ITT MP-Experimenter

Bedienungsanleitung

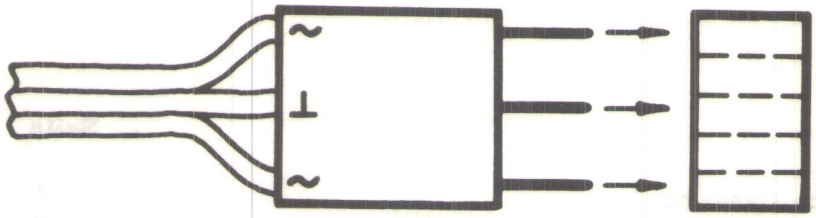
ITT Fachlehrgänge, Pforzheim

ITT MP-Experimenter

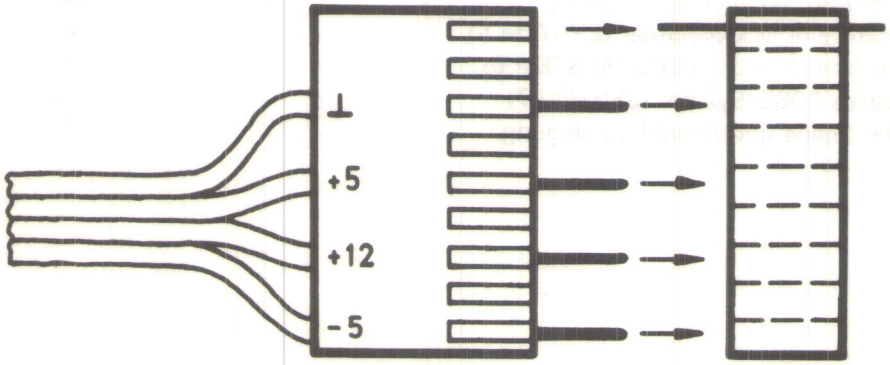
Hinweis zu den Steckverbindungen des MP-Experimenters

Achtung!

Beim Herstellen der Steckverbindungen ist besondere **Sorgfalt** geboten. Werden die Steckverbindungen nicht richtig hergestellt, werden die hochwertigen ICs vernichtet. Deshalb orientieren Sie sich an den Skizzen der beiden Steckverbindungen.



Steckverbindung Trafo-Netzteil



Steckverbindung Netzteil-Experimenter

1. Funktionsbeschreibung

Der MP-Experimenter ist ein Mikrocomputer basierend auf dem MP-System 8080.
Er ist ein Lehrsystem zur praktischen Einführung in den Gebrauch und die Arbeitsweise von Mikrocomputern. Der MP-Experimenter besteht aus der Stromversorgung, der Prozessorplatine und dem Frontpanel mit diversen Schablonen und Code-Karten.
Die Stromversorgung benötigt zum Betrieb lediglich eine Steckdose des 220-V-Netzes.

Der separat gekapselte Netztransformator gibt zum Zwecke einer Zweiweggleichrichtung 2 x 8,5 V_{eff} ab. Die an ihm über eine 3polige Steckverbindung angeschlossene Netzteilplatine enthält 3 geregelte Stromversorgungen für die Betriebsspannungen +5 V, +12 V und −5 V gegen Masse. Je 2 Dioden eines Brückengleichrichters arbeiten als Zweiweggleichrichter für den +5-V- bzw. −5-V-Regler. Der zweite Brückengleichrichter arbeitet als Zweiweg-Villard-Gleichrichter für den +12-V-Regler. Die Ausgänge aller Regler sind kurzschlußfest.
Der Mikrocomputer enthält als CPU den Mikroprozessor 8080 A und die Ergänzungsbausteine 8224 (Taktgenerator) und 8228 (System-Controller und Datenbus-Treiber). Ein Quarz von 8,867 MHz erzeugt eine Taktfrequenz von ca. 1 MHz, so daß der Prozessor mit einer Zykluszeit von ca. 1 µs arbeitet.
Das eingebaute Memory besteht aus einem maskenprogrammierten ROM (8308) von 1 k x 8 bit, welches das Systembetriebsprogramm (Monitor) enthält und einem statischen RAM (2 x 8111) von 1/4 k x 8 bit als STACK, Daten- und Programmspeicher für Anwenderprogramme.

Dem ROM ist der Adreßbereich von $0\,0\,0\,0_{16}$ bis $0\,3\,F\,F_{16}$ und dem RAM der Adreßbereich von $0\,4\,0\,0_{16}$ bis $0\,4\,F\,F_{16}$ durch den Adreßdecoder (2/6 74 LS 04, 6/6 74 LS 05 und 3/4 74 LS 32) zugeordnet. Der verbleibende Adreßbereich von $0\,5\,0\,0_{16}$ bis $F\,F\,F\,F_{16}$ (also 62 3/4 k = 64 256 Adressen) steht ohne Einschränkung für Erweiterungen zur Verfügung.
Weiterhin enthält die Computerplatine insgesamt 5 8-bit-Parallel-I/O-Bausteine (5 x 8212), von denen 3 als Input-Ports und 2 als Output-Ports geschaltet sind. Sie sind in der „isolierten I/O-Adressierung" mit den Adressen $0\,0\,0\,1_{16}$, $0\,0\,0\,2_{16}$ und $0\,0\,0\,4_{16}$ für die Input-Ports und $0\,0\,0\,1_{16}$ und $0\,0\,0\,2_{16}$ für die Output-Ports adressiert.
Da keine vollständige Adreßcodierung angewendet wurde, sondern Adreß-bit-Adressierung, stehen für weitere I/O-Adressen nur noch $0\,0\,0\,8_{16}$, $0\,0\,1\,0_{16}$, $0\,0\,2\,0_{16}$, $0\,0\,4\,0_{16}$ usw. für Inputs und $0\,0\,0\,4_{16}$, $0\,0\,0\,8_{16}$, $0\,0\,1\,0_{16}$ usw. für Outputs zur freien Verfügung.

Am System-RESET (RESET-Eingang des 8224) liegt eine *RCD*-Kombination zum automatischen RESET beim Einschalten der Betriebsspannung.
Die System-Steuersignale HOLD, INT und BUS EN sind über Inverter geführt, die am Eingang einen 1-kΩ-Pull-up-Widerstand tragen. Als äußere Eingangssignale stehen sie somit als active-low-Signale ($\overline{\text{HOLD}}$, $\overline{\text{INT}}$, BUS EN) zur Verfügung, d.h. ein äußeres Low-Signal an $\overline{\text{HOLD}}$ steuert den 8080 in den HOLD-Zustand, ein Low-Signal an $\overline{\text{INT}}$ gibt an den 8080 eine Unterbrechungs-Anforderung (Interrupt-Request) und ein Low-Signal an BUS EN steuert die Datenbus-Ausgänge sowie die Steuerbus-Ausgänge vom System-Controller (8228) in den Tri-State-Zustand. Als weitere Eingänge sind noch $\overline{\text{RES IN}}$ und RDY IN vorhanden. Ein äußeres Low-Signal an $\overline{\text{RES IN}}$ bewirkt einen RESET, ein Low-Signal an RDY IN bringt den Prozessor in den Warte-Zustand.
An Steuersignal-Ausgängen stehen RESET (pos. Impuls) zum Rücksetzen externer Elemente, $\overline{\text{STSTB}}$ (neg. Impuls) zum Abfragen der Statusinformation, WAIT (pos. Impuls) zur Anzeige des Warte-Zustandes, HLDA (pos. Impuls) als Quittierung einer HOLD-Anforderung und INTE, das anzeigt, ob ein Interrupt ein- (INTE = H) oder ausgeschaltet (INTE = L) ist, zur Verfügung.

Herausgeführt sind weiterhin die 5 Signale des Steuerbusses $\overline{\text{MEM R}}$ (neg. Impuls für Speicher Lesen), $\overline{\text{MEM W}}$ (dto. für Speicher Schreiben), $\overline{\text{I/O R}}$ (dto. Eingangs-/Ausgangs-Port Lesen), $\overline{\text{I/O W}}$ (dto. Eingangs-/Ausgangs-Port Schreiben) und $\overline{\text{INTA}}$ (Interrupt-Acknowledge = Interrupt-Bestätigung), der wegen fester Verbindung über 1 kΩ an +12 V einen automatischen RST 7-Befehl erzeugt.
Die Prozessor-Platine ist – von oben verdeckt – unter die Oberplatine (Front-Panel) gesteckt.

Das Front-Panel trägt einmal die Steckleisten zum Anschluß der Stromversorgung, zum Anschluß anderer Ein-/Ausgaben und für Systemerweiterungen und zum anderen die diversen Eingabe-Schalter und LED-Anzeigen zur Ausgabe.

Der mit „SYSTEM" gekennzeichnete Codierschalter mit den Zahlen 0 bis 9 dient zur Auswahl der entsprechenden Monitorprogrammteile, die die einzelnen Experimentierschritte simulieren. Die Stellungen 0 bis 6 sind den Systemen „Addierer/Subtrahierer" (0), „Codierte ALU" (1), „Akkumulator" (2), „Akku mit Speicher" (3), „Vereinfachter Rechner" (4), „Hypothetischer Rechner" (5) und „Prozessorsystem 8080" (6) zugeordnet. Die Stellung 7 dient für Erweiterungen mit zusätzlichen Betriebsprogrammen, während die Stellungen 8 und 9 unbenutzt bleiben.
Die binären Schiebeschalter C_4 bis C_0 dienen als Funktionsschalter unterschiedlicher Bedeutung in den verschiedenen Systemen. Auch die mit A_7 bis A_0 bzw. B_7 bis B_0 gekennzeichneten binären Schiebeschalter haben in den verschiedenen Systemen unterschiedliche funktionelle Bedeutung. Sie dienen in erster Linie zur Eingabe von Daten oder Adressen.

Als Ausgabe- bzw. Anzeigeelemente sind 2 x 8 LEDs mit der Bezeichnung L_7 bis L_0 und R_7 bis R_0 angeordnet. Sie gestatten das Ablesen von Adressen oder Daten im Maschinencode (Binärcode, d.h. leuchtende LED = log. 1, nichtleuchtende LED = log. 0). Ihre funktionelle Bedeutung ist unterschiedlich und wird von den Schablonen festgelegt.
Die mit „RESET" bezeichnete Taste bringt den Prozessor zum Programmstart, d.h. nach deren Betätigung beginnt der Prozessor das Monitorprogramm ab Adresse $0\,0\,0\,0_{16}$ abzuarbeiten. Die RESET-Taste hat keine „Clear-Funktion", d.h. es werden keine Register- oder Speicherinhalte gelöscht! Sie ist jeweils nach der Veränderung des Systemwahlschalters zu betätigen, damit ein eindeutiges Einlaufen in das gewählte Systemprogramm gewährleistet ist.
Die farblich gekennzeichneten und mit der System-Nr. versehenen Schablonen weisen den Bedienungselementen der Frontplatte ihre systemspezifische Funktion zu. Die jeweils in der gleichen Farbe vorhandenen Codekarten (DIN A 6) bzw. Befehlslisten (195 x 175 mm) dienen zur Kurzinformation über Funktions- bzw. Displaycodes (Anzeigecodes).

2. Technische Daten

MP-System	8080 A
mit Taktgenerator	8224
und System-Controller	8228
Quarzfrequenz	8,867 MHz
Zykluszeit	ca. 1 µs
Memory	ROM 1 k x 8 bit (Typ 8308) mit System-Betriebsprogramm RAM 1/4 k x 8 bit (2 x Typ 8111) für Anwenderprogramme
Memory-Erweiterung	möglich ab Adresse $0\,5\,0\,0_{16}$ bis Adresse $F\,F\,F\,F_{16}$
Input	3 x 8 bit parallel mit 3 x I/O-Port 8212 für C-, B- und A-Schalter, bit-weise I/O-Adresse mit Adreß-bit 0, 1 bzw. 2
Input-Erweiterung	möglich mit den Adreß-bits 3, 4, 5, 6 und 7
Output	2 x 8 bit parallel mit 2 x I/O-Port 8212 für L- und R-LEDs, bit-weise I/O-Adresse mit Adreß-bit 0 und 1
Output-Erweiterung	möglich mit den Adreß-bits 2, 3, 4, 5, 6 und 7

fan-out an der Systemleiste links (1 LE = 1 x TTL ≙ 1,6 mA_L, 40 μA_H)

Adreßbus	$AB_0 \ldots AB_{15}$	1	LE
Datenbus	$DB_0 \ldots DB_7$ (bidirektional)	5	LE
Steuerbus	$\overline{MEM\ R}$, $\overline{MEM\ W}$, $\overline{I/O\ R}$, $\overline{I/O\ W}$, $\overline{INTA}$	5	LE
HLDA	(Halt-Quittung)	1	LE
INTE	(Unterbrechungs-Freigabe)	1 1/4	LE
RESET	(Rückstellung)	1 1/4	LE
$\overline{STSTB}$	(Zustandsübernahme)	1	LE
WAIT	(Wartezustandsquittung)	1 1/4	LE

fan-in an der Systemleiste links

Datenbus	$DB_0 \ldots DB_7$ (bidirektional)	1/4 LE
BUS EN	(BUS-Freigabe-Steuerung)	3 1/4 LE
$\overline{HOLD}$	(HALT-Anforderung)	3 1/2 LE
$\overline{INT}$	(Unterbrechungs-Anforderung)	3 1/2 LE
RDY IN	(Bereit-Eingang)	3 1/2 LE
$\overline{RES\ IN}$	(Rücksetz-Eingang)	1/4 LE (+ 1 µF)

fan-out an der Systemleiste rechts (in TTL-LE)

$L_7 \ldots L_0$	(I/O-Port mit Adr. $0\,0\,0\,2_{16}$)	10 LE
$R_7 \ldots R_0$	(I/O-Port mit Adr. $0\,0\,0\,1_{16}$)	10 LE

(U_{OH} = 3,2 V, da LED angeschlossen)

fan-in an der Systemleiste rechts (in TTL-LE)

$C_7 \ldots C_0$	(I/O-Port mit Adr. $0\,0\,0\,4_{16}$)	1 LE
$B_7 \ldots B_0$	(I/O-Port mit Adr. $0\,0\,0\,1_{16}$)	1 LE
$A_7 \ldots A_0$	(I/O-Port mit Adr. $0\,0\,0\,2_{16}$)	1 LE

Als Eingänge nur verwendbar, wenn $C_4 \ldots C_0$, $B_7 \ldots B_0$ und $A_7 \ldots A_0$ in Stellung „1" und Systemschalter in Stellung „0" oder „8" stehen (wired OR)!
(U_{IHmax} = 5,0 V, U_{ILmin} = −0,5 V)

Stromversorgung

Kompaktnetztransformator	220 V/2 x 8,5 V, EI 66
Regelnetzteil	+5 V, 1,5 A
(kurzschlußfest)	+12 V, 250 mA
	−5 V, 100 mA
Strombedarf des MP-Experimenters	+5 V, 1 A typ.
	+12 V, 100 mA typ.
	−5 V, 2 mA typ.
Stromreserve für Erweiterungen	+5 V, 500 mA_{max}
	+12 V, 150 mA_{max}
	−5 V, 100 mA_{max}

3. Bedienungsanleitung

3.1 Allgemeine Hinweise zu den Experimenten

Das Experimentiersystem enthält einen vollständigen Rechner. Im ROM sind 7 Programme zur Simulierung von verschiedenen Systemen fest abgespeichert. Welches Programm ablaufen soll, kann mit dem SYSTEM-Schalter (BCD-Schalter auf der linken Seite) festgelegt werden. Den 7 Programmen sind die Nummern 0 bis 6 zugeordnet. Die Stellung 7 des SYSTEM-Schalters ist für eine eventuelle Erweiterung des Systems vorgesehen, die Stellungen 8 und 9 werden nicht verwendet.

SYSTEM 0:	8-bit-Parallel-Addierer/Subtrahierer
SYSTEM 1:	Codierte 8-bit-ALU
SYSTEM 2:	8-bit-Akkumulator
SYSTEM 3:	8-bit-Akku mit 16 x 8-bit-Datenspeicher
SYSTEM 4:	Vereinfachter 8-bit-Ein-Adreßrechner
SYSTEM 5:	Hypothetischer 8-bit-Mikrorechner (didaktischer Modell-Rechner)
SYSTEM 6:	8-bit-Mikrorechner mit 8080-Mikroprozessor
SYSTEM 7:	für andere Mikrorechner mit dem 8080 und Erweiterungen

Ein Programm wird mit der RESET-Taste gestartet. Diese Taste entspricht in etwa der Löschtaste eines Taschenrechners und **muß am Anfang jedes Experimentes gedrückt werden**. Mit den restlichen Schiebeschaltern können Daten und Steuerinformationen eingegeben werden.

Die Schaltergruppe C_4 bis C_0 wird zur Steuerung des Experimentierablaufes benötigt. Die Schaltergruppen A_7 bis A_0 und B_7 bis B_0 werden bis auf einige Spezialfälle für die Daten- oder Programmeingabe benutzt.
Bei allen Experimenten gelten folgende Festlegungen:

Schalter oben = logisch 1
Schalter unten = logisch 0

Die 2 mal 8 Leuchtdioden dienen zur Anzeige der Rechnerergebnisse sowie zur Anzeige interner Schaltzustände. Hier gelten folgende Festlegungen:

Leuchtdiode leuchtet = logisch 1
Leuchtdiode dunkel = logisch 0

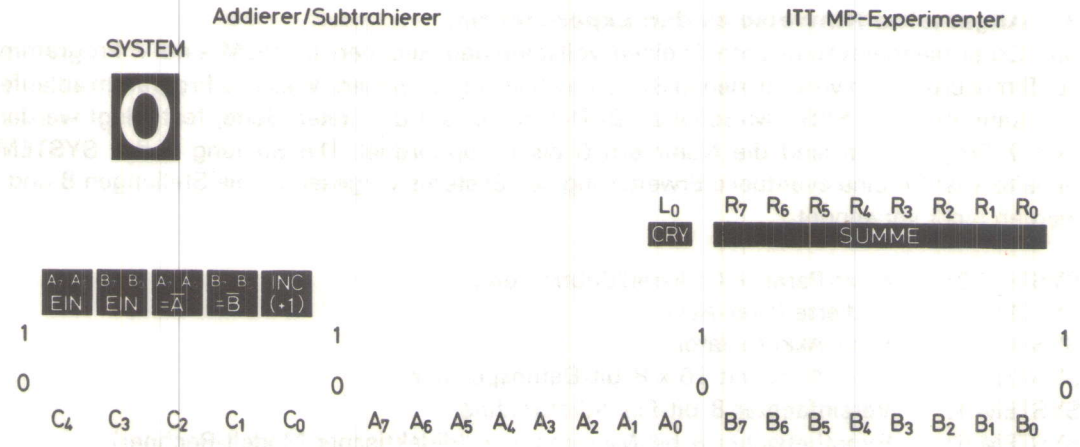
Alle Schalter, die bei einem bestimmten Experimentiervorgang nicht benötigt werden, sollten auf log. 0 geschaltet werden.
Es ist zu empfehlen, daß zu Beginn eines Experimentes alle Schalter auf log. 0 geschaltet werden, bevor die RESET-Taste gedrückt wird. Ausnahmen hiervon werden bei den einzelnen Experimenten angegeben.
Die grundsätzliche Experimentiervorbereitung ist folgende:

1. Die in der Experimentieranweisung angegebene Schablone auflegen
2. SYSTEM-Schalter auf das verlangte Programm einstellen
3. Alle Schiebeschalter auf Null (unten) stellen
4. RESET-Taste drücken

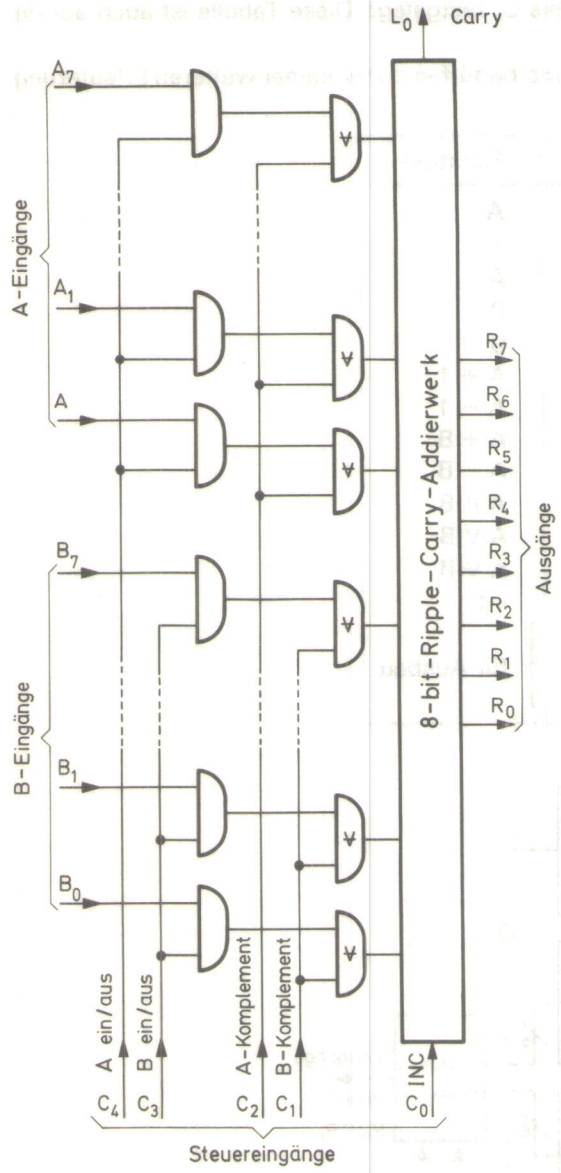
Bei falscher Schalterbetätigung können grundsätzlich keine Schäden am Experimentiersystem entstehen. Allerdings können dadurch die selbst eingegebenen Programme und Daten verändert werden, so daß ein falsches Ergebnis entsteht. Bei umfangreichen und komplizierten Experimenten kann eine falsche Betätigung viel Zeit kosten.

3.2 **Addierer/Subtrahierer** (SYSTEM 0)

Im System 0 läuft auf dem Rechner ein Programm, das ein 8-bit-Parallel-Addier-/Subtrahierwerk simuliert. Eine solche Schaltung ist schaltungstechnisch ein rein kombinatorisches Netzwerk mit statischer Betriebsweise (Bild). Es hat zweimal 8 Dateneingänge (A_7 bis A_0 = Operand A und B_7 bis B_0 = Operand B) und 5 Steuereingänge (C_4 bis C_0), die die in der Tabelle aufgeführten Funktionsmöglichkeiten ergeben.



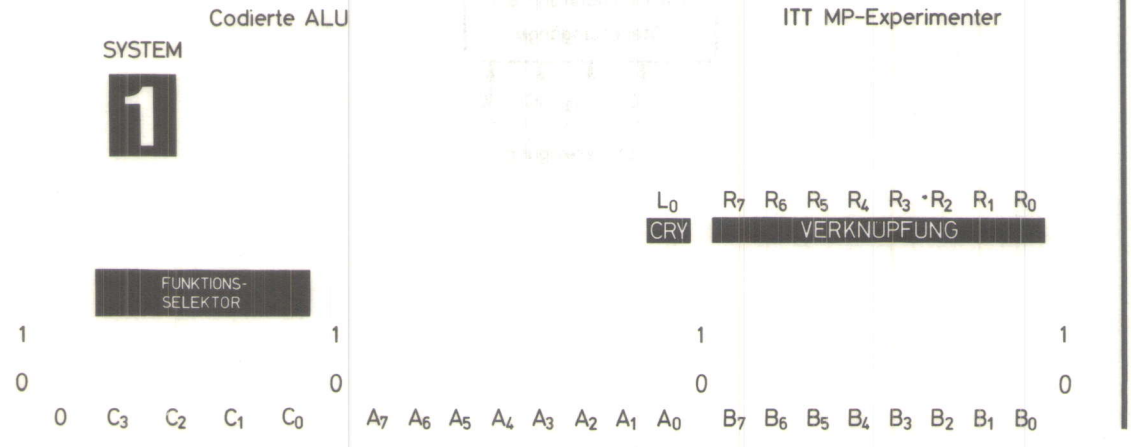
Nach der Experimentiervorbereitung Schalter C_4 und C_3 auf 1. Damit können die an den Schaltern A_7 bis A_0 und B_7 bis B_0 eingestellten Informationen in das System gelangen. Mit den Schaltern $C_2 = \overline{A}$ und $C_1 = \overline{B}$ können die eingegebenen Informationen komplementiert werden (Einerkomplement). Der Schalter C_0 = INC legt bei 1 eine 1 auf den INC-Eingang. Das Ergebnis der Addition erscheint in den rechten 8 LEDs (R_7 bis R_0). Die LED L_0 in der linken Lampengruppe zeigt einen Übertrag (Carry) an. Bei diesem System haben die LEDs L_7 bis L_1 keine Bedeutung.



C_4	C_3	C_2	C_1	C_0	Ausgangs-funktion
0	0	0	0	0	0
0	0	0	0	1	1
0	0	0	1	0	−1
0	0	0	1	1	0
0	0	1	0	0	−1
0	0	1	0	1	0
0	0	1	1	0	−2
0	0	1	1	1	−1
0	1	0	0	0	B
0	1	0	0	1	B + 1
0	1	0	1	0	$-B - 1 = \overline{B}$
0	1	0	1	1	−B
0	1	1	0	0	B − 1
0	1	1	0	1	B
0	1	1	1	0	−B − 2
0	1	1	1	1	$-B - 1 = \overline{B}$
1	0	0	0	0	A
1	0	0	0	1	A + 1
1	0	0	1	0	A − 1
1	0	0	1	1	A
1	0	1	0	0	$-A - 1 = \overline{A}$
1	0	1	0	1	−A
1	0	1	1	0	−A − 2
1	0	1	1	1	$-A - 1 = \overline{A}$
1	1	0	0	0	A + B
1	1	0	0	1	A + B + 1
1	1	0	1	0	A − B − 1
1	1	0	1	1	A − B
1	1	1	0	0	B − A − 1
1	1	1	0	1	B − A
1	1	1	1	0	−A − B − 2
1	1	1	1	1	−A − B − 1

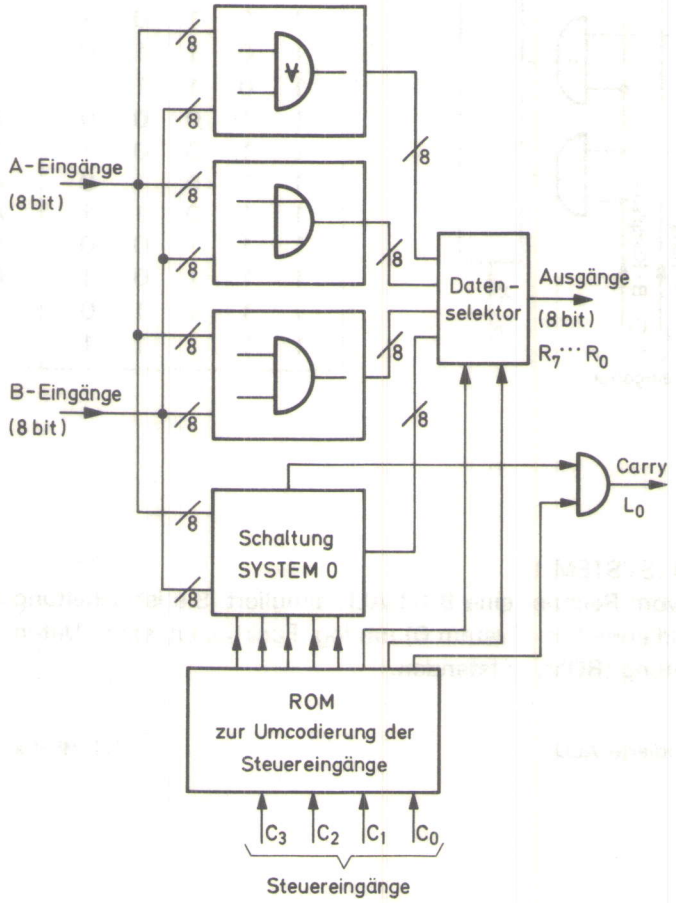
3.3 Codierte ALU (SYSTEM 1)

Im System 1 wird vom Rechner eine 8-bit-ALU simuliert. Sie ist schaltungstechnisch durch Erweiterung des Addierwerkes (System 0) um log. Funktionen, einen Datenselektor und eine Umcodierungsschaltung (ROM) entstanden.



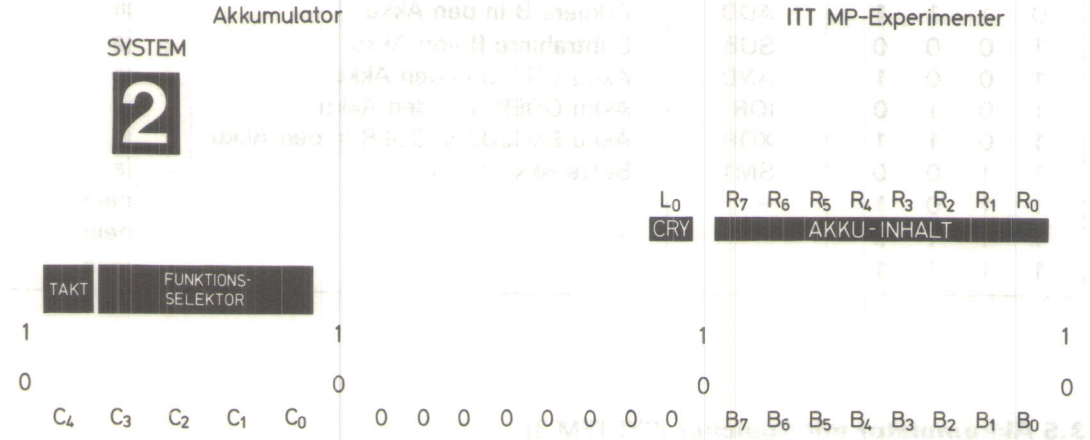
Die Funktionen werden mit den Schaltern C_3 bis C_0 festgelegt. Diese Tabelle ist auch auf der Karte „Codierte ALU" zu finden.
Die Funktionen sind alle ausreichend bekannt und bedürfen daher keiner weiteren Erläuterung.

C_3	C_2	C_1	C_0	Funktion
0	0	0	0	A
0	0	0	1	1
0	0	1	0	$\overline{A}$
0	0	1	1	B
0	1	0	0	0
0	1	0	1	A + 1
0	1	1	0	A − 1
0	1	1	1	A + B
1	0	0	0	A − B
1	0	0	1	A ∧ B
1	0	1	0	A ∨ B
1	0	1	1	A ∀ B
1	1	0	0	−1
1	1	0	1	für Ausbau
1	1	1	0	
1	1	1	1	

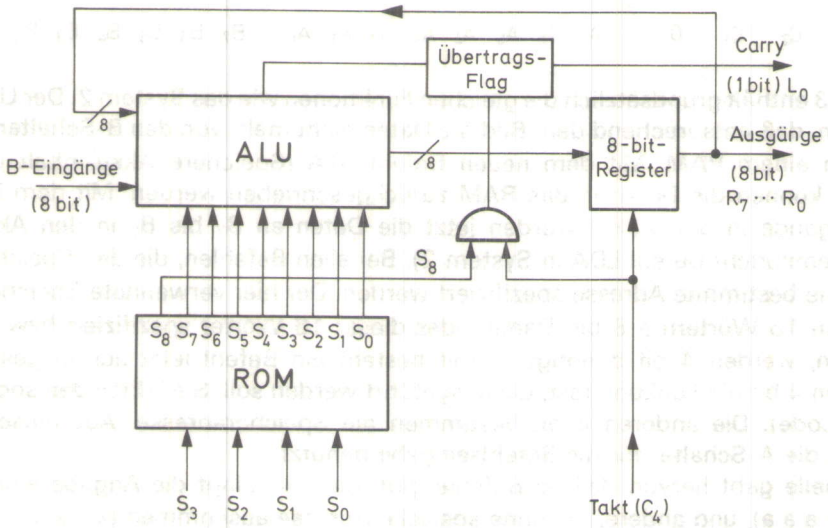


3.4 Akkumulator (SYSTEM 2)

Mit diesem Programm wird ein Akkumulator simuliert. Die in der Tabelle gezeigten Funktionen werden mit den Schaltern C_3 bis C_0 ausgewählt. Der Schalter C_4 dient als Taktschalter. Durch einmaliges Hin- und Herschieben wird ein Ergebnis in das Register übernommen und zur Anzeige gebracht. Die A-Schalter werden in diesem Beispiel nicht gebraucht, weil die A-Eingänge der im Akkumulator enthaltenen ALU mit den Ausgängen des Registers verbunden sind. Das Ergebnis bzw. der momentane Inhalt des Akkus wird wieder in R_7 bis R_0 angezeigt, ein Übertrag in L_0.



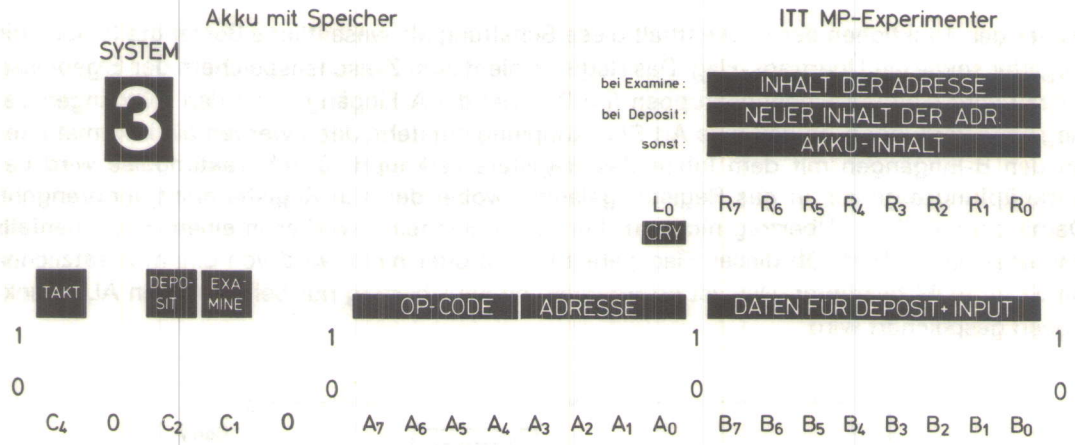
Außer den Funktionen der ALU enthält diese Schaltung als wesentliche Bestandteile noch ein Register sowie ein Übertrags-Flag. Das Register dient zum Zwischenspeichern der Ergebnisse. Hierzu wird eine der Eingangsgruppen (im Beispiel die A-Eingänge) mit den Ausgängen des Registers verbunden, so daß eine Art Rückkopplung entsteht. Jetzt werden die Informationen an den B-Eingängen mit dem Inhalt des Registers verknüpft. Durch Taktimpulse wird das Verknüpfungsergebnis in das Register geladen, wobei der alte Registerinhalt verlorengeht. Damit ein möglicher Übertrag nicht nur kurzzeitig erscheint, wird er in einem Flag ebenfalls zwischengespeichert. Ob dieses Flag getaktet wird oder nicht, wird von einem zusätzlichen bit S_8 im ROM bestimmt. Dies ist erforderlich, da ein Übertrag nur bei sinnvollen ALU-Funktionen gespeichert wird.



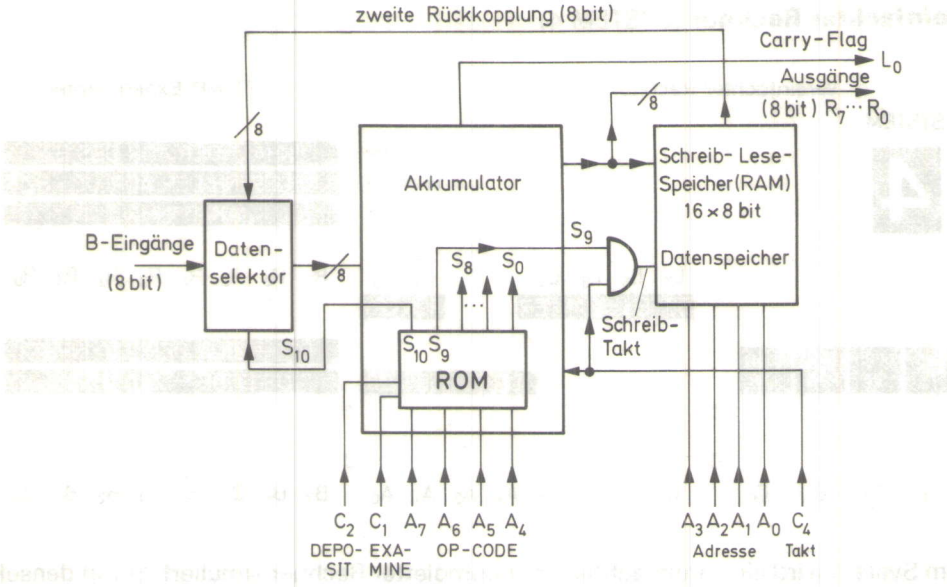
Aus der Spalte Übertrags-Flag kann entnommen werden, ob das Flag getaktet wird oder nicht. In vielen Mikroprozessoren wird dieses Flag auch bei logischen Operationen getaktet. Da hierbei aber normalerweise kein Übertrag entsteht, wird das Flag gelöscht.

U_3	U_2	U_1	U_0	Abkürzung	Funktion	Übertrags-Flag
0	0	0	0	NOP	Keine Operation	nein
0	0	0	1	SP1	Setze Akku = 1	ja
0	0	1	0	CMA	Komplementiere Akku	nein
0	0	1	1	LDA	Lade B in den Akku	nein
0	1	0	0	CLA	Lösche Akku	ja
0	1	0	1	INC	Incrementiere Akku	ja
0	1	1	0	DEC	Decrementiere Akku	ja
0	1	1	1	ADD	Addiere B in den Akku	ja
1	0	0	0	SUB	Subtrahiere B von Akku	ja
1	0	0	1	AND	Akku UND B in den Akku	ja
1	0	1	0	IOR	Akku ODER B in den Akku	ja
1	0	1	1	XOR	Akku EXCLUSIV-ODER in den Akku	ja
1	1	0	0	SM1	Setze Akku = −1	ja
1	1	0	1	–	–	nein
1	1	1	0	–	–	nein
1	1	1	1	–	–	nein

3.5 Akkumulator mit Speicher (SYSTEM 3)



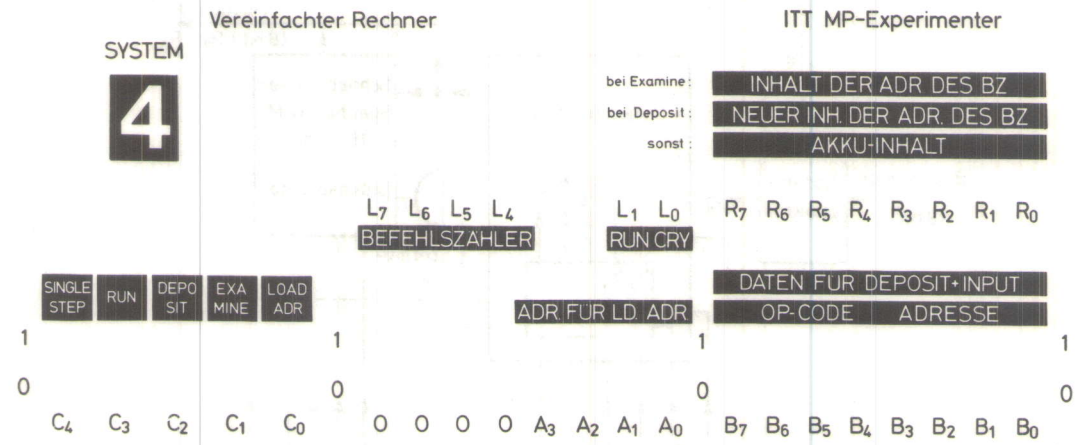
Das System 3 enthält grundsätzlich die gleichen Funktionen wie das System 2. Der Unterschied besteht darin, daß entsprechend dem Bild die Daten nicht mehr von den B-Schaltern kommen sondern von einem RAM. Mit dem neuen Befehl STA (Speichere Akku-Inhalt in Adresse a a a a ab), können die Daten in das RAM zurückgeschrieben werden. Mit dem Befehl INP (Lade B-Eingänge in den Akku) werden jetzt die Daten an B_7 bis B_0 in den Akkumulator eingelesen (entspricht Befehl LDA in System 2). Bei allen Befehlen, die den Speicher nutzen, muß jetzt eine bestimmte Adresse spezifiziert werden. Der hier verwendete Speicher hat eine Kapazität von 16 Wörtern à 8 bit. Damit jedes dieser 16 Wörter spezifiziert bzw. adressiert werden kann, werden 4 bit benötigt. Damit besteht ein Befehl jetzt aus insgesamt 8 bit. Hiervon legen 4 bit die Funktion fest, die ausgeführt werden soll. Sie bilden den sog. OP-Code (Operation-Code). Die anderen 4 bit bestimmen die Speicheradresse. Aus diesem Grunde werden jetzt die A-Schalter für die Befehlseingabe benutzt.
Aus der Tabelle geht hervor, daß es Befehle gibt, die unbedingt die Angabe einer Adresse benötigen (a a a a), und andere, die ohne spezielle Adresse auskommen (x x x x).
Bevor Befehle, die Daten aus dem Speicher unter einer bestimmten Adresse benötigen, benutzt werden können, müssen die entsprechenden Daten in den Speicher geladen werden. Das Laden einer bestimmten Speicheradresse erfolgt mit dem Schalter C_2 DEPOSIT (Laden). Wird dieser Schalter betätigt, d.h. auf 1 und dann wieder auf 0 geschaltet, werden die Daten, die an B_7 bis B_0 liegen, im Speicher bei der Adresse abgespeichert, die von den Schaltern A_3 bis A_0 spezifiziert ist.
Mit dem Schalter C_1 EXAMINE (Abfragen) kann der Speicherinhalt, der mit A_3 bis A_0 spezifizierten Adresse in R_7 bis R_0 abgebildet werden.



U_3	U_2	U_1	U_0	a_3	a_2	a_1	a_0	Abkürzung	Funktion	Übertrags-Flag
0	0	0	0	x	x	x	x	NOP	Keine Operation	nein
0	0	0	1	x	x	x	x	SP1	Setze Akku = 1	ja
0	0	1	0	x	x	x	x	CMA	Komplementiere Akku	nein
0	0	1	1	a	a	a	a	LDA	Lade Inhalt Adresse a a a a	nein
0	1	0	0	x	x	x	x	CLA	Lösche Akku	ja
0	1	0	1	x	x	x	x	INC	Incrementiere Akku	ja
0	1	1	0	x	x	x	x	DEC	Decrementiere Akku	ja
0	1	1	1	a	a	a	a	ADD	Addiere Inhalt Adresse a a a a	ja
1	0	0	0	a	a	a	a	SUB	Subtrahiere Inhalt Adresse a a a a	ja
1	0	0	1	a	a	a	a	AND	Akku UND Inhalt Adresse a a a a	ja
1	0	1	0	a	a	a	a	IOR	Akku ODER Inhalt Adresse a a a a	ja
1	0	1	1	a	a	a	a	XOR	Akku EXCLUSIV-ODER Adresse a a a a	ja
1	1	0	0	x	x	x	x	SM1	Setze Akku = −1	ja
1	1	0	1	x	x	x	x	INP	Lade B-Eingänge in den Akku	nein
1	1	1	0	a	a	a	a	STA	Speichere Akku in Adresse a a a a	nein
1	1	1	1	x	x	x	x	–	–	–

a a a a = eine Datenspeicheradresse
x x x x = „don't care"-Zustand, d.h. beliebig

3.6 Vereinfachter Rechner (SYSTEM 4)



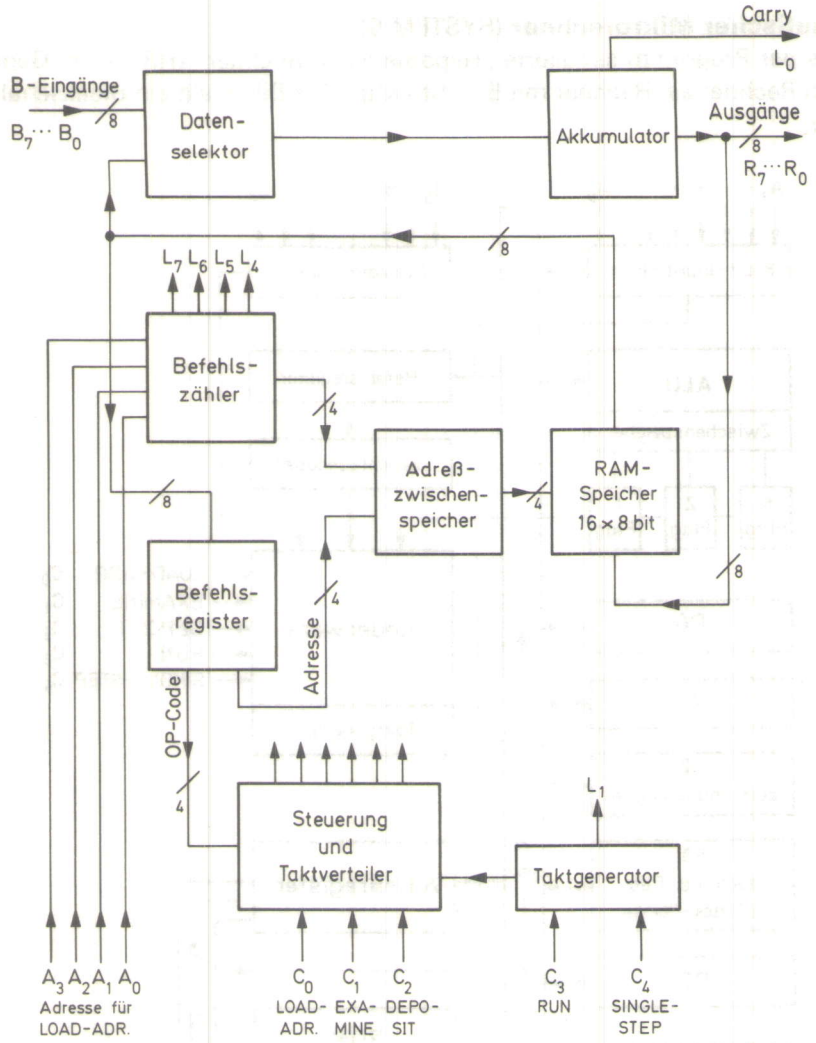
Bei diesem System wird ein vereinfachter, aber kompletter Rechner simuliert. Er hat denselben Befehlsvorrat wie der Akkumulator mit Datenspeicher im System 3. Zusätzlich hat er einen HALT-Befehl (HLT), damit der Rechner am Ende eines Programms angehalten werden kann. Im Gegensatz zum System 3 werden im 16-Wort-Speicher nicht nur Daten sondern auch das Programm angespeichert. Das Programm und die Daten werden mit dem DEPOSIT-Schalter C_2 in den Speicher geladen. Damit ein Programm automatisch ablaufen kann, enthält der simulierte Rechner einen Befehlszähler (BZ). Welche der 16 Adressen gerade selektiert ist, wird durch die LEDs L_7 bis L_4 angezeigt. Die Funktionsweise des Befehlszählers können Sie wie folgt kontrollieren:

– Stellen Sie alle Schalter außer C_4 auf 0. Bei C_4 = 1 arbeitet das System im Single-Step-Betrieb, d.h., der Befehlszähler kann mit Schalter C_3 (RUN) in Einzelschritten getaktet werden.
– In L_7 bis L_4 erscheint jetzt eine beliebige Adresse von 0 0 0 0 bis 1 1 1 1.
– Takten Sie das System mit RUN. An L_7 bis L_4 können Sie sehen, daß der Befehlszähler mit jedem Takt um einen Schritt höher springt.

Mit den Schaltern A_3 bis A_0 können Sie den Befehlszähler auf eine bestimmte Adresse laden.
Wenn Sie z.B. A_3 bis A_0 auf 0 1 1 0 einstellen und den Schalter LOAD-ADRESS (C_0) betätigen, wird der Befehlszähler auf diese Adresse gesetzt (Anzeige durch L_7 bis L_4). Wenn Sie jetzt mit dem RUN-Schalter weitertakten, zählt der Zähler von dieser Stellung weiter.
Mit den Schaltern B_7 bis B_0 können OP-Code und Adresse eingegeben werden. Hierbei ist unbedingt zu berücksichtigen, daß es sich um einen **Befehl** handelt, der in einer bestimmten Adresse abgespeichert wird. Wenn Sie z.B. B_7 bis B_0 auf 0 1 1 1 0 0 1 0 einstellen und den Schalter DEPOSIT C_2 takten, wird dieser Befehl in der Adresse abgespeichert, die gerade vom Befehlszähler selektiert ist. Der Befehl 0 1 1 1 0 0 1 0 besagt laut Tabelle: Addiere den Inhalt der Adresse 0 0 1 0 zum Inhalt des Akkus. Dies bedeutet – und das ist unbedingt zu beachten – daß bei der Befehlszählerstellung, bei der dieser Befehl eingegeben wurde, diese Rechenoperation durchgeführt wird.

Die abzuarbeitende Folge von Steuerwörtern oder Befehlen (das Programm) wird zunächst in den Programmspeicher geladen. Dabei ist natürlich die Reihenfolge der einzelnen Befehle wichtig. Die Befehle werden deshalb im Programmspeicher mit **steigenden aufeinanderfolgenden** Adressen gespeichert. Wenn dann über einen Zähler die Programmspeicheradressen automatisch erzeugt werden, erscheinen die Befehle in der richtigen Reihenfolge und können nacheinander ausgeführt werden. Bevor man allerdings ein solches System benutzen kann, muß das Programm zunächst in den Programmspeicher geladen werden.

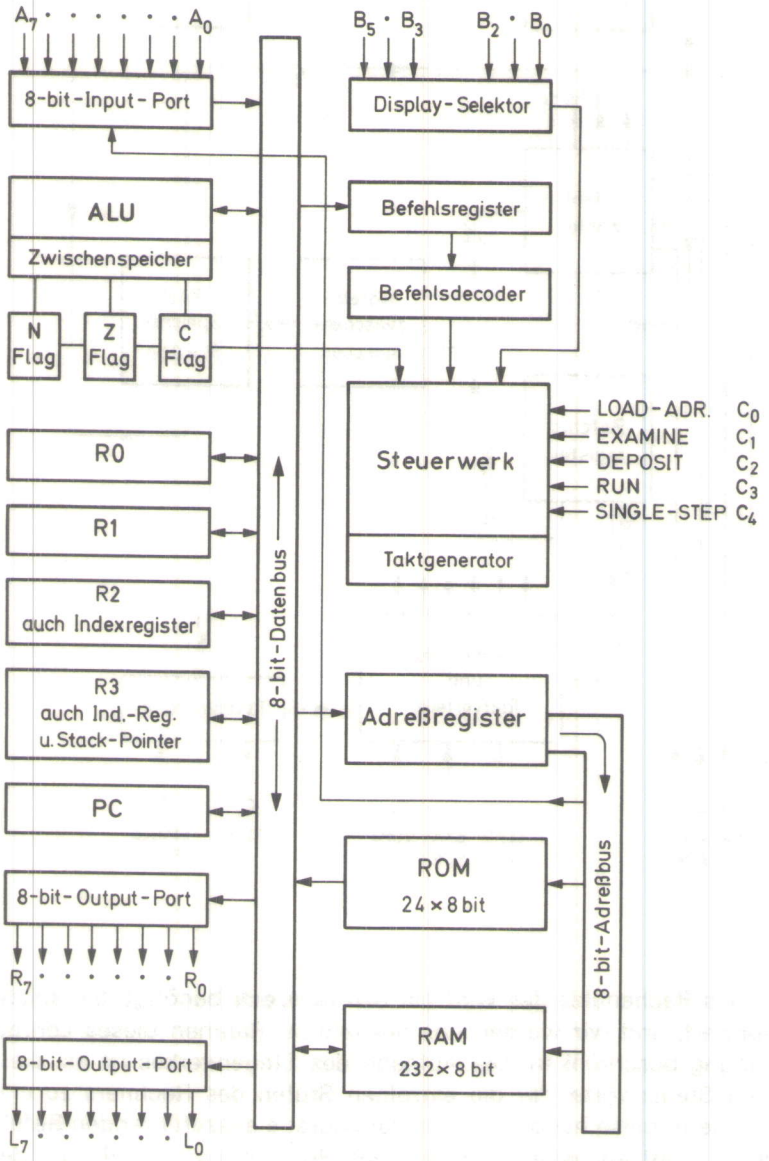
Damit der Rechner anhält, wenn das Programm abgearbeitet worden ist, muß am Ende eines Programms ein HALT-Befehl den Ablauf stoppen. Ohne diesen Befehl hätte das Programm kein Ende. Der Rechner würde auch die Daten ausführen und am Ende des Speichers wieder von vorne beginnen. Dem HALT-Befehl ist der OP-Code 1 1 1 1 zugeordnet.



Zur Steuerung des Rechenablaufes wird ein **Steuerwerk** benötigt. Ein solches Steuerwerk ist recht kompliziert, und wir werden uns deshalb im Rahmen dieses Lehrganges auf eine kurze Beschreibung beschränken. Die Aufgabe des Steuerwerkes ist es, die verschiedenen Taktimpulse und Steuerwörter für die einzelnen Stufen des Rechners zu erzeugen. Die zu erzeugenden Steuerimpulse hängen jeweils vom gerade auszuführenden Befehl ab.
Der Befehlszähler zeigt an, welcher Befehl des Programms (z.B. Nr. 17 des Programms) ausgeführt werden soll. Der Befehlszählerinhalt wird also zuerst auf die Adreßeingänge des Speichers übertragen. Der auszuführende Befehl wird jetzt aus dem Speicher geholt und im Befehlsregister zwischengespeichert. Da der Befehl im allgemeinen aus dem Operationsteil (B_7 . . . B_4) und dem Adreßteil (B_3 . . . B_0) besteht, muß der Inhalt des Befehlsregisters in Operations- und Adreßteil aufgespalten werden. Aus dem Operationsteil (Op-Code) des Befehles erkennt das Steuerwerk durch eine entsprechende Logik, ob dieser Befehl eine Adresse benötigt oder nicht. Wenn nicht, veranlaßt das Steuerwerk direkt die entsprechende Operation (z.B. Op-Code = 0 0 0 1). Wenn ja, wird der Adreßteil über den Adreßzwischenspeicher auf die Adreßeingänge des Speichers gegeben. Das unter der angesprochenen Adresse liegende Datenwort gelangt aus dem Speicher zur Ausführung der Operation in den Akkumulator. Damit ist der Befehl ausgeführt, und der Rechner kann nach Erhöhen des Befehlszählers den nächsten Befehl der Programmliste durchführen. War dieser Befehl ein HALT-Befehl (letzter Befehl jedes Programms), stoppt das Steuerwerk den Rechenablauf.

3.7 Hypothetischer Mikrorechner (SYSTEM 5)

Der ebenfalls per Programm simulierte „Hypothetische Rechner" (HR) ist im Gegensatz zum vereinfachten Rechner ein Rechner mit Bus-Struktur. Das Bild stellt ein Blockschaltbild dieses Rechners dar.



Wie das Blockschaltbild zeigt, besitzt der HR 4 Arbeitsregister (RO bis R3), von denen R2 und R3 als Indexregister für besondere Adressierung und R3 zusätzlich als Stack-Pointer benutzt wird.
Zur Signalisierung der Registerzustände sind ein Negativ-Flag (N), ein Zero-Flag (Z) und ein Carry-Flag (C) vorhanden, die als Sprung-Conditionen verwendet werden.
Der 8 bit breite Programmzähler (PC) kann $2^8 = 256$ Adressen spezifizieren. Im Adreßbereich 0 bis 24 (0 0_{16} bis 1 8_{16}) ist ein ROM untergebracht mit Betriebsprogramm. Dem Benutzer steht ein RAM mit 231 Plätzen im Adreßbereich von 25 bis 255 (1 9_{16} bis F F_{16}) zur Verfügung.
In diesen RAM-Plätzen ist bei Bedarf auch der Stack unterzubringen.
Die Adresse F F_{16} ist als Speicher-Adresse auch nicht verfügbar, da sie für den Input-Port (A-Schalter) reserviert ist, der damit wie ein Speicherplatz angesprochen werden kann.

Das Befehlsformat des HR besteht aus 8 bit, davon 4 bit (MSD) als Op-Code und 4 bit (LSD) zur Adressierung. Aufgrund des 4-bit-Op-Codes ergeben sich 16 Grundbefehle, die die Tabelle zeigt.
Es gibt 1- und 2-Byte-Befehle, wobei das 2. Byte je nach Adreßmode eine Kostante (Daten) oder eine Adresse sein kann.