDRUCKEN
0F81 47         2953        MOV     B,A
0F82 2100D5     2954        LXI     H,0D500H        ;MASKE MIT BITPOSITION DES FLAG-REGISTERS
0F85 78         2955 PREG2A: MOV    A,B
0F86 29         2956        DAD     H
0F87 47         2957        MOV     B,A
0F88 D2920F     2958        JNC     PREG2B
0F8B 07         2959        RLC
0F8C 7D         2960        MOV     A,L
0F8D CE30       2961        ACI     '0'
0F8F CDB60B     2962        CALL    WCHAR
0F92 78         2963 PREG2B: MOV    A,B
0F93 87         2964        ADD     A
0F94 47         2965        MOV     B,A
0F95 7C         2966        MOV     A,H
0F96 B7         2967        ORA     A
0F97 C8         2968        RZ
0F98 C3850F     2969        JMP     PREG2A
                2970
0F9B 1A         2971 PREG3: LDAX    D               ;16BIT-REGISTER AUSDRUCKEN
0F9C 6F         2972        MOV     L,A
0F9D 13         2973        INX     D
0F9E 1A         2974        LDAX    D
0F9F 67         2975        MOV     H,A
0FA0 CD1C0C     2976        CALL    WHLHEX
0FA3 C9         2977        RET
                2978
0FA4 CD010C     2979 PREG4: CALL    WCRLF           ;CR+LF+SPACE + 16BIT-REGISTER AUSDRUCKEN
0FA7 CD8E0B     2980        CALL    WBLANK
0FAA CD9B0F     2981        CALL    PREG3
0FAD C9         2982        RET
                2983
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                  2984 ;*************************************************************************
                  2985 ;* PREGT -      REGISTERUEBERSCHRIFT AUSDRUCKEN
                  2986 ;*       CALL    PREGT
                  2987 ;*************************************************************************
OFAE CDB00B       2988 PREGT:  CALL     WBUFI
OFB1 0D           2989         DB       CR,LF
OFB2 0A
OFB3 20504320     2990         DB       ' PC   LABEL:  OP   ADR.FELD  '
OFB7 20204C41
OFBB 42454C3A
OFBF 20204F50
OFC3 20202041
OFC7 44522E46
OFCB 454C4420
OFCF 20
OFD0 4120204E     2991         DB       'A  NZHPC B  C  D  E  H  L  I  SP',0
OFD4 5A485043
OFD8 20422020
OFDC 43202044
OFE0 20204520
OFE4 20482020
OFE8 4C202049
OFEC 20205350
OFF0 00
OFF1 C9           2992         RET
                  2993
                  2994 ;*************************************************************************
                  2995 ;* REGRES -      REGISTERINHALTE WIEDERHERSTELLEN
                  2996 ;*       CALL    REGRES
                  2997 ;*************************************************************************
OFF2 3AD8FC       2998 REGRES: LDA      IMWERT          ;INTERRUPT-MASKE
OFF5 E60F         2999         ANI      0FH
OFF7 F608         3000         ORI      08H
OFF9 30           3001         SIM
OFFA D1           3002         POP      D               ;(DE) = (PC) DES RUFENDEN PROGRAMMES
OFFB 2AD9FC       3003         LHLD     SPWERT          ;STACK-ZEIGER
OFFE F9           3004         SPHL
OFFF 2AD7FC       3005         LHLD     PCWERT          ;PROGRAMMZAEHLER DES UNTERBROCHENEN PROGRAMMES
1002 E5           3006         PUSH     H               ;  AUF STACK; RET  IM RUFENDEN PROGRAMM
                  3007                                  ;  BEWIRKT RUECKKEHR ZUM UNTERBROCHENEN PROGR.
1003 D5           3008         PUSH     D               ;(PC) DES RUFENDEN PROGRAMMES AUF STACK
1004 2AD1FC       3009         LHLD     BCWERT          ;(BC)
1007 44           3010         MOV      B,H
1008 4D           3011         MOV      C,L
1009 2AD3FC       3012         LHLD     DEWERT          ;(DE)
100C EB           3013         XCHG
100D 2ACFFC       3014         LHLD     PSWERT          ;(A) UND CONDITION FLAGS
1010 E5           3015         PUSH     H
1011 F1           3016         POP      PSW
1012 2AD5FC       3017         LHLD     HLWERT          ;(HL)
1015 C9           3018         RET
                  3019
```

```
LOC  OBJ          LINE        SOURCE STATEMENT
                  3020 ;*****************************************************************************
                  3021 ;* REGSAV -      ALLE REGISTERINHALTE SICHERSTELLEN
                  3022 ;*       CALL    REGSAV
                  3023 ;*               (HL) = PROGRAMMZAEHLER
                  3024 ;*****************************************************************************
1016 22D5FC       3025 REGSAV: SHLD     HLWERT          ;(HL)
1019 F5           3026         PUSH     PSW
101A E1           3027         POP      H
101B 22CFFC       3028         SHLD     PSWERT          ;(A) + CONDITION FLAGS
101E EB           3029         XCHG
101F 22D3FC       3030         SHLD     DEWERT          ;(DE)
1022 60           3031         MOV      H,B
1023 69           3032         MOV      L,C
1024 22D1FC       3033         SHLD     BCWERT          ;(BC)
1027 210400       3034         LXI      H,4
102A 39           3035         DAD      SP
102B 22D9FC       3036         SHLD     SPWERT          ;STACK-ZEIGER
102E D1           3037         POP      D
102F E1           3038         POP      H
1030 22D7FC       3039         SHLD     PCWERT          ;PROGRAMMZAEHLER
1033 20           3040         RIM
1034 32DBFC       3041         STA      IMWERT          ;INTERRUPT-MASKE
1037 D5           3042         PUSH     D
1038 C9           3043         RET
                  3044
                  3045 ;*****************************************************************************
                  3046 ;* SUB2 -        ZWEI 16-BIT WERTE SUBTRAHIEREN
                  3047 ;*               (HL) = (HL) - (DE)
                  3048 ;*****************************************************************************
1039 7D           3049 SUB2:   MOV      A,L
103A 93           3050         SUB      E
103B 6F           3051         MOV      L,A
103C 7C           3052         MOV      A,H
103D 9A           3053         SBB      D
103E 67           3054         MOV      H,A
103F C9           3055         RET
                  3056
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT
                   3057 ;*************************************************************************
                   3058 ;* ZUASC -       AUF KONTROLLZEICHEN PRUEFEN
                   3059 ;*       MVI     A,ASCII-ZEICHEN
                   3060 ;*       CALL    ZUASC
                   3061 ;*       JC      KONTROLLZEICHEN
                   3062 ;*************************************************************************
 1040 FE20         3063 ZUASC:   CPI     20H             ;KONTROLLZEICHEN ?
 1042 C9           3064          RET                     ;C: JA
                   3065
                   3066 ; ZUASC1
 1043 57           3067 ZUASC1:  MOV     D,A
 1044 C9           3068          RET
                   3069
                   3070 ;*************************************************************************
                   3071 ;* ZUBIN -       ASCII-ZEICHEN BINAER WANDELN UND PRUEFEN
                   3072 ;*       MVI     A,ASCII-ZEICHEN
                   3073 ;*       CALL    ZUBIN
                   3074 ;*       JC      NICHTBINAER
                   3075 ;*               (A) = BINAERWERT
                   3076 ;*************************************************************************
 1045 D630         3077 ZUBIN:   SUI     '0'             ;ASCII-0=30HEX ABZIEHEN
 1047 D8           3078          RC                      ;C: NICHT BIN
 1048 FE02         3079          CPI     2
 104A 3F           3080          CMC
 104B C9           3081          RET                     ;C: NICHT BIN
                   3082
                   3083 ;*************************************************************************
                   3084 ;* ZUBIN8 -      ZEICHEN FUER ZEICHEN BINAER WANDELN
                   3085 ;*               (D) = (D)*2 + (A)-GEWANDELT
                   3086 ;*       MVI     A,ASCII-ZEICHEN
                   3087 ;*       CALL    ZUBIN8
                   3088 ;*       JC      NICHT BINAER
                   3089 ;*               (D) = BINAERRESULTAT
                   3090 ;*               (A) = BINAERER WERT DES ZEICHENS
                   3091 ;*************************************************************************
 104C CD4510       3092 ZUBIN8:  CALL    ZUBIN
 104F D8           3093          RC
 1050 E5           3094          PUSH    H
 1051 6A           3095          MOV     L,D
 1052 29           3096          DAD     H
 1053 B5           3097          ORA     L
 1054 57           3098          MOV     D,A
 1055 E1           3099          POP     H
 1056 C9           3100          RET
                   3101
```

```
LOC  OBJ        LINE        SOURCE STATEMENT
                3102 ;*************************************************************************
                3103 ;* ZUDEZ -      ASCII-ZEICHEN DEZIMAL WANDELN UND PRUEFEN
                3104 ;*      MVI     A,ASCII-ZEICHEN
                3105 ;*      CALL    ZUDEZ
                3106 ;*      JC      NICHTDEZIMAL
                3107 ;*              (A) = DEZIMALWERT
                3108 ;*************************************************************************
1057 D630       3109 ZUDEZ:  SUI     '0'             ;ASCII-0=30HEX ABZIEHEN
1059 D8         3110         RC                      ;C: NICHT DEZ
105A FE0A       3111         CPI     10
105C 3F         3112         CMC
105D C9         3113         RET                     ;C: NICHT DEZ
                3114
                3115 ;*************************************************************************
                3116 ;* ZUDEZ3 -     DEZIMAL PRUEFEN, WANDELN UND 8-BIT RESULTAT
                3117 ;*              (D) = (D)*10 + (A)-GEWANDELT
                3118 ;*      MVI     A,ASCII-ZEICHEN
                3119 ;*      CALL    ZUDEZ3
                3120 ;*      JC      NICHTDEZ ODER RESULTAT GROESSER ALS 255 (8 BIT)
                3121 ;*************************************************************************
105E CD5710     3122 ZUDEZ3: CALL    ZUDEZ
1061 D8         3123         RC                      ;C: ZEICHEN NICHT DEZ
1062 5F         3124         MOV     E,A
1063 7A         3125         MOV     A,D
1064 87         3126         ADD     A               ;*2
1065 D8         3127         RC
1066 87         3128         ADD     A               ;*4
1067 D8         3129         RC
1068 82         3130         ADD     D               ;*5
1069 D8         3131         RC
106A 87         3132         ADD     A               ;*10
106B D8         3133         RC
106C 83         3134         ADD     E               ;(ALTER WERT)*10 + (NEUER WERT)
106D D8         3135         RC
106E 57         3136         MOV     D,A
106F 7B         3137         MOV     A,E
1070 C9         3138         RET
                3139 ;*************************************************************************
                3140 ;* ZUFOR -      ENTSPRECHEND FORMAT EINLESEN UND WANDELN
                3141 ;*      MVI     C,FORMATINDEX
                3142 ;*      MVI     A,ASCII-ZEICHEN
                3143 ;*      CALL    ZUFOR
                3144 ;*************************************************************************
1071 E5         3145 ZUFOR:  PUSH    H               ;REGISTERINHALTE AUF STACK SICHERN
1072 C5         3146         PUSH    B
1073 217C10     3147         LXI     H,ZUFTAB        ;(C) = FORMATINDEX
1076 CD750E     3148         CALL    CALLTB          ;      0=ASCII, 1=BIN, 2=DEZ, 3=HEX
1079 C1         3149         POP     B               ;REGISTER WIEDERHERSTELLEN
107A E1         3150         POP     H
107B C9         3151         RET
107C 4310       3152 ZUFTAB: DW      ZUASC1          ;1 STELLE ASCII -) 1 BYTE
107E 4C10       3153         DW      ZUBIN8          ;1 STELLE BIN.  -) 1 BYTE
1080 5E10       3154         DW      ZUDEZ3          ;1 STELLE DEZ.  -) 1 BYTE
1082 9610       3155         DW      ZUHEX2          ;1 STELLE HEX.  -) 1 BYTE
                3156
```

```
LOC  OBJ        LINE        SOURCE STATEMENT
                 3157
                 3158 ;*****************************************************************************
                 3159 ;* ZUHEX -      ASCII-ZEICHEN HEXADEZ. WANDELN UND PRUEFEN
                 3160 ;*       MVI    A,ASCII-ZEICHEN
                 3161 ;*       CALL   ZUHEX
                 3162 ;*       JC     NICHTHEX
                 3163 ;*       JNC    HEX
                 3164 ;*              (A) = HEX-WERT
                 3165 ;*****************************************************************************
1084 D630        3166 ZUHEX:  SUI    '0'
1086 D8          3167         RC                          ;0-2F FALSCH
1087 C6E9        3168         ADI    '0'-'G'
1089 D8          3169         RC                          ;47-FF FALSCH
108A C606        3170         ADI    6
108C F29210      3171         JP     $+6                  ;A-F
108F C607        3172         ADI    7
1091 D8          3173         RC                          ;3A-40 FALSCH
1092 C60A        3174         ADI    10
1094 B7          3175         ORA    A
1095 C9          3176         RET
                 3177
                 3178 ;*****************************************************************************
                 3179 ;*       MVI    A,ASCII-ZEICHEN
                 3180 ;*       CALL   ZUHEX2
                 3181 ;*       JC     ZEICHEN NICHT HEX.
                 3182 ;*              (D) = (D)*16 + (A)-GEWANDELT
                 3183 ;*              (A) = HEX.WERT DES ZEICHENS
                 3184 ;*****************************************************************************
1096 CD8410      3185 ZUHEX2: CALL   ZUHEX
1099 D8          3186         RC
109A E5          3187         PUSH   H
109B 6A          3188         MOV    L,D
109C 29          3189         DAD    H
109D 29          3190         DAD    H
109E 29          3191         DAD    H
109F 29          3192         DAD    H
10A0 B5          3193         ORA    L
10A1 57          3194         MOV    D,A
10A2 E1          3195         POP    H
10A3 C9          3196         RET
                 3197
```

10. RAM-RESERVIERUNGEN

```
LOC  OBJ          LINE        SOURCE STATEMENT
                  3198 ;
                  3199 ;*************************************************************************
                  3200 ;*       RAM - RESERVIERUNGEN
                  3201 ;*************************************************************************
                  3202 ;
FD90              3203          LBRAM   EQU     0FD90H  ;RAM FUER LABELTABELLE
FCE1              3204          BUFFER  EQU     0FCE1H  ; ASCII-DATENPUFFER ("BREAD" ETC.)
FCF2              3205          BUFEND  EQU     0FCF2H
FC00              3206          RAM     EQU     0FC00H  ;BASISADRESSE RAM-HILFSSPEICHER
                  3207
                  3208
FC00              3209          ORG RAM
                  3210
                  3211
FC00              3212          DS      50              ;ANWENDER STACK
                  3213 USRSTK:
FC32              3214          DS      128-50          ;MONITOR STACK
                  3215 MONSTK:
                  3216 ; EINSPRUNG-ADRESSEN IN I/O-TREIBER-ROUTINEN
FC80              3217 SERIN:   DS      3               ;SERIELLER INPUT
FC83              3218 SEROUT:  DS      3               ;SERIELLER OUTPUT
FC86              3219 CASIN:   DS      3               ;CASSETTEN INPUT
FC89              3220 CASOUT:  DS      3               ;CASSETTEN OUTPUT
                  3221
                  3222 ; RESTART-EINSPRUNGADRESSEN IN RAM
FC8C              3223 RST2:    DS      3
FC8F              3224 RST3:    DS      3
FC92              3225 RST5:    DS      3
FC95              3226 RST55:   DS      3               ;RST 5.5
FC98              3227 RST6:    DS      3
FC9B              3228 RST65:   DS      3               ;RST 6.5
FC9E              3229 RST75:   DS      3               ;RST 7.5
                  3230
                  3231 BRAM:                            ;BEGINN RAM, DAS MIT RESET GELOESCHT WIRD
                  3232
                  3233 ; DEFAULT-WERTE (NULL GESETZT NACH KALTSTART)
                  3234
FCA1              3235 FORMAT:  DS      1               ;ZAHLENFORMAT (ASCII, BIN, DEZ, HEX)
                  3236 BPTADR:                          ;BREAKPOINT-ADRESSEN UND OPCODES
FCA2              3237          DS      BPTANZ*3        ;REIHENFOLGE: ADR-OP-ADR-OP-...ADR-OP
                  3238 ; DEFAULT PARAMETER
FCAE              3239 MBADR:   DS      2               ;'M'-START-ADRESSE
FCB0              3240 PBADR:   DS      2               ;'P'-START-ADRESSE
FCB2              3241 PEADR:   DS      2               ;'P'-STOP -ADRESSE
FCB4              3242 ABADR:   DS      2               ;'A'-START-ADRESSE
FCB6              3243 DBADR:   DS      2               ;'D'-START-ADRESSE
FCB8              3244 DEADR:   DS      2               ;'D'-STOP -ADRESSE
FCBA              3245 TBADR:   DS      2               ;'T'-START-ADRESSE
FCBC              3246 TEADR:   DS      2               ;'T'-STOP -ADRESSE
FCBE              3247 LBADR:   DS      2               ;'L'-START-ADRESSE
FCC0              3248 SBADR:   DS      2               ;'S'-START-ADRESSE
FCC2              3249 SEADR:   DS      2               ;'S'-STOP -ADRESSE
                  3250
                  3251 ; FLAGS
FCC4              3252 TRCFLG:  DS      1               ;=0: KEIN TRACE
```

```
 LOC  OBJ          LINE        SOURCE STATEMENT

                   3253                                ;=1: TRACE
 FCC5              3254 BPTACT: DS      1             ;=1: AKTIV
                   3255                                ;=0: NICHT AKTIV
 FCC6              3256 BPTFLG: DS      1             ;=0: KEINE BREAKPOINTS
                   3257                                ;=1: BREAKPOINTS
 FCC7              3258 BCKFLG: DS      1             ;=0: CR UND SPACE ERLAUBT
                   3259                                ;=1: ZUSAETZLICH + UND - ERLAUBT
 FCC8              3260 CRTFLG: DS      1             ;=0: CRT
                   3261                                ;=1: TTY
 FCC9              3262 GROFLG: DS      1             ;=0: NUR GROSSBUCHSTABEN
                   3263                                ;=1: AUCH KLEINBUCHSTABEN
 FCCA              3264 RUBFLG: DS      1             ;=0: KEINE WIRKUNG
                   3265                                ;=1: KEIN RUB OUT BEI TTY
 FCCB              3266 STEPS:  DS      1             ;ANZAHL EINZELSCHRITTE
                   3267
                   3268 ; ZWISCHENWERTE FUER KOMMANDOS
 FCCC              3269 LINES:  DS      1             ;ZEILENZAEHLER (PRINT TRACE, DISASM, NEXT ETC.)
 FCCD              3270 DATA:   DS      1             ;PORT DATEN
 FCCE              3271 PORT:   DS      1             ;PORT ADRESSE
                   3272
                   3273 ; REGISTERWERTE (ZUM RETTEN UND RESTAURIEREN)
                   3274 REGW:
 FCCF              3275 PSWERT: DS      2             ;A-PSW
 FCD1              3276 BCWERT: DS      2             ;B-C
 FCD3              3277 DEWERT: DS      2             ;D-E
 FCD5              3278 HLWERT: DS      2             ;H-L
 FCD7              3279 PCWERT: DS      2             ;PROGRAMMZAEHLER
 FCD9              3280 SPWERT: DS      2             ;ANWENDER STACK ZEIGER
 FCDB              3281 IMWERT: DS      1             ;INTERRUPT MASKE
 FCDC              3282 SPMON:  DS      2             ;MONITOR STACK ADR.
                   3283
                   3284 ERAM:                         ;RAM - ENDADR.+1 (FUER INITIALISIERUNG)
                   3285
 FCDE              3286 RESBUF: DS      1             ;RESET-TESTBYTE
 FCDF              3287 STIME:  DS      2
                   3288 ;
                   3289 ;*************************************************************************
                   3290 ;*                                                                       *
                   3291 ;*                              E N D E                                  *
                   3292 ;*                                                                       *
                   3293 ;*************************************************************************
                   3294 ;
                   3295
```

USER SYMBOLS

ABADR	FCB4	ABORT	024D	AREG	0001	ASSEMB	02A2	B0	0316	B1	033B	B300	03E8
BCKFLG	FCC7	BCLR	0CAA	BCLR1	0CB3	BCWERT	FCD1	BEL	0007	BFZ	0000	BGET	0CC0
BGETF	0CCD	BGETL	0CDB	BPTACT	FCC5	BPTADR	FCA2	BPTANZ	0004	BPTFLG	FCC6	BPTIN1	0DDC
BPTIN2	0DF0	BPTIN3	0DF9	BPTIN4	0DFE	BPTINS	0DC5	BPTNUM	0E09	BPTRE1	0E1C	BPTRE2	0E2D
BPTRE3	0E32	BPTREM	0E0C	BPTSE0	0E46	BPTSE1	0E4B	BPTSE2	0E5D	BPTSE3	0E6C	BPTSE4	0E6E
BPTSET	0E37	BPUT	0CE9	BPUT1	0CEF	BPUT2	0D02	BPUT3	0D06	BRAM	FCA1	BREA22	0D37
BREA51	0D63	BREA52	0D74	BREA53	0D77	BREA71	0D9C	BREAD	0D0A	BREAD0	0D0F	BREAD1	0D26
BREAD2	0D2B	BREAD3	0D3B	BREAD4	0D40	BREAD5	0D46	BREAD6	0D81	BREAD7	0D96	BREAD8	0DAE
BREAD9	0DC1	BREAK	02DF	BREAK1	02E0	BREAKP	02CB	BREG	0003	BS	0008	BUFEND	FCF2
BUFFER	FCE1	BUFLG	0010	CALLHL	0E74	CALLTB	0E75	CASI	07EF	CASIN	FC86	CASINI	07F9
CAS0	0821	CAS01	0822	CASOUT	FC89	CHTAB	0EA0	CHTST	0E84	CMD	01AC	CMD1	01C7
CMDAB0	0225	CMDEX1	01EB	CMDEX2	01F0	CMDEX3	0205	CMDEX4	020A	CMDEX5	0215	CMDTAB	00B6
CMDUSE	023F	CMP2	0EA8	CMPL	0EB0	CMPL1	0EB4	CMPL2	0EBF	CMPL3	0EC6	CMPLI	0ECC
CR	000D	CREG	0002	CRTFLG	FCC8	CRTTST	0ED2	DATA	FCCD	DBADR	FCB6	DEADR	FCB8
DEWERT	FCD3	DISA1	19F9	DISASM	19AE	DISASS	034A	DISLIN	0EDD	DREG	0005	ERAM	FCDE
EREG	0004	ESC	001B	EXPORT	0532	FBIN	0001	FHEX	0003	FORMAT	FCA1	FREG	0000
G0	0374	G01	0389	G02	0394	G03	03AE	GROFLG	FCC9	GROSS	0EE9	HADR	08DF
HADR1	08F3	HADR2	0905	HBPTD	0908	HBPTD1	0914	HDATA	0941	HDATA1	0959	HDATA2	096A
HDATAD	096E	HEINO	097E	HEIN1	0997	HEIN2	09C3	HEINAU	097B	HELP	03B5	HELP1	03B8
HELP2	03C0	HELP3	03CC	HEXASC	0EF2	HFORD0	09C8	HFORD1	09F1	HFORD2	0A08	HFORMD	09C5
HJANEI	0A0C	HLWERT	FCD5	HPORD1	0A5B	HPORTD	0A2C	HREG	0007	HSTARD	0A74	HSTART	0A6D
HSTEPD	0A7B	HSTOPD	0A9C	I1	03D6	I2	03D9	IMWERT	FCDB	INKLB	127D	INPORT	03D1
IREG	000C	L1	042F	L11	0451	L2	0458	L3	046A	L4	046C	L5	0476
L6	048C	LBADR	FCBE	LBANF	0004	LBINIT	10B8	LBRAM	FD90	LCLR	08C6	LEER	FFFF
LF	000A	LINES	FCCC	LINIT	08CD	LOAD	03F4	LREG	0006	LSPACE	041B	LTST	08D5
M0	04AA	M1	04BE	M2	04DE	M3	04EA	M4	04EB	M5	04FF	MBADR	FCAE
MEMORI	0499	MONSTK	FC80	NEXTIN	0501	NLINE	000F	O1	051A	O2	051D	OUTPOR	0515
P1	0569	P2	0572	PBADR	FCB0	PCWERT	FCD7	PDATA	0AA3	PEADR	FCB2	PORT	FCCE
PREG	0008	PREG1	0F7B	PREG2	0F80	PREG2A	0F85	PREG2B	0F92	PREG3	0F9B	PREG4	0FA4
PREGF	0EFB	PREGP	0F0A	PREGS	0F12	PREGS0	0F1B	PREGS1	0F3D	PREGS2	0F49	PREGT	0FAE
PREGTB	0F67	PRINT	0541	PSTARD	0AC5	PSTART	0AB4	PSTOPD	0ACC	PSWERT	FCCF	R2HEX	05C2
R4HEX	061A	R4HEX0	062D	R4HEX1	0637	R4HEX2	0643	R4HEX3	064C	R4HEX4	0655	R4HEX5	065B
R4HEXP	0666	R5CC	05D0	R5CC1	05E6	R5CC2	05F4	R5CC3	05FB	R5CC4	0614	RAM	FC00
RCAS	0C2E	RCASA1	0C58	RCASAH	0C34	RCHAR	0AE0	REG1	0594	REGIST	0584	REGRES	0FF2
REGSAV	1016	REGTAB	0673	REGTB	0F6F	REGW	FCCF	RESBUF	FCDE	RESET	0149	RESET2	0175
RST2	FC8C	RST3	FC8F	RST5	FC92	RST55	FC95	RST6	FC98	RST65	FC9B	RST75	FC9E
RTAB	05BA	RUBFLG	FCCA	RUBOUT	007F	S1	06D3	S11	06D5	S2	06E8	SAVE	067F
SBADR	FCC0	SEADR	FCC2	SERI	082D	SERI1	0841	SERI2	0846	SERIN	FC80	SERINI	0860
SERIT1	0869	SERIT2	086E	SERIT3	0888	SERIT4	0898	SERO	089F	SERO1	08AB	SERO2	08B3
SEROUT	FC83	SPMON	FCDC	SPWERT	FCD9	SREG	000A	SSPACE	06BA	STEPS	FCCB	STIME	FCDF
SUB2	1039	TASTER	0265	TBADR	FCBA	TEADR	FCBC	TRACE	10F6	TRACE1	0716	TRAP	0286
TRCFLG	FCC4	TRHALT	07CA	TRINIT	10E4	TRINT	0756	TRINT1	076A	TRINT3	0771	TRINT4	079D
TRSTEP	0732	TRSTP1	0735	TRSTP2	073B	UARTD	00FE	UARTST	00FF	USRSTK	FC32	WAASC	0AEE
WAASC1	0B00	WABIN	0B04	WABIN1	0B0A	WADEZ	0B19	WADEZ1	0B31	WADEZ2	0B33	WADEZ3	0B48

WAFOR	0B4F	WAFORB	0B68	WAFTAB	0B60	WAHEX	0B6F	WAHEXB	0B82	WBELL	0BB9	WBLANK	0B0E
WBLNK1	0B96	WBLNKI	0B93	WBUF	0BA1	WBUF1	0BA2	WBUF2	0BAE	WBUFI	0BB0	WCAS	0C76
WCASAH	0C7C	WCASB1	0C93	WCASBU	0C91	WCASHL	0C9F	WCHAR	0BB6	WCHAR2	0BE4	WCHAR3	0BF4
WCHARI	0BF7	WCRLF	0C01	WCRLFI	0C13	WHLHEX	0C1C	WHLHXB	0C27	ZUASC	1040	ZUASC1	1043
ZUBIN	1045	ZUBINB	104C	ZUDEZ	1057	ZUDEZ3	105E	ZUFOR	1071	ZUFTAB	107C	ZUHEX	1084
ZUHEX2	1096												

ASSEMBLY COMPLETE, NO ERRORS

Wegweiser durch das MFA-Mediensystem

Ein Wegweiser durch das MFA-

Einsatzbereich

Einführung in die Hardware von Mikrocomputern auf unterschiedlichen Vermittlungsebenen.

Vermittlungsschwerpunkte

Zusammenwirken der Funktionseinheiten eines Mikrocomputers auf einer Vermittlungsebene, die weitgehend von der Schaltungstechnik losgelöst ist (Basiswissen).	Aufbau und Zusammenwirken der Funktionseinheiten eines Mikrocomputers auf einer Vermittlungsebene, bei der die Darstellungen der Schaltungen auf die wesentlichen Bauelemente reduziert sind.	Aufbau und Zusammenwirken der Funktionseinheiten eines Mikrocomputers und Wirkungsweise spezieller Bausteine auf einer Vermittlungsebene, bei der die konkrete Schaltungstechnik verwendet wird.

Erforderliche Baugruppen

- Grundgerät [1]
  (ASCII-Tastatur, Video-Interface und Monitor sind hierzu nicht unbedingt erforderlich)
- Adapterkarte 64polig

Erforderliche Arbeitsmittel

Aus FTÜ Band 1[2] jeweils Theorie- und Übungsteil 1 von: - Aufbau von DV-Anlagen und Bus-Systemen - Ein- und Ausgabe-Einheiten - Speicher-Einheiten - Mikroprozessor-Mikrocomputer	Aus FTÜ Band 1[2] jeweils Theorie- und Übungsteile 1 und 2 von: - Aufbau von DV-Anlagen und Bus-Systemen - Ein- und Ausgabe-Einheiten - Speicher-Einheiten - Mikroprozessor-Mikrocomputer	FPÜ Band 1[3] und aus FTÜ Band 1[2] jeweils Theorie- und Übungsteile 1 und 2 von: -Aufbau von DV-Anlagen u.Bus-Syst. -Ein- und Ausgabe-Einheiten -Speicher-Einheiten -Mikroprozessor-Mikrocomputer

Zielgruppen (Beispiele)

-Personen aus allen gewerblich-technischen und naturwissenschaftlichen Berufsfeldern - Ausbildungsberater - Berufsberater, Arbeitsvermittler - Technische Berater - Führungskräfte	- Informations-, Energie- und Funkelektroniker in Erstausbildung und Umschulung - Elektro-Facharbeiter in der Weiterbildung - Facharbeiter aus der Meß- und Regelungstechnik	- Informations-, Energie- und Funkelektroniker in Erstausb. u. Umsch. - Elektro-Facharbeiter in der Weiterbildung - Meister, Techniker und Ingenieure in der Weiterbildung - Ausbilder und Lehrer in der Weiterbildung

---

[1]) Zum Grundgerät gehören die Baugruppen:
- Baugruppenträger mit Bus-Verdrahtung
- Bus-Abschluß
- Trafo-Einschub
- Spannungsregelung
- Prozessor 8085
- 8-K-RAM/EPROM bestückt mit 2-K-RAM
- 8-K-RAM/ERPOM bestückt mit MAT 85
- 8-Bit-Parallel-Ausgabe
- 8-Bit-Parallel-Eingabe
- Bus-Signalgeber
- Bus-Signalanzeige
- ASCII-Tastatur
- Video-Interface

[2]) FTÜ Band 1 enthält:
- Aufbau von DV-Anlagen und Bus-Systemen
- Ein- und Ausgabe-Einheiten
- Speicher-Einheiten
- Mikroprozessor-Mikrocomputer
- Steuerung einer Paketwendeanlage
- Softwarepaket SP1
- Betriebsprogramm MAT 85

[3]) FPÜ Band 1 enthält alle technischen Unterlagen, die zum Bau und zur Inbetriebnahme der unter [1]) aufgeführten Baugruppen benötigt werden.

[4]) Die Software-Erweiterung SP1 enthält MAT 85+, SPS und Steuer-BASIC (4 EPROMs)

[5]) FPÜ Band 2 wie FPÜ Band 1, jedoch für die Baugruppen:
- 16-K-RAM/EPROM
- Progr. Parallelschnittstelle
- EPROM-Programmierer
- Drucker-Interface
- Zeitwerk (4fach)
- Progr. Serienschnittstelle
- Kassetten-Interface
- Analoge Ein-/Ausgabe (2kanalig)
- Zähler und Zeitgeber
- Fehlersimulation
- Demonstrationsmodell

Mögliche Hardware-Erweiterungen (Arbeitsmittel siehe [5])

Einsatzmerkmale für die Erweiterungen

Mediensystem Mikrocomputer-Technik

Einführung in die Programmierung von Mikrocomputern auf unterschiedlichen Sprachebenen

Programmierung in Maschinen- und Assemblersprache anhand von Beispielen aus Technikbereichen wie ... - *Steuerungstechnik* - *Meß- und Regelungstechnik* - *Nachrichtenübertragungstechnik*	*Programmierung von "Speicherprogrammierbaren Steuerungen" (SPS)*	*Programmierung in BASIC*
- *Grundgerät [1]) und Monitor* - *Adapterkarte 64polig*	- *Grundgerät [1]) und Monitor* - *Adapterkarte 64polig* - *2-K-RAM-IC (z.B. 6116)* - *8-K-EPROM-Baugruppe bestückt mit 8-K-Software-Erweiterung SP1 [4])* - *Zeitwerk (4fach)*	- *Grundgerät [1]) und Monitor* - *Adapterkarte 64polig* - *8-K-RAM-Baugruppe mit 8-K-RAM bestückt* - *8-K-EPROM-Baugruppe bestückt mit 8-K-Software-Erweiterung SP1 [4])*
Aus FTÜ Band 1 [2]): - *Betriebsprogramm MAT 85* - *Steuerung einer Paketwendeanlage*	*Aus FTÜ Band 1 [2]):* - *Betriebsprogramm MAT 85* - *Softwarepaket SP1 (hieraus SPS)* - *Steuerung einer Paketwendeanlage* *Aus FPÜ Band 2 [3]):* - *Zeitwerk (4fach)*	*Aus FTÜ Band 1 [2])* - *Betriebsprogramm MAT 85* - *Softwarepaket SP1 (hieraus BASIC)* - *Steuerung einer Paketwendeanlage*
- *Informations-, Energie- und Funkelektroniker in Erstausbildung und Umschulung* - *Elektro-Facharbeiter in der Weiterbildung* - *Meister, Techniker und Ingenieure in der Weiterbildung* - *Ausbilder u. Lehrer in der Weiterbild.*	- *Energieelektroniker in Erstausbildung und Umschulung* - *Facharbeiter, Meister und Techniker in der Weiterbildung* - *Ausbilder und Lehrer in der Weiterbildung*	- *Informations-, Energie- und Funkelektroniker in Erstausbildung und Umschulung* - *Facharbeiter, Meister, Techniker und Ingenieure in der Weiterbildung* - *Ausbilder und Lehrer in der Weiterbildung*
- *Drucker-Interface* - *Kassetten-Interface* - *EPROMS mit SP1 [4]), hierzu 8-K-RAM-Baugruppe oder 16-K-RAM/EPROM-Baugruppe (für MAT 85 und SP1)* - *Drucker* - *Kassetten-Recorder* - *EPROM-Programmierer* - *Programmierbare Parallelschnittst.* - *Programmierbare Serienschnittst.* - *Analoge Ein-/Ausgabe (2kanalig)*	- *Drucker-Interface* - *Drucker* - *Kassetten-Interface* - *Kassetten-Recorder*	- *Drucker-Interface, Drucker* - *Kassetten-Interface, Kassetten-Recorder* - *Progr. Parallelschnittstelle* - *Progr. Serienschnittstelle* - *Analoge Ein-/Ausgabe (2kanalig)*
- *Zusätzliche Betriebs-Kommandos* - *Drucken von Programmen* - *Speichern von Programmen auf Band* - *Speichern der Programme im EPROM* - *Mehrkanalige Ein- und Ausgabe* - *Serielle Datenübertragung* - *A/D- und D/A-Wandlung*	- *Drucken und Speichern von Übungsprogrammen*	- *Drucken und Speichern von Übungsprogrammen* - *Projektbezogene Interface-Technik*

ANPASSUNGS-FORTBILDUNG

DIGITAL-/MIKROCOMPUTER-TECHNIK

Die stürmische Entwicklung auf den Gebieten der Digital- und Mikrocomputer-Technik in den vergangenen 15 Jahren, ausgelöst durch das Aufkommen der Mikroprozessoren, führt zu ständig zunehmenden Anwendungen dieser Techniken in den verschiedensten Produkten. Dies hat zur Folge, daß insbesondere immer mehr Elektronikfachkräfte mit den neuen Techniken in Berührung kommen. Die betroffenen Fachkräfte lassen sich grob in zwei Gruppen einteilen.

Zielgruppe: Digitale Steuerungstechnik

Die erste Gruppe bilden Fachkräfte, die im Zuge der Automatisierung verstärkt mit modernen Geräten der Steuerungstechnik, z.B. mit "Speicherprogrammierbaren Steuerungen" (SPS) in Berührung kommen. Die schnelle Weiterentwicklung dieser Technik führt zu immer leistungsfähigeren Geräten, die in ständig komplexer werdende Prozesse (Geräte, Maschinen, Anlagen) eingebunden werden. Dies verlangt von den Fachkräften, neben den Grundkenntnissen der Digitaltechnik, neue Kenntnisse und Arbeitsmethoden ...

- zum Aufbau von Steuerungen,
- zur Verfolgung komplexer Funktionsabläufe,
- zur systematischen Fehlersuche,
- zur Funktion und Handhabung moderner Steuerungen
- und deren Programmierung.

Zielgruppe: Mikrocomputer-Technik

Fachkräfte, die konkret mit der Mikrocomputer-Technik sowohl gerätetechnisch als auch programmiertechnisch in Berührung kommen, bilden die zweite Gruppe. Sie benötigen über die Digitaltechnik hinaus Kenntnisse ...

- über den Aufbau und die Wirkungsweise von Mikrocomputern
- und deren Programmierung.

Sofern Ihr Tätigkeitsfeld speziell auf den Mikrocomputer bzw. auf seine Funktionseinheiten ausgerichtet ist, werden Kenntnisse erforderlich ...

- zu den verwendeten Bauelementen,
- zu deren Schaltungstechniken,
- zur Inbetriebnahme und
- zu den Fehlersuchmethoden in MC-Systemen

Aufgrund der vielfältigen Einsatzgebiete eines Mikrocomputers können darüber hinaus spezielle Kenntnisse über die Schnittstellen zwischen Computer und Umfeld erforderlich werden, die man mit den Begriffen ...

- Interface-Technik und
- Anwendungs-Technik

umschreibt.

Für diese beiden Zielgruppen wurde am Berufsförderungszentrum Essen ein Anpassungsfortbildungsprogramm entwickelt. Dieses Fortbildungsprogramm besteht aus den Lehrgängen "Digitale Steuerungstechnik" und "Mikrocomputer-Technik". Innerhalb beider Lehrgänge werden aufeinander aufbauende Kurse angeboten, die den verschiedenen Anforderungen an die betroffenen Fachkräfte gerecht werden sollen.

Lehrgang: Digitale Steuerungstechnik
- Grundfunktionen/Grundschaltungen
- Schaltwerke/Schaltnetze
- Digitale Steuerungstechnik
- Speicherprogrammierbare Steuerungstechnik
- Referentenschulung *)

Lehrgang: Mikrocomputer-Technik
- Grunlagen Hardware
- Inbetriebname/Fehlersuche
- Grundlagen Software
- Intefacetechnik
- Anwendungstechnik
- Referentenschulung *)

Durchführungsformen:

Die Fortbildungsmaßnamen wenden sich einerseits an die Gruppen der arbeitslosen Facharbeiter, Techniker und Meister der elektrotechnischen Ausbildungsberufe bzw. an Fachhoch- und Hochschulabsolventen der ingenieur- und naturwissenschaftlichen Disziplinen, für die mehrmonatige Vollzeitmaßnahmen angeboten werden.

Beschäftigte Fachkräfte haben die Möglichkeit, an Wochenkursen oder an berufsbegleitenden Kursen teilzunehmen.

Soll eine größere Zahl von Mitarbeitern eines Betriebes gleichzeitig an einem betriebsinternen Lehrgang teilnehmen, so können hierfür nach entsprechender Absprache Referenten des BFZ zur Verfügung gestellt werden. In diesem Fall können die Lehrgangsinhalte an die speziellen Belange des Auftraggebers angepaßt werden.

*) Die Kurse Referentenschulung sind für Personen vorgesehen, die in der Aus- und Weiterbildung tätig sind oder tätig werden und für andere Träger die Fortbildungsmaßnahmen mit der BFZ-Lehrgangskonzeption durchführen wollen.

ANPASSUNGS-FORTBIDUNG

NC-TECHNIK

Datenverarbeitung und Mikroelektronik im Fertigungsbereich manifestieren sich im wesentlichen in numerisch gesteuerten Werkzeugmaschinen (NC-Maschinen). Heute werden beim Austausch vorhandener und bei der Installation neuer Werkzeugmaschinen überwiegend numerisch gesteuerte Werkzeugmaschinen eingesetzt. Damit ergeben sich selbstverständlich Auswirkungen auf die im Fertigungsbereich Beschäftigten.

Dies hat zur Folge, daß der Bedarf an qualifizieten Fachkräften in der NC-Technik steigen wird, während der Anteil an Fachkräften ohne Kenntnisse der NC-Technik zurückgeht. Der Großteil der Fachkräfte ist bisher während der Ausbildung auf folgende Punkte der neuen Technologie nicht oder nur unzureichend vorbereitet worden:

- Kenntnisse über CNC-Werkzeugmaschinen wie Aufbau und Wirkungsweise von Maschinenelementen, Meßeinrichtungen, Spannmitteln und Werkzeugsystemen
- Kenntnisse der prinzipiellen Funktion der elektrischen Baugruppen von CNC-Werkzeugmaschinen
- Verstehen der angezeigten Daten der Steuerung und sichere Kenntnis über Bedeutung und Funktion der Steuerungstastatur und der Maschinenschalter
- Programmieren von CNC-Werkstück-Bearbeitungsprogrammen an der CNC-Maschine
- Einrichten einer CNC-Werkzeugmaschine mit automatischem Werkzeugwechsel
- Korrigieren und Optimieren von CNC-Werkstückbearbeitungsprogrammen
- Fertigen von Werkstücken nach selbst- und fremderstellten Programmen auf CNC-Werkzeugmaschinen
- Gewährleistung der Werkstückqualität
- Störfallbeseitigung, Wartung und Instandhaltung von CNC-Werkzeugmaschinen
- Verständnis der Organisation des Fertigungsabstandes bei rechnergestützter Programmierung im on-line- und off-line-Betrieb
- Kenntnis der wirtschaftlichen und fertigungstechnischen Vorteile der CNC-Werkzeugmaschinen sowie deren Nachteile.

Um diesen neuen Anforderungen gerecht zu werden, ist es erforderlich, daß Fachkräfte die Möglichkeit haben, sich auf diesem Gebiet durch kurzfristige Anpassungsmaßnahmen weiter fortzubilden.

Das BFZ Essen führt für diese Zielgruppe im Rahmen der Anpassungsfortbildung Lehrgänge in der "NC-Technik" durch.

Voraussetzungen:

Die Maßnahmen wenden sich an die Gruppe der Facharbeiter, Gesellen, Meister und Techniker sowie Personen ohne Berufsabschluß, jedoch mit 4- bis 6-jähriger Tätigkeit in der Metallbranche.

Ausbildungsinhalte:

Die Anpassungsfortbildung "CNC-Drehen" oder "CNC-Fräsen" besteht aus folgenden Themenkreisen:

1. Grundlagen und Einführung in die NC-Tecnik
   - Anwendung und Entwicklung der NC-Technik
   - Aufbau und Arbeitsweise von NC-Maschinen
   - technologische, mathematische und zeichnerische Grundlagen

2. Allgemeine Fertigkeiten und Kenntnisse der NC-Technik
   - Programmaufbau und Programmvorbereitung
   - Programmerstellung und Programmeingabe

3. CNC-Drehen oder CNC-Fräsen
   - Vorbereitung der Maschine und Bereitstellen der Werkzeuge
   - Testlauf und Programmoptimierung

Ausbildungsziel:

Das Ausbildungsziel der Schulungsmaßname ist es, den Teilnehmern zu einer hohen Arbeitsplatztüchtigkeit zu verhelfen. Der hierzu erforderliche Erwerb von anwendbaren CNC-Fertigkeiten und CN-Kenntnissen wird durch eine konsequent praxisbezogene CNC-Ausbildung erreicht. Dadurch wird der Teilnehmer in die Lage versetzt, die erlernten Fähigkeiten und Kenntnisse in der Bedienung und im Umgang mit den CNC-Werkzeugmaschinen auch auf andere CNC-Werkzeugmaschinen und Steuerungen zu transferieren.

Duchführungsformen:

Die Fortbildungsmaßnahmen werden für arbeitslose Fachkräfte als mehrmonatige Vollzeitmaßnamen durchgeführt.

Beschäftigte Fachkräfte haben die Möglichkeit, an berufsbegleitenden Kursen (Teilzeitform) teilzunehmen.

Für die Gruppe der Ausbilder und Lehrer befindet sich ein Kurs in der Vorbereitung.

AUSBILDER-WEITER-BILDUNG

Das Berufsförderungszentrum Essen führt Kurse zur Lehrer- und Ausbilder-Weiterbildung auf dem Gebiete der Mikrocomputer-Technik durch. Dieses Weiterbildungsprogramm ist im Modellversuch zum

"Einsatz der Mikrocomputer-Technik in der Facharbeiterausbildung (MFA)"

unter Mitwirkung von Ausbildenden entwickelt und erprobt worden. Zielsetzung dieser Lehrgänge ist es, neben der Vermittlung von Fachinhalten, den Ausbildenden Wege und Hilfen zur Vermittlung der MC-Technik aufzuzeigen.

Ausbildungsinhalte:

1. Teil: Aufbau eines Mikrocomputers
- BUS-Systeme
- Hexadezimales Zahlensystem
- Aufbau, Einsatz und Wirkungsweise von BUS-Signalanzeige und BUS-Signalgeber

Aufbau, Einsatz und Wirkungsweise von
- 8-bit-Parallel-Eingabe
- 8-bit-Parallel-Ausgabe
- 8-K-RAM/EPROM

Mikrocomputer-Minimalsystem
- Arbeitsweise eines Mikroprozessors
- Einzelschrittsteuerung
- Befehlsabarbeitung
- Ein-, Zwei- und Drei-Byte-Befehle

Einsatz der Datensichtstation
- Serielle und parallele Datenübertragug
- Einführung und Anwendung der Monitor-Kommandos
- Maschinencode/Mnemonischer Code
- Übungen, Vertiefung

Einführung in die Assembler-Programmierung

2. Teil: Blockschaltbild und Funktion der Baugruppen
- Anhand von Messungen typischer Signalverläufe wird die Funktion der Baugruppen und das Vorgehen bei der Fehlersuche erklärt

Inbetriebnahme CPU/System
- Inbetriebnahmemessungen mit dem Oszilloskop
- Free-Run-Mode
- Hardware-Breakpoint
- Single-Step
- Testprogramme

Assembler-Programmierung
- Anhand von verschiedenen Übungsbeispielen werden die Befehle des 8085, Stack-Operationen, Flags und Unterprogrammbehandlung erarbeitet.
- Einsatz von Tracer und Breakpoints
- Verwendung von Unterprogramen aus dem Betriebssystem

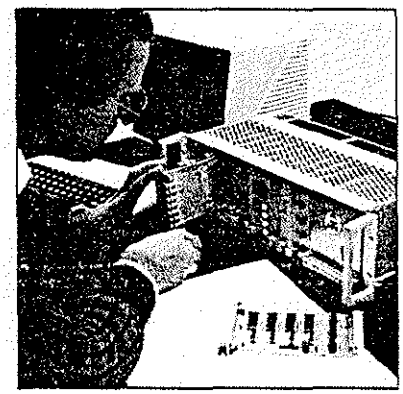
3. Teil: Interface-Technik
- Parallele Ein-/Ausgabetechnik am Beispiel Drucker-Interface
- Serielle Ein-/Ausgabetechnik am Beispiel Kassetten-Interface
- Analoge Ein-/Ausgabetechnik
- Interrupt-Technik

Methodik und Didaktik der Mikrocomputer-Technik
- Qualifikationsebenen
- Struktur der Ausbildungsunterlagen
- Rolle des Ausbilders

Vorstellung und Vorführung von Systemerweiterungen

Durchführungsformen:

Diese Weiterbildungsmaßnahmen werden bundesweit von Referenten des Berufsförderungszentrums Essen in Form von drei zeitversetzten Kurswochen durchgeführt.



„Das MFA-Mediensystem ist ein Lehr- und Lernsystem, mit dem in der Aus- und Weiterbildung praktisches und theoretisches Wissen über Mikrocomputer-Technik vermittelt wird."

Im Zuge fortschreitender Automatisierung erobert der Mikrocomputer immer neue Einsatzbereiche. Beschleunigt wird diese Entwicklung durch den raschen technologischen Fortschritt bei der Integrationstechnik von Halbleitern und durch Kostenminderung und andere Vorteile beim Einsatz von Mikrocomputern in den unterschiedlichsten Sparten von Industrie und Wirtschaft sowie in vielen Bereichen von Wissenschaft, Verwaltung usw.

Durch diese Entwicklung kommen heute mehr und mehr Angehörige der verschiedensten Berufsgruppen mit Geräten und Anlagen in Berührung, die mit Mikrocomputern ausgerüstet sind. Das erfordert in vielen Bereichen eine völlig neue Art der Erstausbildung oder auch eine intensive Weiterbildung.
Zum kompetenten und effektiven Umgang mit Mikrocomputern müssen auf breiter Basis vor allem Kenntnisse vermittelt werden über:

- die Funktionseinheiten eines Mikrocomputers und ihr Zusammenwirken;
- die Inbetriebnahme von Mikrocomputern;
- die Beschreibung und Verfolgung der komplexen Funktionsabläufe in Mikrocomputern;
- die Fehlersuche und -beseitigung an Mikrocomputern und mikrocomputergesteuerten Anlagen.

Das MFA-Mediensystem Mikrocomputer-Technik kann bei der vgs bestellt werden und umfaßt folgende Teile:

– MFA-Mikrocomputer-Baugruppensystem mit Peripheriegeräten
– Fachpraktische Übungen in zwei Bänden
– Fachtheoretische Übungen
– Ausbilder-Handbuch mit Overheadprojektor-Folien

ISBN 3-8025-1233-2