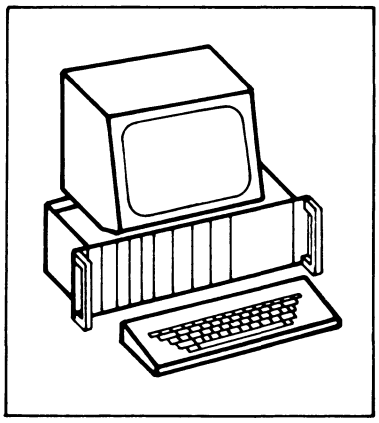
FACHPRAKTISCHE ÜBUNG MIKROCOMPUTER-TECHNIK



Softwarepaket SP 1

BFZ/MFA 7.2.

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

Inhaltsverzeichnis

Inhaltsverzeichnis

Inhaltsverzeichnis

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.

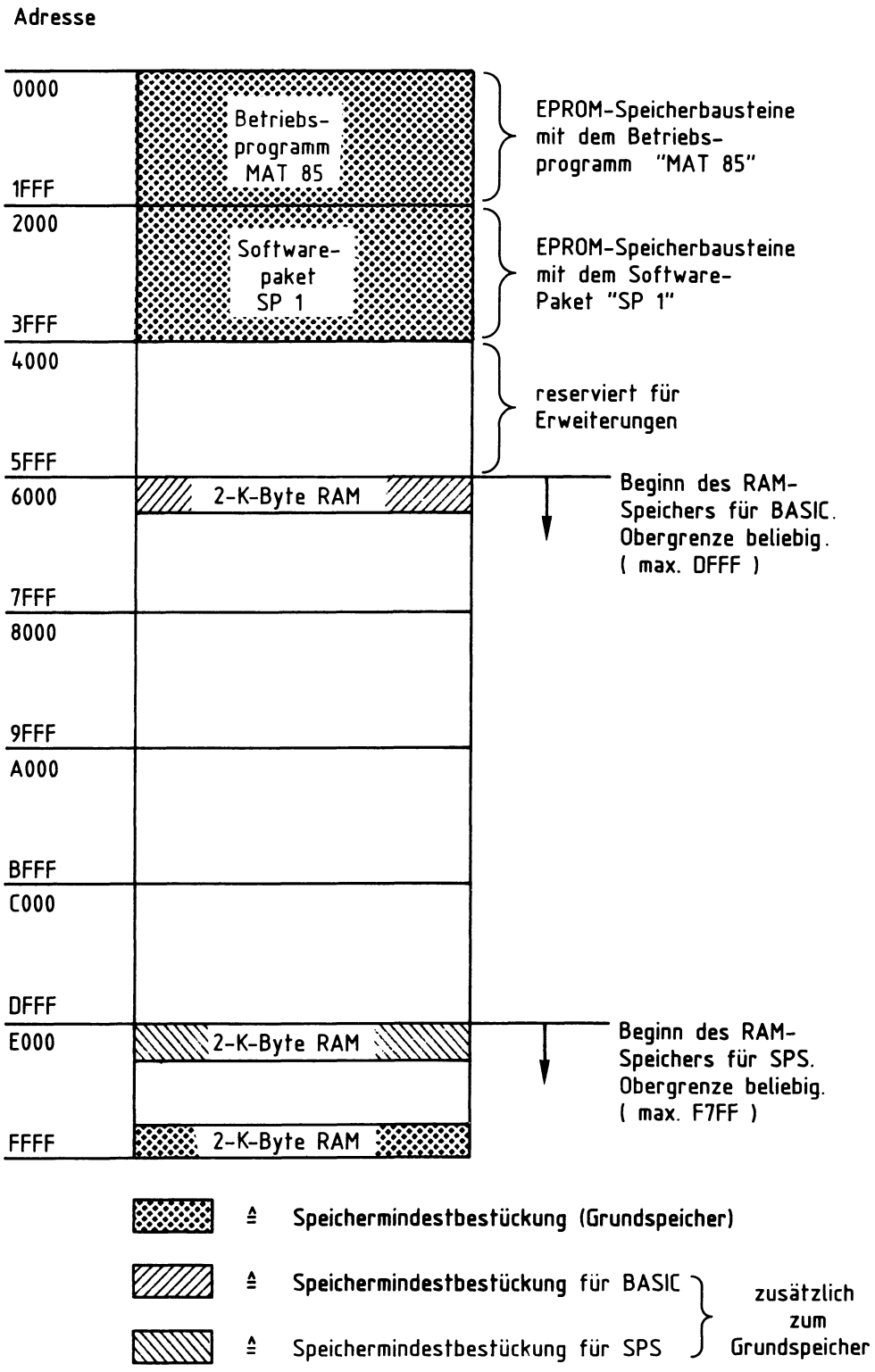


Bild 1: Speicherbelegung

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:

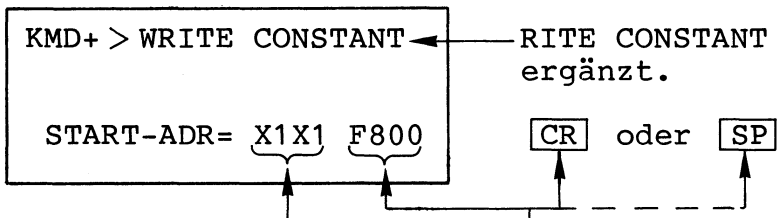
Aufruf des WRITE CONSTANT-Kommandos (wenn KMD+> angezeigt wird):

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:



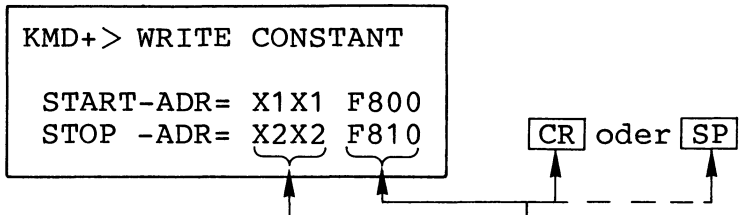
RITE CONSTANT wird vom Programm ergänzt.

[CR] oder [SP]

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.

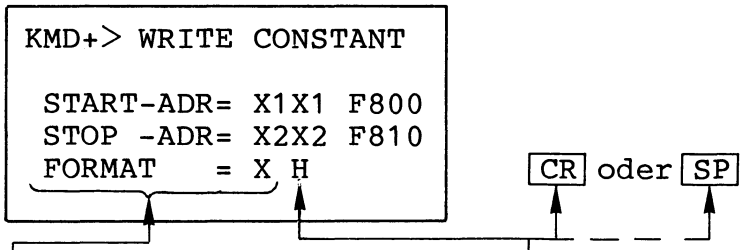
Wirkung:



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!

Arbeitsblatt	BFZ / MFA 7.2. - 16
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.

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.

Softwarepaket SP 1

Name:

MAT 85+ / COPY-Kommando

Datum:

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.

Arbeitsblatt | BFZ / MFA 7.2. - 22

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`	A [CR] oder [SP] (Aufruf nur von MAT 85 aus. ASSEMBLER-Beschreibung siehe FPÜ 7.1.)

`START-ADR = 0000 F800`
`F800 DB 01  START: IN  01`
`F802 C3 00F8  JMP START`
`F805  END`
`*** RESTART ? (JA/NEIN) N`
	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 > _`	[SP] (Auruf von MAT 85+)

`KMD+> INSERT`
`PROGRAMM:`
`START-ADR =AAAA F800`
`STOP -ADR =2000 F804`
`INSRT-ADR =F800 F802`
`BYTE -ANZ =2`
	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.

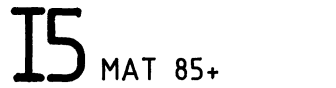
Softwarepaket SP 1

Name:

MAT 85+ / INSERT-Kommando

Datum:

Vollziehen Sie das obige Beispiel nach.



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	R [CR] eintippen, "AM-TEST" 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]

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	V [CR] eintippen, "ERIFY" 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]
VERGL-BER = X3X3 Y3Y3	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.

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.

Arbeitsblatt	BFZ / MFA 7.2. - 40
Softwarepaket SP 1	Name:
MAT 85+ / WRITE-CONSTANT-Kommando	Datum:

Füllen Sie den Speicherbereich von F900 bis F9FF mit

W5 MAT 85+

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).

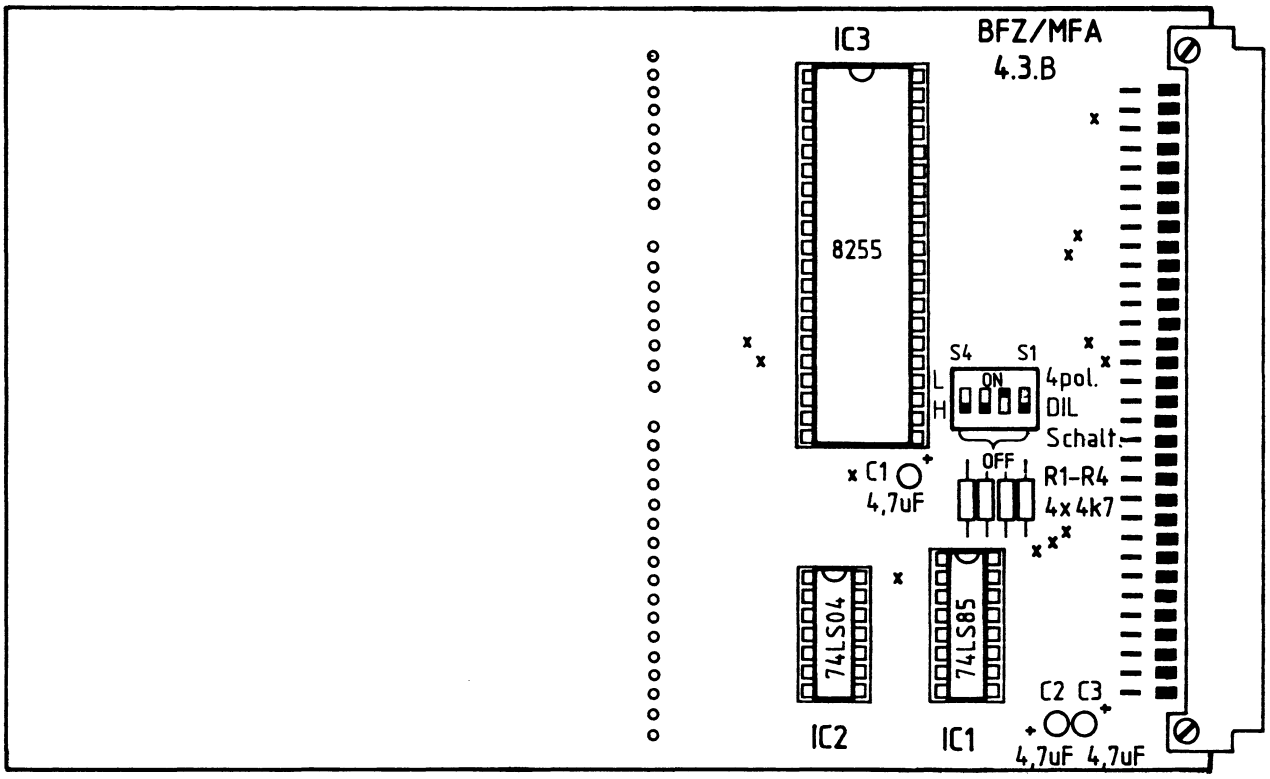


Bild 2: Adreßeinstellung der Baugruppe EPROM-Programmierer

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.

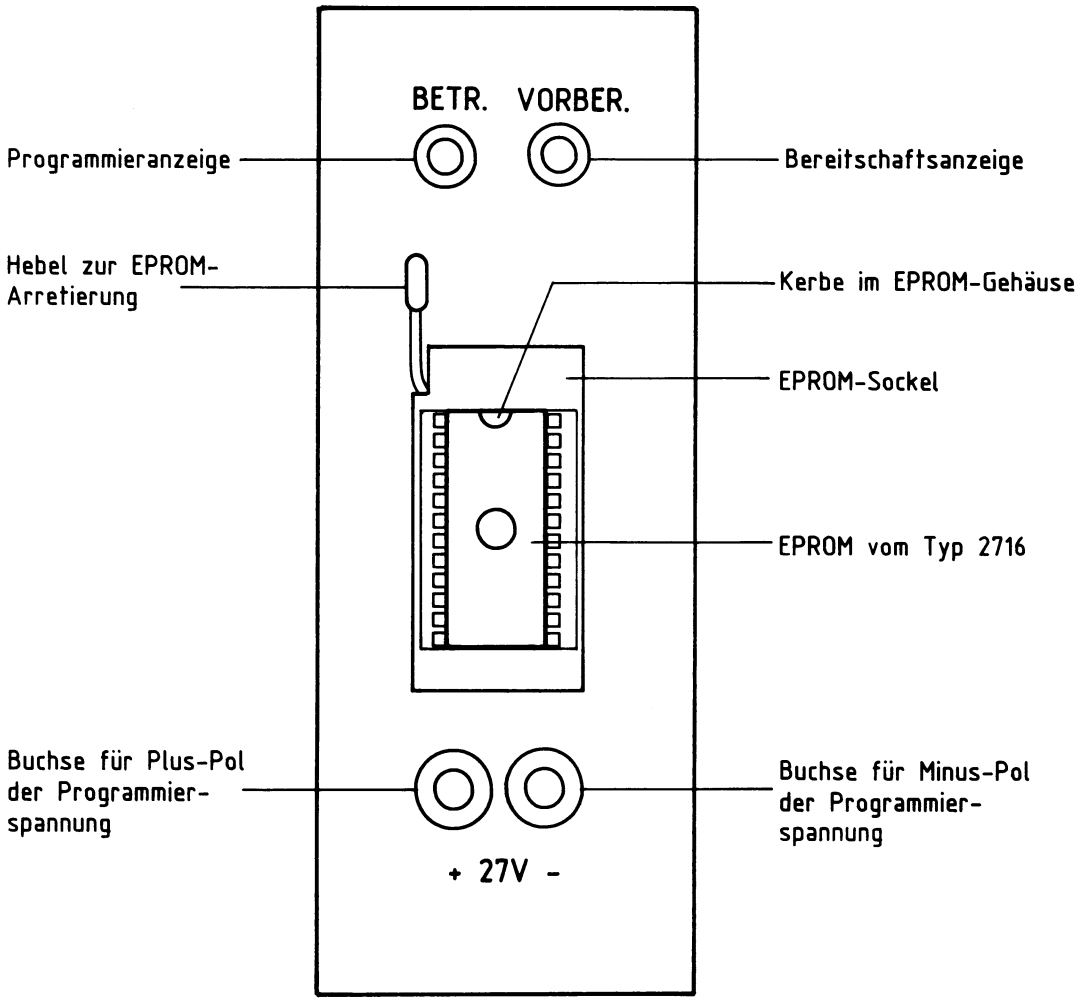


Bild 3: Frontplatte des EPROM-Programmierers

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	C [CR] eintippen, "OMPARE" wird ergänzt
RAM:	zuerst werden die Adressen für den Speicher angefordert
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]
EPROM:	die Adresse für das EPROM wird angefordert
START-ADR = X3X3 Y3Y3	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-Speicher-zellen 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	R [CR] eintippen, "EAD" wird ergänzt
RAM:	Das Programm fragt nach der START- und STOP-Adresse des RAMs, in den der EPROM-Inhalt eingelesen werden soll.
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]
EPROM:	Das Programm erfragt nun die EPROM-START-Adresse:
START-ADR = X3X3 Y3Y3	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 Speicherzellen 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

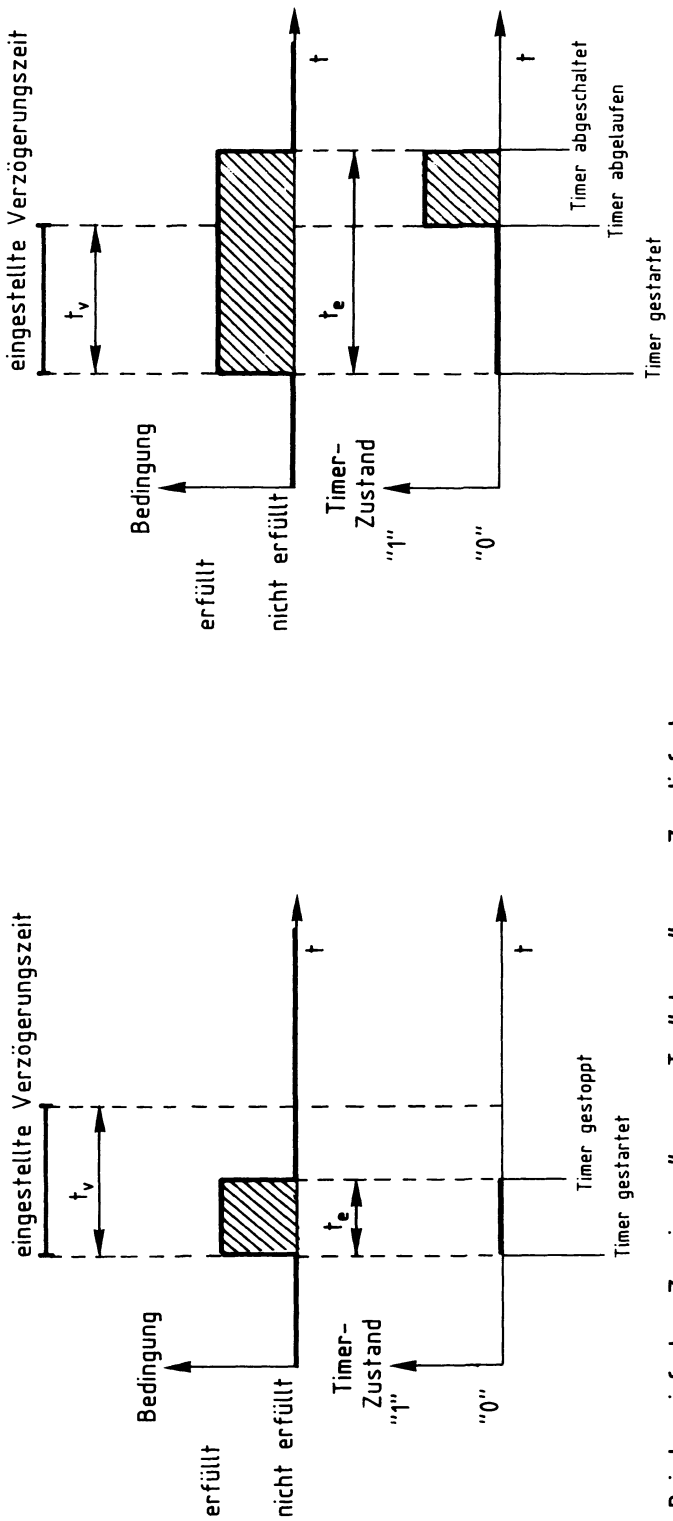
Startet man Timer mit der SETZ-Anweisung, werden die Timer nicht gestoppt, wenn die SETZ-Bedingung bei einem späteren Programm-Durchlauf nicht mehr erfüllt ist. Der Zustand "1" (abgelaufen) wird in diesem Fall gespeichert und kann nur mit dem RÜCKSETZ-Befehl gelöscht werden.
Startet man Timer mit einer 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
```

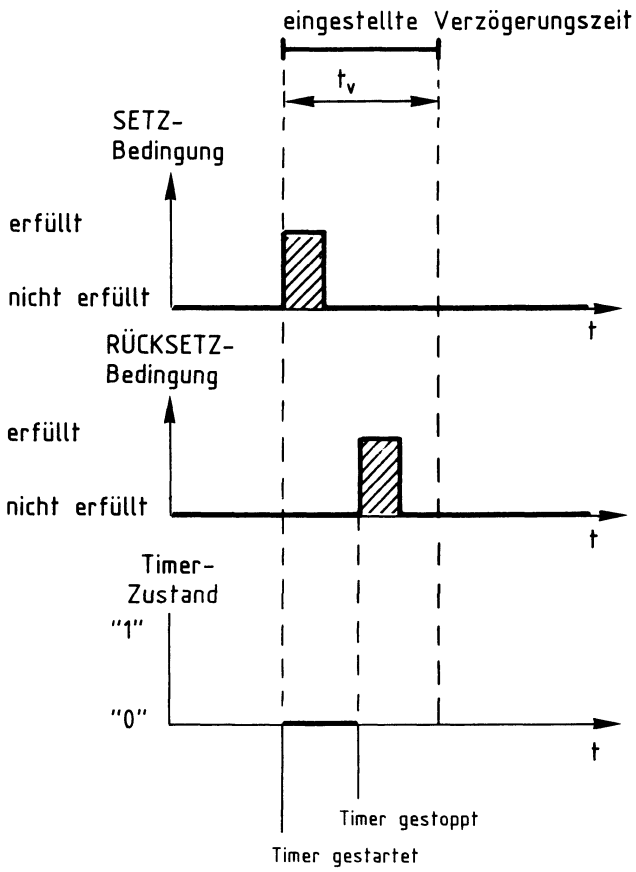
entspricht: Lade den Software-Timer Z00 mit der Laufzeit 10 Sekunden (100/10 Sekunden). Der Ladewert wird gespeichert und durch die Aktivität des Timers nicht verändert. Dies hat zur Folge, daß die einmal abgespeicherte Laufzeit mehrfach im Programm verwendet werden kann. Eine Änderung des Wertes ist durch einen neuen Ladebefehl möglich. Software-Timer dürfen auf der Bedingungs- und auf der Zuweisungsseite eines Ausdrucks vorkommen.

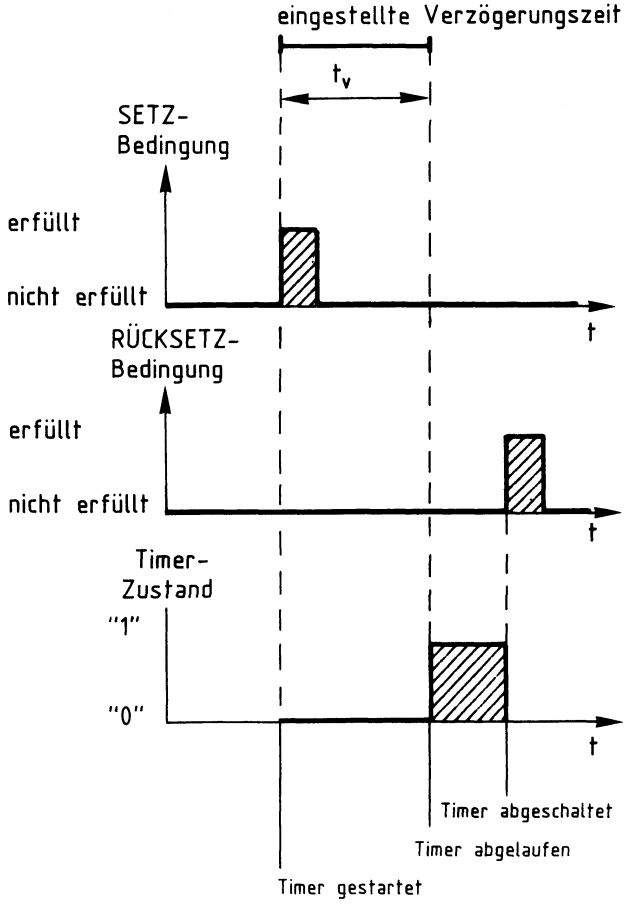
Hinweis: Zum Betrieb der Software-Timer ist die im Anhang beschriebene Schaltungserweiterung erforderlich.



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

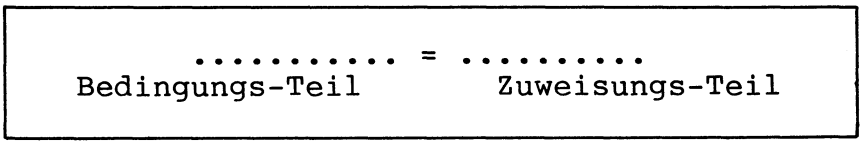
4.3.2. Die Operationen

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.



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:

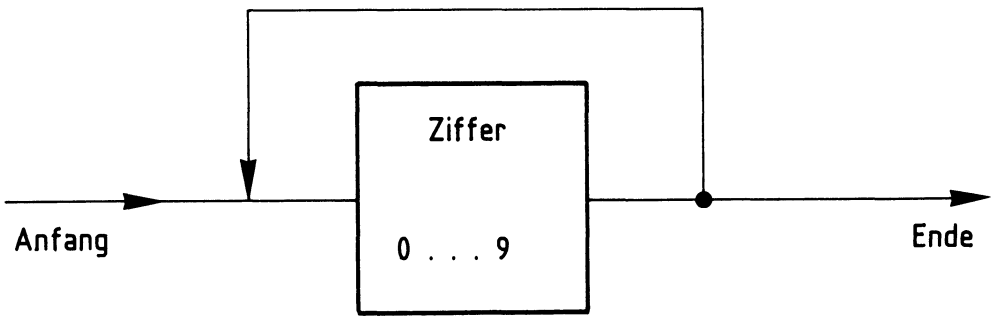
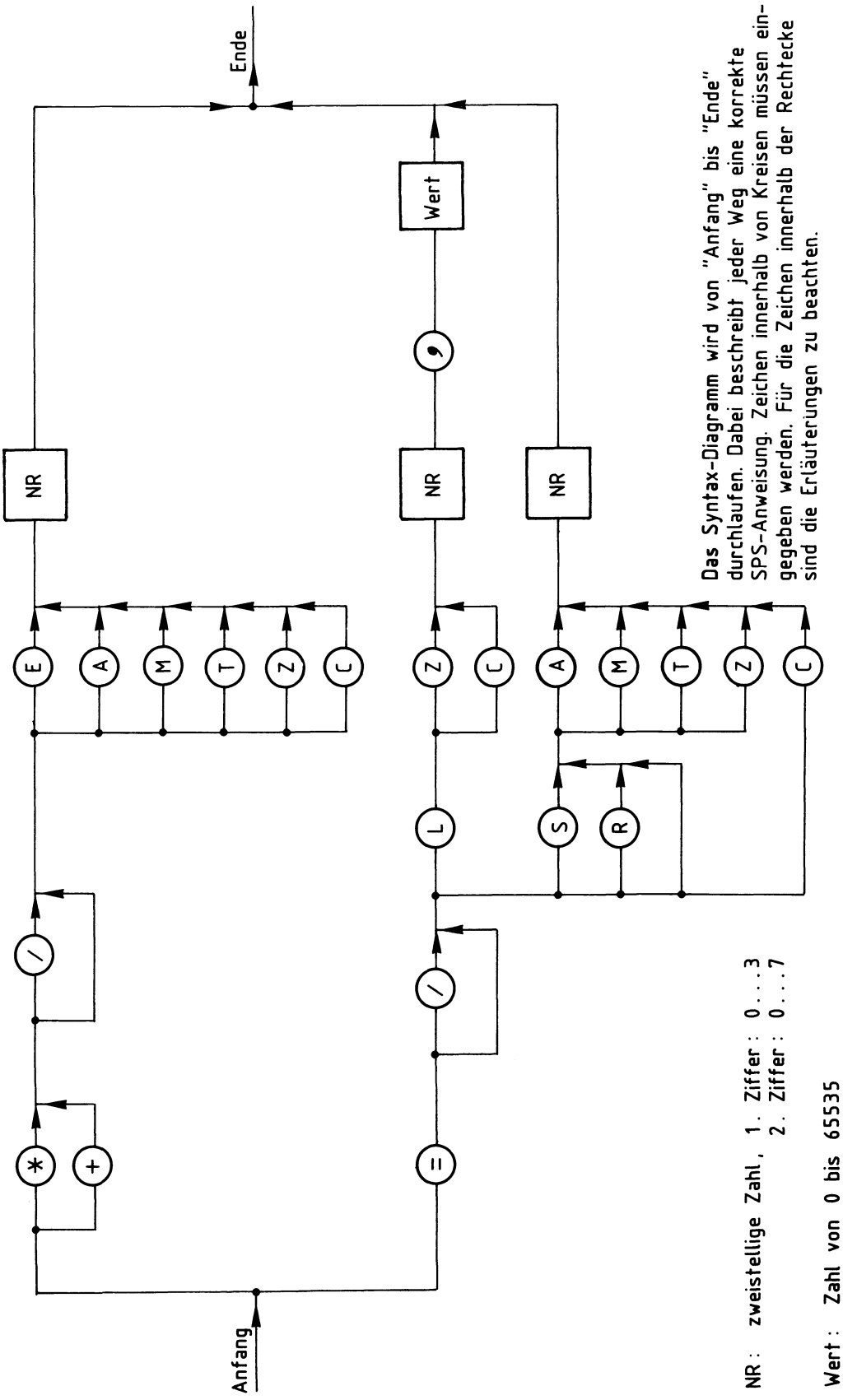


Bild 5: Syntax-Diagramm zur Bildung von Zahlen

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

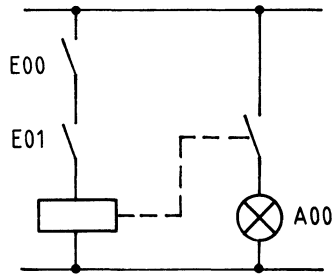
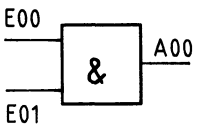
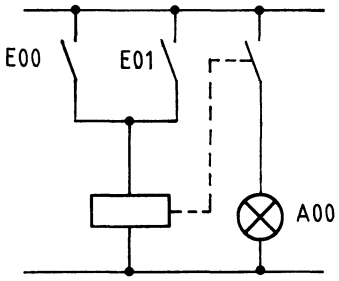
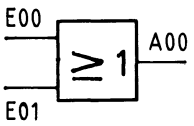
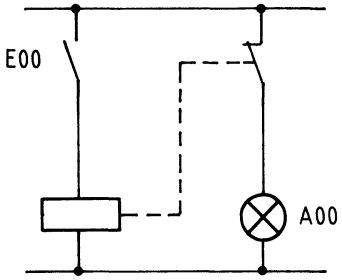
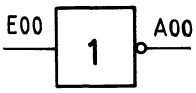
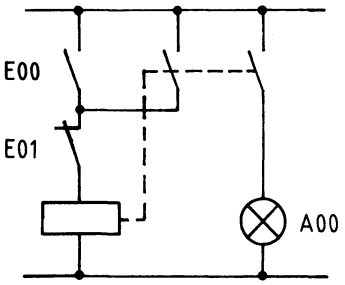
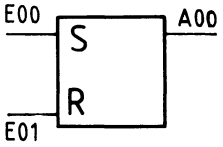
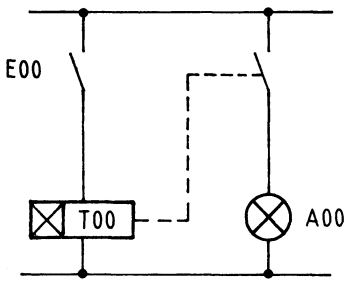
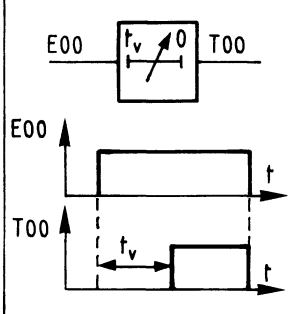
	Kontaktschaltung	Logikplan	Sprachliche Beschreibung	SPS-Programm
UND			(UND) E00 UND E01 GLEICH A00	* E00 * E01 = A00
ODER			(UND) E00 ODER E01 GLEICH A00	* E00 + E01 = A00
NICHT			(UND) NICHT E00 GLEICH A00 oder UND E00 GLEICH NICHT A00	*/E00 = A00 oder * E00 =/A00
SETZE/RÜCKSETZE			(UND) E00 UND NICHT 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:

```
*E01  <*E02


*E01


*E02
```

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:

```
*E02  >


*E01
```

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	Eingabe: ?0 CR Suche nächste Anweisung, die eine Null enthält.
*E01 ?0	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.
=A00 ?0	Die nächste Anweisung,die eine Null enthält, wird angezeigt. Eingabe: ? CR Suche das gesuchte Zeichen ("0") erneut.
=A00 ?0 NICHT GEFUNDEN =A00	Keine weiteren Anweisungen mit "0". Es wird die gleiche Anweisung angezeigt wie vor dem Suchbefehl. Eingabe: ↑ (Sprung an den Programmanfang)
ANF: *E00 ?E01	Eingabe: ?E01 CR Suche nächste Anweisung mit dem Operanden E01.
*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 entgegen-zunehmen.

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 entgegen-zunehmen.

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.

Arbeitsblatt	BFZ / MFA 7.2. - 98
Softwarepaket SP 1	Name:
SPS-Programm / STEP-Kommando	Datum:

Geben Sie folgendes Programm ein:

S3 SPS

```
*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		

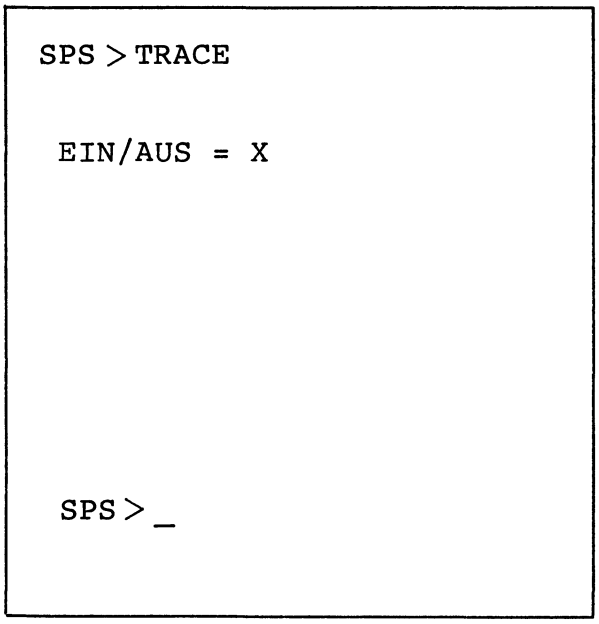
- Schalten Sie nun E00 auf "EIN"
- Protokollieren Sie die nächsten Schritte:

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:



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
```

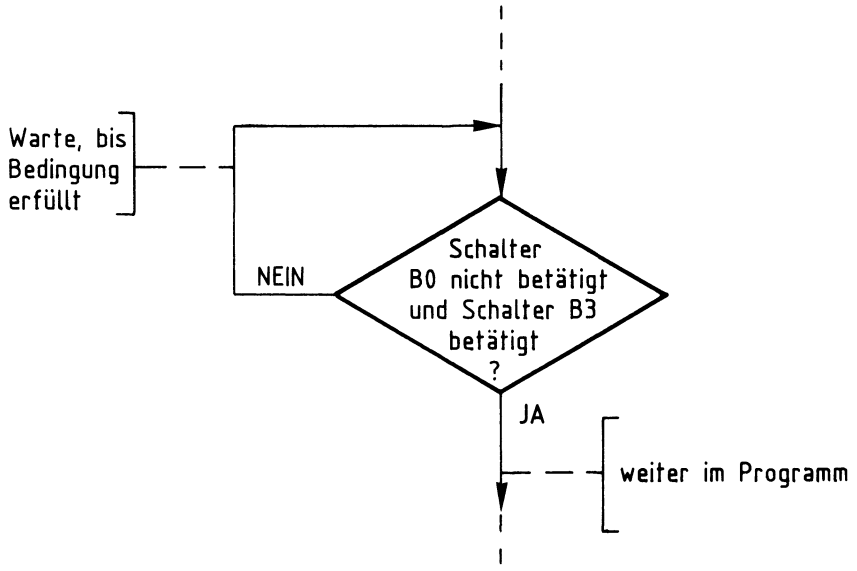
Rufe das Unterprogramm auf, das ab der Adresse 5678 (dezimal) im Speicher steht.
Der Wert 44 (dezimal) steht beim Start des Unterprogramms im HL-Registerpaar der CPU.
Der Wert, der bei der Rückkehr vom Unterprogramm im HL-Registerpaar steht, wird mit 2 multipliziert und der Variablen X zugewiesen.

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.

Die 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!

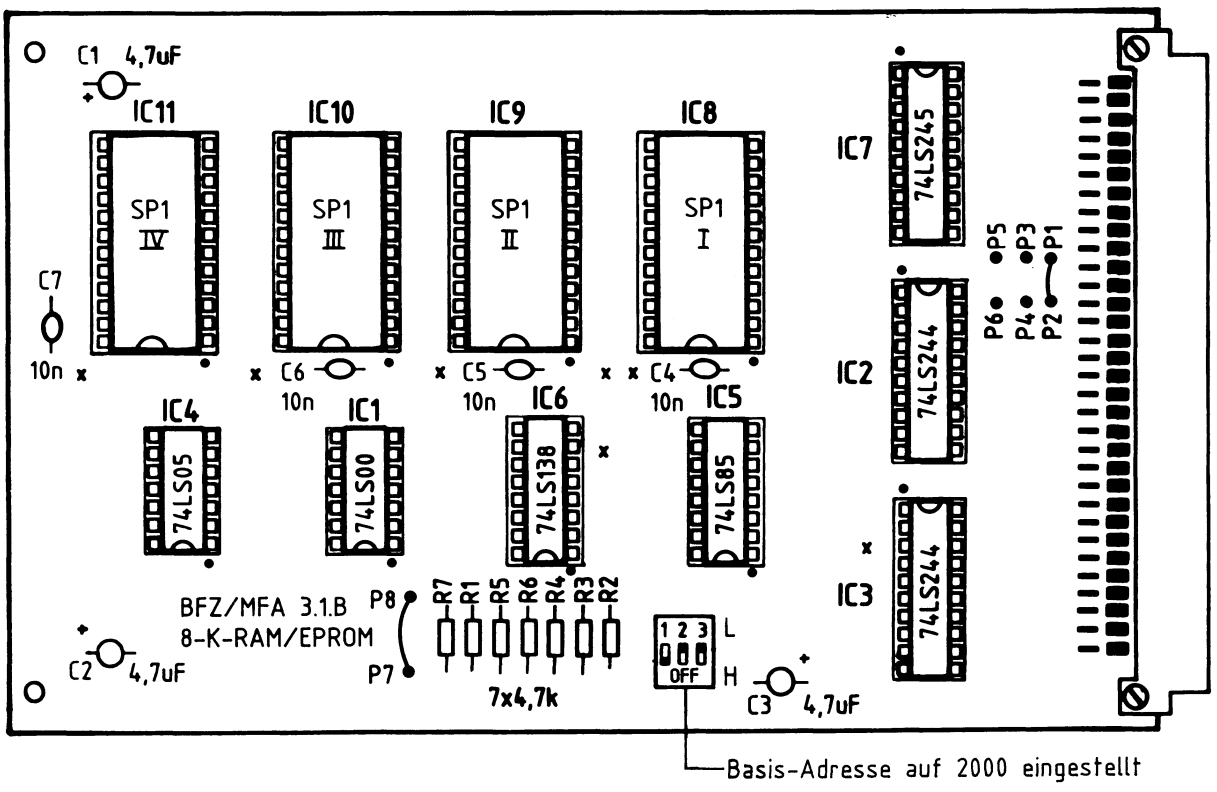
4.4.8. Das #-Symbol

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).

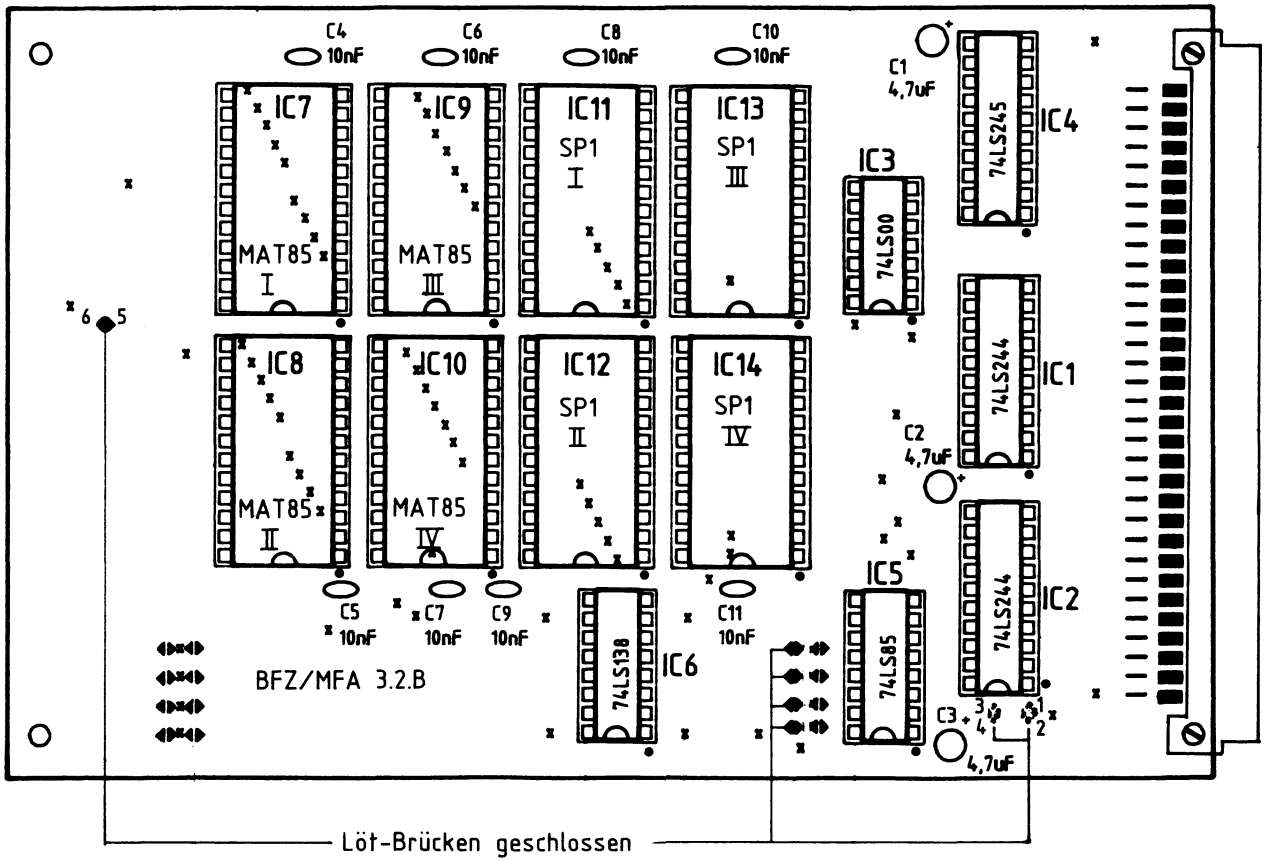
5. Anhang

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



(Lötbrücken 1-2 und 3-4 auf der Leiterbahnseite)

16-K-RAM/EPROM-Baugruppe bestückt mit MAT 85 und SP 1

5.2. Schaltungserweiterungen

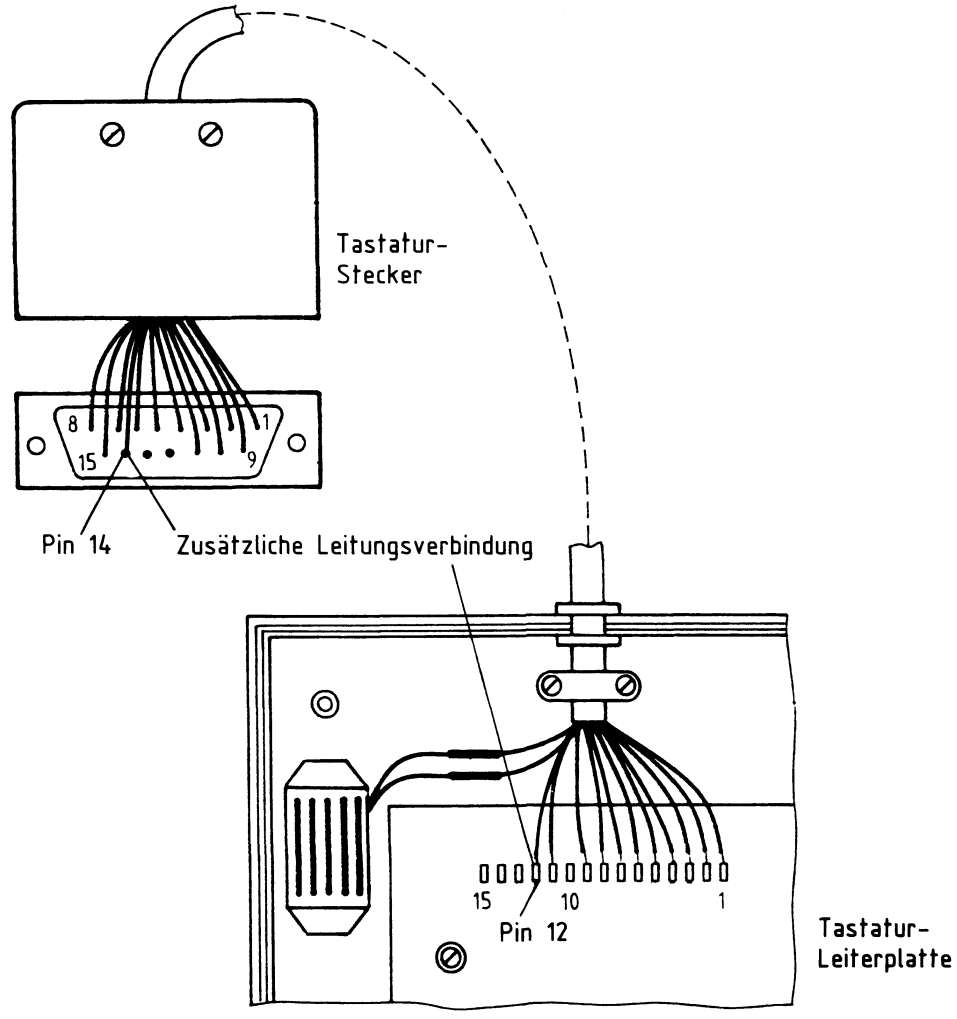
Um den SPS- und den BASIC-Interpreter voll nutzen zu können, sind zwei Schaltungs-Ergänzungen notwendig:

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



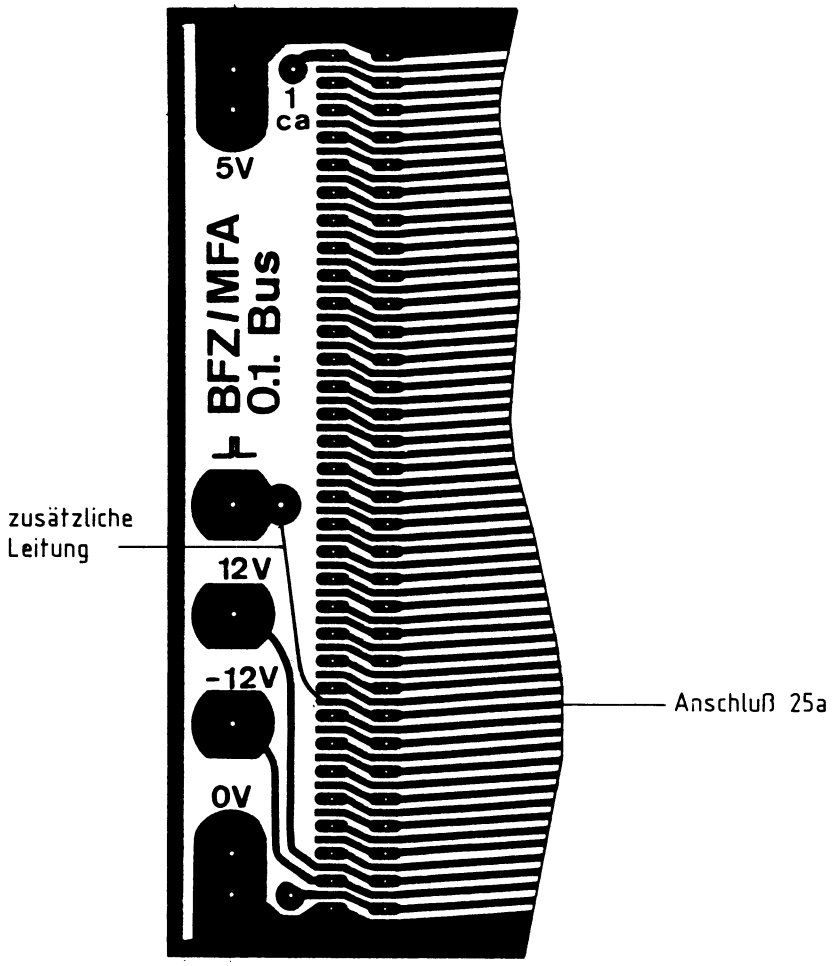
2. Der Anschlußpin 14 der Tastaturbuchse auf der Videokarte muß mit dem Anschluß-Stift 27c der 64-poligen Steckerleiste verbunden werden.

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



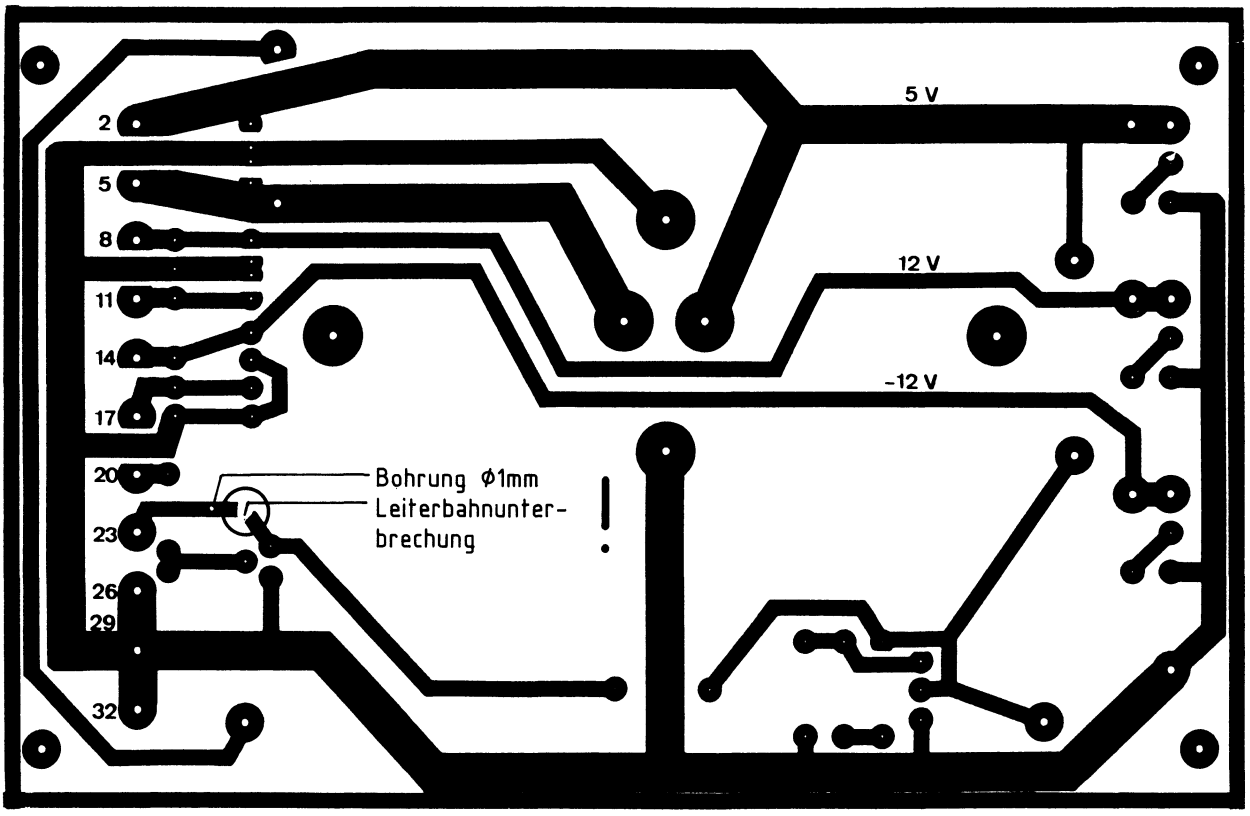
Anhang

Aufgrund von Bauteilstreuungen kann es vorkommen, daß der Rechteckimpuls verformt ist. In diesem Fall sollte die Schaltung der Spannungsregelung folgendermaßen ergänzt werden:

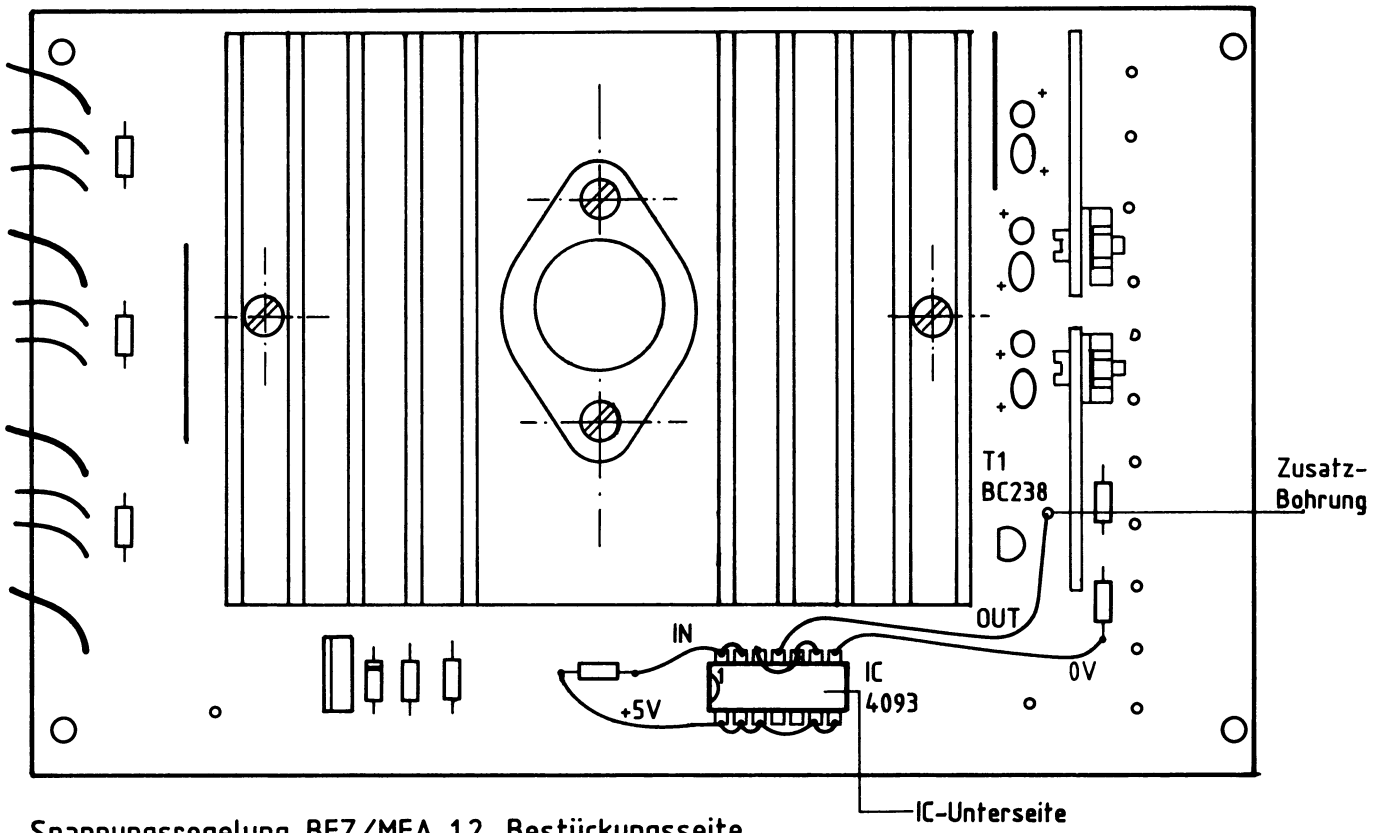
1. Trennen Sie die im Layout gekennzeichnete Leiterbahn auf und bohren Sie an der gekennzeichneten Stelle ein Loch von 1mm-Durchmesser.

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.

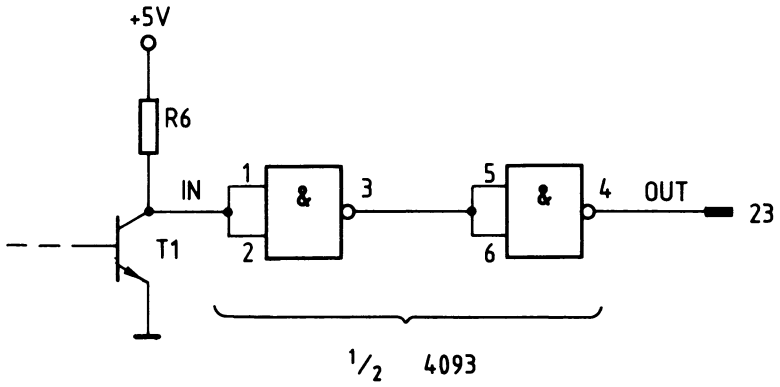
3. Verdrahten Sie das IC entsprechend dem Schaltbild.



Spannungsregelung BFZ/MFA 1.2. Leiterbahnseite



Spannungsregelung BFZ/MFA 1.2. Bestückungsseite



+5V: Anschlüsse 8, 9, 12, 13, 14
0V: Anschluß 7

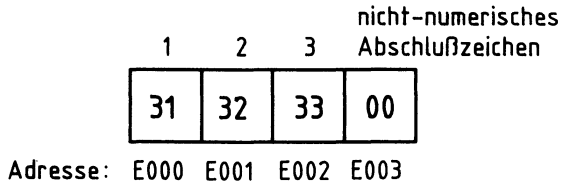
Schaltbild

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.

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`
			.

Anhang

Unterpr. Name	Eingangs-Adresse	veränd. Register	Funktion des Unterprogramms
TNO	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.

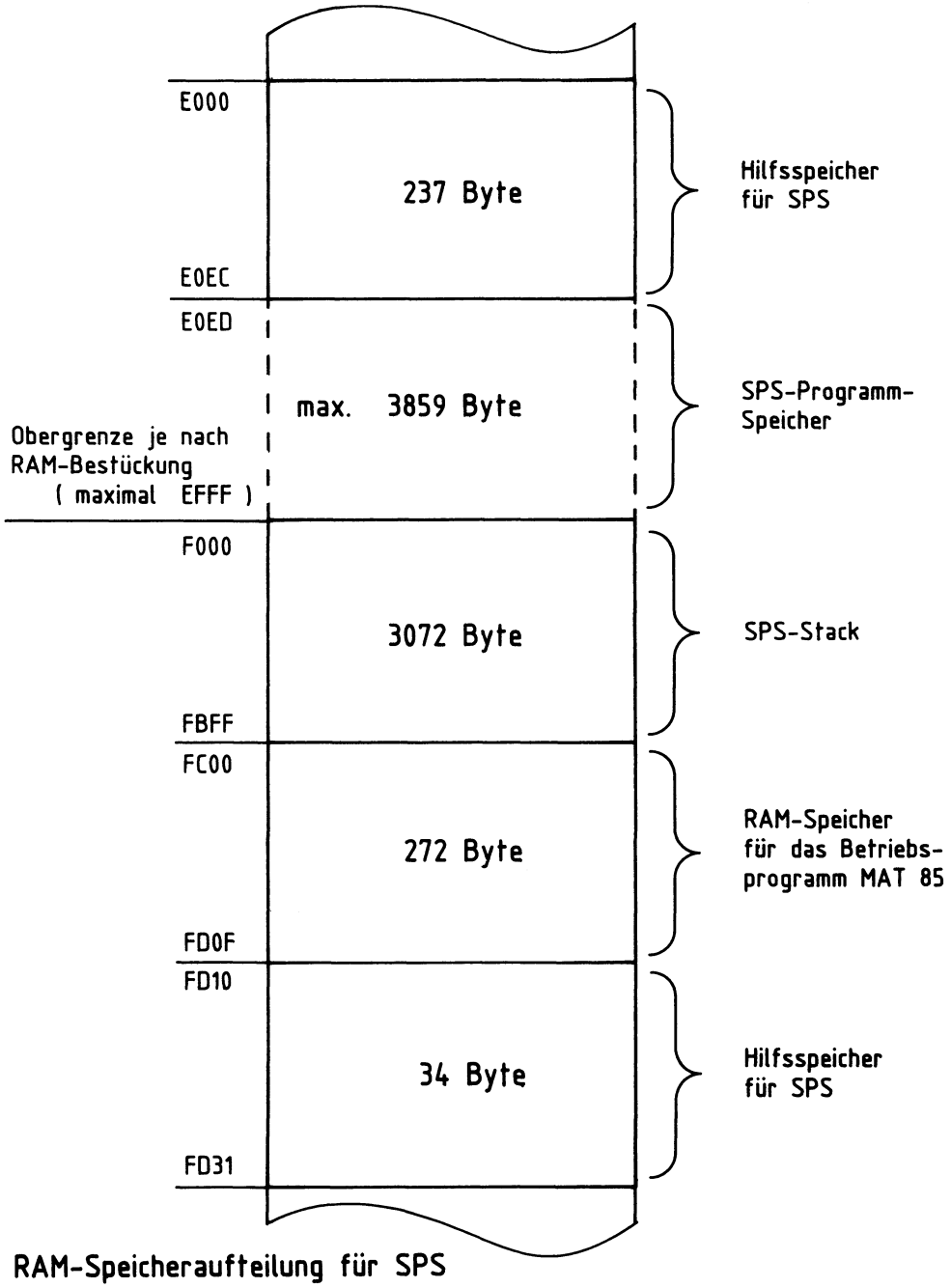
Das Ende des Programm-Speichers ist in den Speicherzellen 6066 (LSB) und 6067 (MSB) gespeichert.

Die Adresse der obersten RAM-Speicheradresse plus Eins, die von BASIC genutzt wird, ist in den Speicherzellen 606C (LSB) und 606D (MSB) gespeichert.

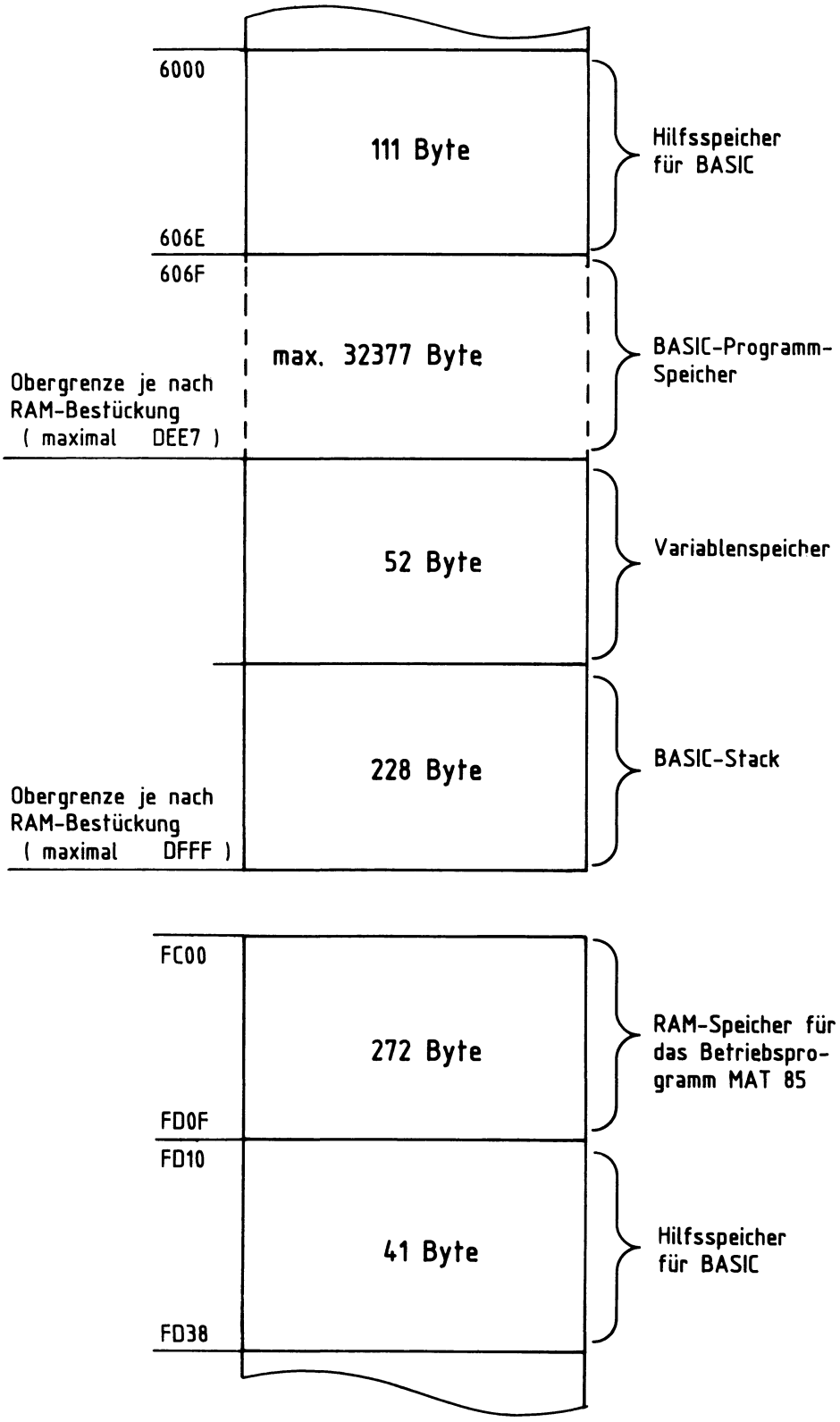
Speicheradressen für BASIC-Variable:

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



RAM-Speicheraufteilung für BASIC