

Einplatinencomputer KOOP 85.1

Robert Krenz

17.01.1990

Inhaltsverzeichnis

1	Beschreibung des KOOP 85.1	2
1.1	Spannungsversorgung	2
1.2	Der Mikroprozessor 8085	2
1.3	Der Adressenspeicher 74HCT573	2
1.4	Der Adreßdecoder 74LS138	3
1.5	Die Speicherbausteine	4
1.6	Die Parallelschnittstelle 8255	5
1.7	Der Timer 8253	6
1.8	Der USART 8251	7
1.9	Der Pegelumsetzer MAX 232	7
1.10	Der serielle RESET	7
2	Test-EPROM Version 1.0	9
2.1	Funktionsbeschreibung	9
2.1.1	Test der Parallel-Ausgabe (Kanal A)	9
2.1.2	Test der Parallel-Eingabe (Kanal C)	10
2.1.3	Test des Speicherbereichs (RAM-Test)	10
2.1.4	Test des Timers und der seriellen Schnittstelle	11
2.2	Assembler-Listing	12
2.2.1	Bemerkungen zum Assemblerprogramm TEST1.ASM	12
2.2.2	Listing	13
3	Tabellen und Skizzen	19
3.1	Schaltungsaufbau	19
3.1.1	Bauteilliste	19
3.1.2	Schaltplan	21
3.1.3	Bestückungsplan	22
3.2	Übersicht Speicherbausteine	23
3.3	8085-Befehlssatz	24
3.4	Unterprogramme	28
3.5	Steuerbefehle des Editors	32

Kapitel 1

Beschreibung des KOOP 85.1

1.1 Spannungsversorgung

Der Einplatinencomputer wird mit einer Betriebsspannung von 5 V betrieben, die aus einem stabilisierten Netzteil zugeführt werden muß. Die Stromaufnahme beträgt ca. 300 mA.

Als Schutz gegen Verpolung ist eine Diode 1N4007 vorgesehen, die in die Plus-Versorgungsleitung geschaltet ist. Für den Rechner ergibt sich somit eine Versorgungsspannung von ca. 4,3 V. Diese geringe Versorgungsspannung hat bisher noch zu keinerlei Beeinträchtigung in bezug auf die Betriebssicherheit geführt.

1.2 Der Mikroprozessor 8085

Zur Takterzeugung wird ein 4 MHz-Quarz verwendet. Durch den internen Frequenzteiler der CPU ergibt sich ein Systemtakt von 2 MHz.

Der CPU-Eingang RESET IN ist mit einer Power-On-RESET-Schaltung versehen, so daß beim Anlegen der Betriebsspannung von +5 V ein RESET ausgeführt wird. Desweiteren läßt sich mit Hilfe des Tasters S1, der an der Pfostenleiste X12 angeschlossen ist, ein Taster-RESET auslösen. Als dritte Möglichkeit zum Zurücksetzen des Einplatinencomputers ist eine Schaltung vorgesehen, mit deren Hilfe eine RESET-Auslösung über den Eingang RxD der seriellen Schnittstelle ermöglicht wird (siehe 1.10).

Die Interrupt-Eingänge TRAP, RST 7.5, RST 6.5, RST 5.5 und INTR, sowie die Anschlüsse READY, HOLD, SID und SOD sind auf die Pfostenleiste X14 gelegt. Von hier aus lassen sich diese Signale bei Bedarf verwenden.

Alle CPU-Signale werden ungepuffert auf die entsprechenden Bausteine geschaltet.

1.3 Der Adressenspeicher 74HCT573

Zur Speicherung der 8 niederwertigen Adreßsignale wird ein 8-Bit-Kippglied verwendet. Mit Hilfe des ALE-Signals der CPU werden die Adreßsignale im ersten Taktzyklus eines jeden Maschinenzyklus gespeichert und stehen für die gesamte Zeit des Maschinenzyklus an den Ausgängen zur Verfügung. Diese Ausgangssignale bilden mit den 8 höherwertigen Adreßsignalen den eigentlichen Adreßbus (A0...A15). Der Adressenspeicher wird beim Auftreten eines HLDA-Signals in den Tristate-Zustand geschaltet (HLDA = HOLD ACKNOWLEDGE, Quittierungssignal der CPU auf eine HOLD-Anforderung). Ebenso werden nach einer HOLD-Anforderung die CPU-Daten-, Adreß- und Steuerausgänge (RD, WR, IO/ \bar{M}) hochohmig.

1.4 Der Adreßdecoder 74LS138

Mit Hilfe des CPU-Steuersignals IO/\overline{M} findet eine Selektion zwischen Memory-Zugriffen ($IO/\overline{M} = \text{Low}$) und I/O-Zugriffen ($IO/\overline{M} = \text{High}$) statt. Die Adreßsignale A14 und A15 teilen sowohl den Memory-Bereich als auch den I/O-Bereich in vier gleichgroße Adreßbereiche, so daß sich folgender Zusammenhang ergibt:

IO/\overline{M}	A15	A14	Zugriff	Ausgang=L	Adreßbereich
0	0	0	MEM0	Y0	0000-3FFF
0	0	1	MEM1	Y1	4000-7FFF
0	1	0	MEM2	Y2	8000-BFFF
0	1	1	MEM3	Y3	C000-FFFF
1	0	0	(IO0)	Y4	00-3F
1	0	1	IO1	Y5	40-7F
1	1	0	IO2	Y6	80-BF
1	1	1	IO3	Y7	C0-FF

Die vier Memory-Aktivierungssignale werden mit Hilfe einer Diodenlogik zu den eigentlichen CHIP-SELECT-Signalen für die Speicherbausteine verknüpft. Insgesamt sind drei Sockel zur Aufnahme von Speicherbausteinen verschiedener Typen vorgesehen (siehe 1.5 und 3.2). Die oben angegebenen Adreßbereiche lassen sich verschiedenartig auf die drei Speichersockel aufteilen. (Ansicht von der Bestückungsseite, Stecker für die serielle Schnittstelle liegt rechts):

	U3		U2		U1
X9:	5	4	3	2	1
	MEM3	MEM2	MEM1	MEM0	
	C000	8000	4000	0000	
	⋮	⋮	⋮	⋮	
	FFFF	BFFF	7FFF	3FFF	

Zu beachten ist die Lage der drei Speicher-Sockel auf der Platine (siehe 3.1.3)!

Der Speichersockel U1 ist grundsätzlich mit MEM0 verbunden (0000-3FFF), und kann mit einem EPROM bis 16 kByte bestückt werden. Wird ein Speicher mit geringerer Speicherkapazität verwendet, so wiederholt sich seine Adreßlage innerhalb des 16 k-Bereiches entsprechend oft. Wird für den Adreßbereich 0000-7FFF ein 32 kByte-EPROM verwendet, so müssen die Stifte 2-1 an X9 miteinander verbunden werden.

Ebenso läßt sich im oberen Adreßbereich bei unbeschaltetem Stift 5 an X9 ein RAM-Baustein bis 16 kByte einsetzen (U3). Auch hier werden Speicher mit geringerer Speicherkapazität mehrfach im Adreßbereich selektiert. Durch eine Brücke 5-4 an X9 wird U3 im Adreßbereich von 8000-FFFF selektiert, und es kann ein 32 kByte-RAM verwendet werden.

Werden U1 oder U3 nicht mit 32 kByte bestückt (16 kByte oder weniger), so kann jeweils ein 16 k-Adreßbereich MEM1 oder MEM2 auf den mittleren Sockel U2 gelegt werden.

So sind je nach Anwendungsfall verschiedene Kombinationen von Speicherbausteinen möglich.

1.5 Die Speicherbausteine

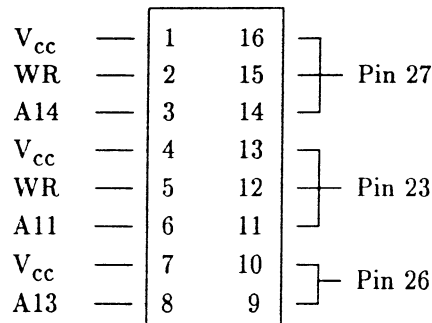
Um verschiedene Variationsmöglichkeiten bei der Bestückung mit Speicherbausteinen zu erhalten, sind U1 bis U3 als 28-polige Sockel für Bausteine nach dem Byte-wide-Konzept ausgelegt:

2 k (24-pol.)	EPROM 2716	RAM 6116 / 4116 o.ä.
4 k (24-pol.)	EPROM 2732	
8 k (28-pol.)	EPROM 2764	RAM 6164 / 6264 o.ä.
16 k (28-pol.)	EPROM 27128	
32 k (28-pol.)	EPROM 27256	RAM 51256 / 43256 o.ä.

Bei der Bestückung mit 24-poligen Speicherbausteinen müssen diese mit der Markierung nach oben aber bündig zum unteren Sockelrand eingesetzt werden (Ansicht von der Bestückungsseite, Stecker für die serielle Schnittstelle liegt rechts).

Durch Einsetzen von Brücken (Jumper) auf die Stiftleisten X2, X3 und X4 lassen sich die verschiedenen Speichertypen wählen.

Bedeutung der Stiftleisten X2, X3, X4:



Notwendige Brücken:

EPROM				
2 k	4 k	8 k	16 k	32 k
-	-	(1-16)	(1-16)	-
-	-	-	-	-
-	-	-	-	(3-14)
(4-13)	-	-	-	-
-	-	-	-	-
-	(6-11)	(6-11)	(6-11)	(6-11)
(7-10)	(7-10)	-	-	-
-	-	-	(8-9)	(8-9)

RAM		
2 k	8 k	32 k
-	-	-
-	(2-15)	(2-15)
-	-	-
-	-	-
(5-12)	-	-
-	(6-11)	(6-11)
(7-10)	(7-10)	-
-	-	(8-9)

Für die RAM-Bestückung des Speichersockels U3 muß je nach verwendetem RAM-Baustein folgende Umschaltung vorgenommen werden:

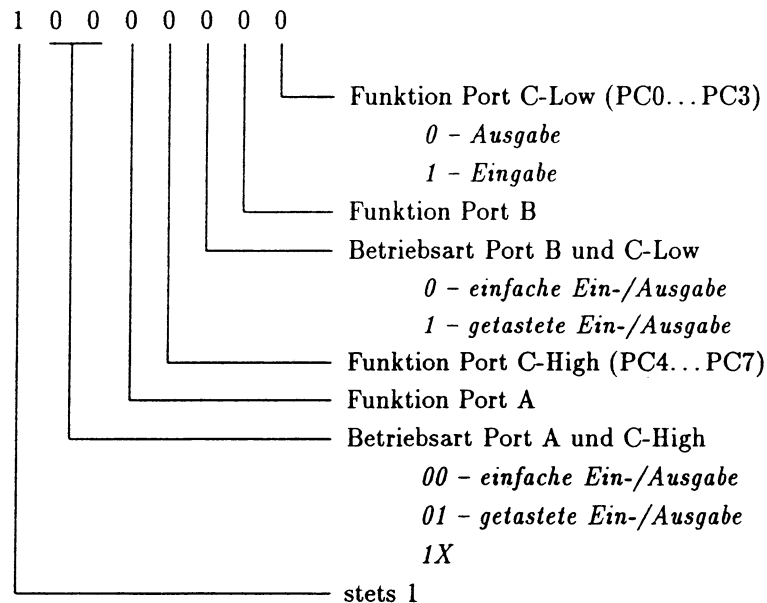
RAM-Umschaltung X16:	2/8 k	32 k			
	<table border="1"><tr><td>(1-4)</td></tr><tr><td>-</td></tr></table>	(1-4)	-	<table border="1"><tr><td>-</td></tr><tr><td>(2-3)</td></tr></table>	-
(1-4)					
-					
-					
(2-3)					

1.6 Die Parallelschnittstelle 8255

Als CHIP-SELECT-Signal für den programmierbaren Schnittstellenbaustein wird IO3 verwendet, so daß die Parallelschnittstelle im Adreßbereich von C0-FF erreichbar ist. Mit Hilfe der Adreßleitungen A0 und A1 erfolgt die Auswahl der verschiedenen Register innerhalb des Bausteins. Aufgrund der unvollständigen Dekodierung erscheinen die I/O-Register insgesamt 32 mal über den angegebenen Adreßbereich verteilt (immer im Abstand von 4). Es kann daher eine beliebige Basisadresse (modulo 4) innerhalb des angegebenen Adreßbereiches gewählt werden. So gilt z.B. für die Basisadresse E0 hex:

	<i>Lesen</i>	<i>Schreiben</i>
Kanal A	IN 0E0	OUT 0E0
Kanal B	IN 0E1	OUT 0E1
Kanal C	IN 0E2	OUT 0E2
Steuerkanal	—	OUT 0E3

Die Programmierung des 8255 erfolgt durch Ausgabe eines Betriebsartenwortes auf den Steuerkanal der Parallelschnittstelle. Dieses Betriebsartenwort ist wie folgt codiert:



Die Ausgänge der 3 8-Bit-Kanäle sind auf die zweireihige Stiftleiste X13 geführt (Ansicht von der Bestückungsseite, Stecker für die serielle Schnittstelle liegt rechts):

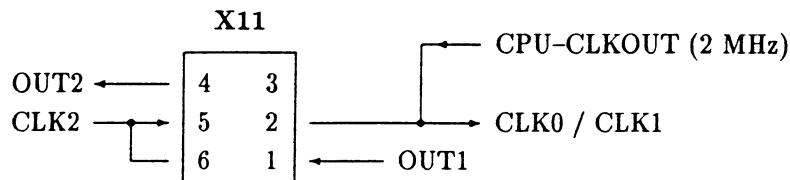
GND	—	14	13	—	PB7
PB6	—	15	12	—	PB5
PB4	—	16	11	—	PB3
PB2	—	17	10	—	PB1
PB0	—	18	9	—	PC0
PC1	—	19	8	—	PC2
PC3	—	20	7	—	PC4
PC5	—	21	6	—	PC6
PC7	—	22	5	—	PA7
PA6	—	23	4	—	PA5
PA4	—	24	3	—	PA3
PA2	—	25	2	—	PA1
PA0	—	26	1	—	V _{cc}

1.7 Der Timer 8253

Der Timer wird mit IO1 selektiert und ist daher im Adreßbereich 40...7F anzusprechen. Ebenso wie bei der Parallelschnittstelle erfolgt die Auswahl der internen Register mit Hilfe von A0 und A1. Diese unvollständige Dekodierung führt wiederum zu einer Mehrfachselektion innerhalb des angegebenen Adreßbereiches. Für die Basisadresse 40 hex ergibt sich:

	<u>Lesen</u>	<u>Schreiben</u>
Kanal 0	IN 040	OUT 040
Kanal 1	IN 041	OUT 041
Kanal 2	IN 042	OUT 042
Steuerkanal	—	OUT 043

Die Taktein- bzw. Ausgänge des Timer-Bausteins sind wie folgt mit Hilfe von X11 verschaltet (Ansicht von der Bestückungsseite, Stecker für die serielle Schnittstelle liegt rechts):



Kanal 0 dient als Baudratengenerator für den seriellen Schnittstellenbaustein 8251. Aus diesem Grund wird der Takteingang CLK0 mit dem Systemtakt von 2 MHz versorgt. So wird z.B. für eine Übertragungsrate von 9600 Bd die 16-fache Frequenz als Schiebetakt benötigt (153,6 kHz). Der Schiebetakt läßt sich mit Hilfe des Kanals 0 in der Betriebsart 3 (Rechteckgenerator) mit einem Teilfaktor von 13 erzeugen: $2 \text{ MHz}/13=153,8 \text{ kHz}$. Der Ausgang OUT0 ist mit den Takteingängen TxC und RxC des USART verbunden. Kanal 1 kann frei programmiert werden. Die Eingangsfrequenz beträgt ebenfalls 2 MHz. Das Ausgangssignal OUT1 (X11, Pin 1) kann u.U. zur Interruptauslösung der CPU benutzt werden. Der Kanal 2 erhält als Taktsignal wahlweise den Systemtakt von 2 MHz (Brücke 5-2) oder das Ausgangssignal OUT1 (Brücke 6-2), so daß die Kanäle 1 und 2 auch kaskadiert werden können. Das Ausgangssignal OUT2 steht an X11, Pin 4 zur weiteren Verwendung zur Verfügung. Die GATE-Eingänge aller 3 Kanäle sind inaktiv geschaltet (H-Pegel).

1.8 Der USART 8251

Der serielle Schnittstellenbaustein 8251 wird mit IO2 selektiert und ist daher im Adreßbereich 80-BF hex anzusprechen. Zur Auswahl der internen Register dient zusätzlich die Adreßleitung A0. Die unvollständige Dekodierung führt wiederum zur Mehrfachselektion im angegebenen Adreßbereich. Für die Basisadresse B0 ergibt sich:

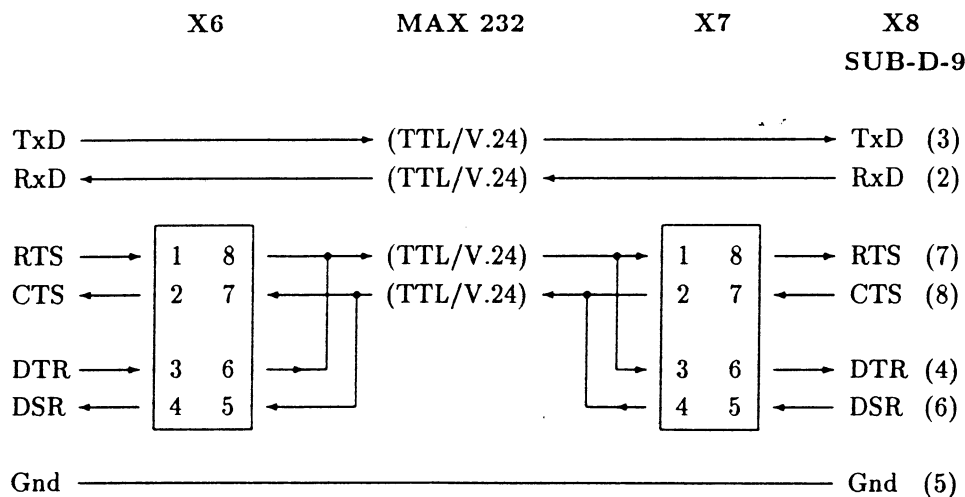
Register	Adresse
Datenregister (TxBuffer, RxBuffer)	B0
Controlregister (Betriebsart, Kommando, Status)	B1

Die Anschlüsse zur seriellen Übertragung werden dem Pegelwandler MAX 232 über die Stiftleiste X6 zugeführt (siehe 1.9). Die USART-Signale TxREADY, TxEMPTY und RxREADY sind auf die Stiftleiste X14 gelegt und können von hier aus u.U. zur Interruptauslösung der CPU weiterverwendet werden.

1.9 Der Pegelumsetzer MAX 232

Der Baustein MAX 232 dient zur Pegelumsetzung zwischen TTL- und V.24-Pegeln. Die Elektrolytkondensatoren C7, C10, C13 und C14 dienen zur Ladungsspeicherung für die Erzeugung der V.24-Pegel +12 V und -12 V durch den Pegelumsetzer MAX 232.

Die Signale zur seriellen Übertragung sind wie folgt mit Hilfe von X6, X7, dem Pegelwandler MAX 232 und dem seriellen Anschlußstecker X8 verschaltet (Ansicht von der Bestückungsseite, Stecker für die serielle Schnittstelle liegt rechts):



1.10 Der serielle RESET

Die monostabile Kippstufe 74LS122 kann zur zusätzlichen RESET-Auslösung eingesetzt werden. Die RESET-Auslösung läßt sich mit Hilfe des seriellen Empfangssignals RxD steuern. Zur Triggerung der Kippstufe werden die Signale RxD und der Systemtakt CLKOUT (2 MHz) verwendet. Im Ruhezustand der seriellen Leitung liegt der RxD-Pegel auf High, so daß die Kippstufe ständig durch den Systemtakt nachgetriggert wird (alle 500 ns). Während einer seriellen Übertragung über RxD wird der Pegel auf dieser

Leitung längstens für 10 Bitzeiten auf Low geschaltet (1 Startbit, 8 Datenbits, 1 Paritybit). Spätestens bei der Übertragung der Stopbits geht der RxD-Pegel wieder auf High, so daß die Kippstufe durch den Systemtakt wieder nachgetriggert wird.

Wird eine Übertragungsrate von 9600 Bd gewählt, so bedeutet dieses, daß die Kippstufe spätestens nach $10 \cdot 1/9600 \text{ s} = 1 \text{ ms}$ nachgetriggert wird. Da die Impulszeit der Kippstufe auf ca. 2 ms festgelegt ist (C8, R_{In}), bleibt bei einer ordnungsgemäßen Übertragung das Ausgangssignal Q ständig auf H-Pegel, und es wird kein RESET ausgelöst. Wird die RxD-Leitung dagegen länger als ca. 2 ms auf L-Pegel geschaltet, so erfolgt bei geschlossener Brücke X10 eine RESET-Auslösung, die erst wieder aufgehoben wird, wenn RxD wieder H-Pegel erlangt. Die Erzeugung eines genügend langen RxD-Low-Pegels könnte z.B. gezielt durch den angeschlossenen seriellen Partner erfolgen, wenn dieser seine Sende-Übertragungsrate auf 1200 Bd einstellt und das Datenbyte 00 sendet.

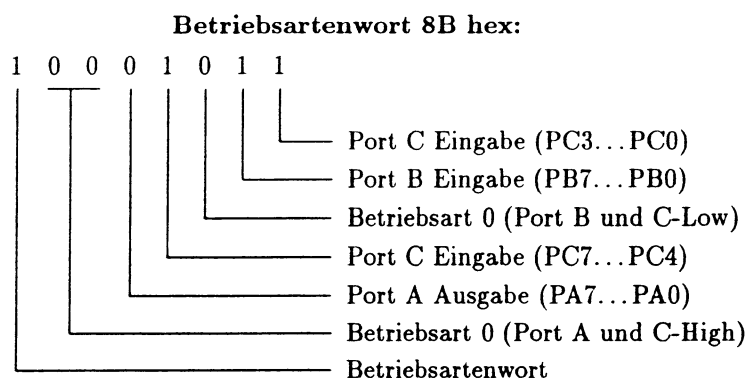
Kapitel 2

Test-EPROM Version 1.0

2.1 Funktionsbeschreibung

2.1.1 Test der Parallel-Ausgabe (Kanal A)

Zu Beginn des Testprogramms wird die Parallelschnittstelle 8255 wie folgt initialisiert:



Die Ausgabe des Betriebsartenwortes erfolgt über den Steuerkanal des 8255 (hier Portadresse 0E3 hex). Anschließend erfolgt die Ausgabe eines Bitmusters im Abstand von ca. 1 s. Als Bitmuster wird eine links-rotierende 1-Bit-Ausgabe verwendet (Laufflicht) :

1.Sekunde:	0 0 0 0 0 0 0 1
2.Sekunde:	0 0 0 0 0 0 1 0
3.Sekunde:	0 0 0 0 0 1 0 0
⋮	⋮
8.Sekunde:	1 0 0 0 0 0 0 0

Da innerhalb der 1-Sekunden-Zeitschleife das Bitmuster immer wieder in den Ausgabekanal geschrieben wird, ist eine Kontrolle des CHIP-SELECT-Signals und des WRITE-Signals an der Parallelschnittstelle mit Hilfe des Oszilloskops möglich.

Der Test des Ausgabekanal A ist nach der Ausgabe von insgesamt 8 Bitmustern beendet. Es erfolgt automatisch der Test des Eingabekanal C.

2.1.2 Test der Parallel-Eingabe (Kanal C)

Der Eingabekanal C wird gelesen und der Eingabewert zum schon getesteten Ausgabekanal A ausgegeben. Damit können alle Bits der Eingabe einzeln oder kombiniert getestet werden. Das Einlesen und Ausgeben der Eingabewerte erfolgt periodisch mit Hilfe einer Programmschleife, so daß auch hier bei Fehlfunktion eine Überprüfung der entsprechenden Signale am Baustein 8255 vorgenommen werden kann. Der Test des Eingabekanals C wird beendet, wenn folgender Eingabwert eingestellt wird:

1 0 0 0 0 0 0 0 80 hex \Rightarrow Ende Eingabe-Test

Es erfolgt ein Sprung zur Testroutine des Speicherbereiches.

2.1.3 Test des Speicherbereichs (RAM-Test)

Der Speicherbereich von 64 kByte wird in 4 Blöcken zu je 16 kByte getestet. Die Auswahl des zu testenden Speicherblocks wird mit Hilfe der Eingabebits B3...B0 vorgenommen. Dabei muß das höchstwertige Eingabebit B7 immer den Wert 1 beibehalten:

<i>Eingabe</i>	<i>getesteter Speicherbereich</i>
1 0 0 0 0 0 0 1	0000...3FFF
1 0 0 0 0 0 1 0	4000...7FFF
1 0 0 0 0 1 0 0	8000...BFFF
1 0 0 0 1 0 0 0	C000...FFFF

Beim Test eines Speicherbereiches wird für jede der 16384 Speicherstellen folgende Überprüfung vorgenommen:

- Inhalt des Speicherplatzes lesen und in CPU zwischenspeichern
- 55 hex in den Speicherplatz schreiben
- vergleichen, ob Inhalt 55 hex
- AA hex in den Speicherplatz schreiben
- vergleichen, ob Inhalt AA hex
- gespeicherten alten Inhalt zurückschreiben

Während des Speichertests erfolgt folgende Anzeige über den Ausgabekanal A:

<i>Ausgabe</i>	<i>Bedeutung</i>
1 0 0 0 0 0 0 0	kein RAM vorhanden (EPROM oder RAM unbestückt)
1 1 1 1 1 1 1 1	RAM vorhanden
0 0 0 0 0 0 0 0	Test des Speicherbereichs beendet

Nach dem durchgeführten Speichertest erfolgt noch eine kurze Zeitverzögerung von ca. 0,8 s, während der die Ausgabeanzeige erhalten bleibt. Danach wird die Ausgabe auf LOW geschaltet.

Zu beachten ist, daß aufgrund der nicht-vollständigen Decodierung der Speicherbereiche alle Speicherbausteine mit einer geringeren Speicherkapazität als 16 kByte mehrfach innerhalb des selektierten Speicherblocks angesprochen werden. So wird z.B. ein 2 kByte RAM-Baustein im gesamten 16 kByte Speicherbereich als RAM erkannt und angezeigt.

Wurde ein intakter RAM-Bereich erkannt, so wird der Stack-Pointer auf die höchste RAM-Adresse in diesem Speicherbereich eingestellt, so daß für die nachfolgenden Test-Routinen auch die Verwendung von Unterprogrammen möglich ist.

Wird B7 auf 0 geschaltet, so wird der RAM-Test beendet und es erfolgt ein Sprung zu den Test-Routinen für den Timer-Baustein und die serielle Schnittstelle. Da für diese Test-Routinen auch Unterprogramme verwendet werden, muß vor diesem Test ein intaktes RAM erkannt worden sein:

0 0 0 0 0 0 0 0 00 hex \Rightarrow Ende Speicher-Test

2.1.4 Test des Timers und der seriellen Schnittstelle

Innerhalb des folgenden Testabschnitts sind folgende Umschaltungen möglich:

<i>Eingabe</i>	<i>Test-Routine</i>
0 0 0 0 0 0 0 1	Periodische Timerinitialisierung
0 0 0 0 0 0 1 0	Ausgabe eines Textes zum Terminal
0 0 0 0 0 1 0 0	Tastaturabfrage und Bildschirmausgabe
0 0 0 0 1 0 0 0	Software-RESET

Periodische Timerinitialisierung

Der Timer-Kanal 0 wird als symmetrischer Rechteckgenerator initialisiert und mit der Zählkonstanten 13 dez geladen. Damit wird der systemtakt von 2 MHz durch 13 dividiert und am ausgang des Timerkanals OUT0 steht ein Rechtecksignal mit einer frequenz von ca 154 kHz zur Verfügung. Dieser Takt wird auch zur Versorgung der seriellen Schnittstelle benötigt. Die Initialisierung des Timers wird periodisch wiederholt, solange mit Hilfe der Paralleleingabe dieser Testabschnitt ausgewählt bleibt. Damit ist eine meßtechnische Überprüfung der entsprechenden Bausteinsignale des Timers 8253 möglich.

Ausgabe eines Textes zum Terminal

Die serielle Schnittstelle wird auf folgende Betriebsart initialisiert:

- 9600 Baud (TxC und RxC = 154 kHz, Teiler 16 \rightarrow 9625 Hz Schiebetakt)
- No Parity, 8 Bit/Char, 2 StopBits, asynchron

Anschließend erfolgt die ständige Ausgabe des folgenden Textes zum Terminal, solange mit Hilfe der Paralleleingabe dieser Testabschnitt ausgewählt bleibt:

TERMINAL-TEST:

THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS, 0123456789

Tastaturabfrage und Bildschirmausgabe (Terminal)

Innerhalb dieser Routine wird der serielle Eingabekanal und damit die Tastatur des Terminals abgefragt und ein gelesenes Zeichen zum Terminalbildschirm ausgegeben. Damit läßt sich die serielle Übertragung zwischen dem Einplatinencomputer und einem angeschlossenen Terminal überprüfen.

Software-RESET (starte Test-EPROM erneut)

Wird mit Hilfe der Paralleleingabe auf diesen Testabschnitt umgeschaltet, so erfolgt ein Sprung zur Adresse 0000 hex, so daß das Testprogramm erneut gestartet wird.

2.2 Assembler-Listing

2.2.1 Bemerkungen zum Assemblerprogramm TEST1.ASM

Aufgrund der I/O-Adreßdekodierung (74LS138) des KOOP 85.1 werden die I/O-Bausteine in folgenden Adreßbereichen angesprochen:

<i>Selektierter I/O-Baustein</i>	<i>Adreßbereich</i>
—	00...3F
Timer-Baustein 8253	40...7F
Serieller Schnittstellenbaustein 8251	80...BF
Paralleler Schnittstellenbaustein 8255	C0...FF

Als Basisadresse zur Dekodierung der I/O-Bausteine werden hier folgende Adressen gewählt, um eine Kompatibilität zum MFA-MC-System zu erreichen:

<i>I/O-Baustein</i>	<i>Adreßbereich</i>
Timer	4X
Serielle Schnittstelle	BX
Parallele Schnittstelle	EX

Die Dekodierung einzelner Register innerhalb der I/O-Bausteine erfolgt mit Hilfe der Adreßleitungen A0 (und) A1, so daß die verschiedenen I/O-Register wie folgt angesprochen werden können:

<i>Timer</i>	<i>I/O-Adresse</i>
Kanal 0	40
Kanal 1	41
Kanal 2	42
Steuerkanal	43

<i>Serielle Schnittstelle</i>	<i>I/O-Adresse</i>
Datenregister (TxBuffer, RxBuffer)	B0
Serielle Schnittstelle	B1

<i>Parallele Schnittstelle</i>	<i>I/O-Adresse</i>
Kanal 0	E0
Kanal 1	E1
Kanal 2	E2
Steuerkanal	E3

Zur Assemblierung eines Programmes, daß später als lauffähiges Maschinenprogramm innerhalb eines Adreßbereiches eingesetzt werden soll, in dem das MFA-MC-System seinen EPROM-Bereich liegen hat,

muß der durch den Assembler erzeugte Maschinencode nicht in diesen Original-EPROM-Adreßbereich abgelegt werden, sondern innerhalb eines anderen Adreßbereiches im RAM-Arbeitsspeicher. Von dort aus läßt sich das Programm in ein EPROM programmieren.

Um dennoch die richtigen absoluten Adressen für das Maschinenprogramm zu erzeugen, muß bei allen Sprungbefehlen und allen Unterprogrammaufrufen ein entsprechender OFFSET abgezogen werden. Mit Hilfe dieses Tricks wird der assemblierte Maschinencode zwar an die durch die ORG-Anweisung definierte Adresse innerhalb des RAM-Arbeitsspeichers abgelegt, wobei aber die innerhalb des Maschinenprogramms notwendigen absoluten Adressen den tatsächlichen späteren Zieladressen entsprechen.

Aufgrund dieses beschriebenen Assemblierens mit OFFSET EQU 0C000 ergibt sich im Assembler-Listing ein Maschinencode, der im Arbeitsspeicher ab Adresse C000 abgelegt ist. Der lauffähige Maschinencode arbeitet dagegen im Original-Zielbereich 0000.

2.2.2 Listing

Im folgenden ist das Assemblerlisting des TEST-EPROM's Version 1.0 dargestellt.

```

0000      ;*****
0000      ;*
0000      ;*   T E S T - E P R O M ( V 1 . 0 )
0000      ;*   FÜR EINPLATINENCOMPUTER KOOP 85.1
0000      ;*   (C) DEZ. 1989, ROBERT KRENZ, 4716 OLFEN
0000      ;*
0000      ;*****
0000
0000      ;---  PARALLEL-SCHNITTSTELLE  -----
0000
0000      PIO    EQU    0E0    ; I/O-ADRESSE PARALLEL-I/O 8255
0000      PIOA   EQU    PIO+0  ; PARALLEL-I/O (KANAL A)
0000      PIOB   EQU    PIO+1  ; (KANAL B)
0000      PIOC   EQU    PIO+2  ; (KANAL C)
0000      PIOST  EQU    PIO+3  ; (STEUERKANAL)
0000
0000      ;---  TIMER  -----
0000
0000      TIMER  EQU    040    ; I/O-ADRESSE TIMER8053
0000      TIME0  EQU    TIMER+0 ; TIMERKANAL 0
0000      TIME1  EQU    TIMER+1 ; TIMERKANAL 1
0000      TIME2  EQU    TIMER+2 ; TIMERKANAL 2
0000      TIMEST EQU    TIMER+3 ; STEUERKANAL
0000
0000      ;---  SERIELLE SCHNITTSTELLE  -----
0000
0000      SER    EQU    0B0    ; I/O-ADRESSE USART 8251
0000      SERDAT EQU    SER+0  ; SER.-I/O-DATA (SENDER/EMPF.DATEN)
0000      SERCTR EQU    SER+1  ; SER.I/O-CONTROL (BETRIEBSART,
0000      ; KOMMANDO, STATUS)
0000
0000      ;---  KONSTANTENDEFINITIONEN  -----
0000
0000      OFF    EQU    0C000  ; OFFSET FÜR ASSEMBLER
0000
0000      ;---  PROGRAMMBEGIN  -----
0000

```

```

0000                                     ; <-----
0000                                     ; <-- TEST-EPROM
0000      ORG 0000+OFF                   ; <--- PROGRAMM-ANFANG
C000                                     ; <-- BEI ADRESSE 0000
C000                                     ; <-----
C000
C000
C000      ;--- 1. TEST DER PARALLEL-AUSGABE (KANAL A) -----
C000      3E 8B      TEST:  MVI A,8B      ; BETRIEBSARTENWORT:
C002      D3 E3      OUT PIOST           ; 1000 1011 (BETRIEBSART 0 FÜR A,B,C)
C004                                     ; A: AUSGABE, B UND C: EINGABE
C004      16 01      MVI D,01           ; ANFANGSWERT FÜR AUSGABE: 0000 0001
C006      01 97CD   OUT1:  LXI B,0CD97   ; ZEITSCHLEIFE (CA. 1 S)
C009      7A        OUT2:  MOV A,D       ; AUSGABEWERT AN KANAL A AUSGEBEN
C00A      D3 E0      OUT P10A
C00C      0B        DCX B               ; ZÄHLER DEKREMENTIEREN
C00D      78        MOV A,B
C00E      B1        ORA C
C00F      C2 0900   JNZ OUT2-OFF
C012
C012      7A        MOV A,D             ; AUSGABEWERT UM 1 NACH LINKS
C013      07        RLC                 ; ROTIEREN
C014      57        MOV D,A            ; UND IN D ZWISCHENSPEICHERN
C015      D2 0600   JNC OUT1-OFF       ; INSGESAMT 8-MAL AUSGEBEN
C018
C018
C018      ;--- 2. TEST DES EINGABE-KANALS C -----
C018
C018      DB E2      INTST: IN PIOC      ; EINGABEWERT VON KANAL C HOLEN
C01A      FE 80      CPI 80            ; WENN EINGABE 80, DANN RAM-TEST
C01C      CA 2400   JZ RAMTST-OFF
C01F      D3 E0      OUT P10A         ; AUSGABE ZUM KANAL A
C021      C3 1800   JMP INST-OFF
C024
C024
C024      ;--- 3. RAM-TEST -----
C024
C024      3E 00      RAMTST: MVI A,00    ; AUSGABE AUSSCHALTEN
C026      D3 E0      OIT P10A
C028
C028      DB E2      IN PIOC           ; EINGABE ABFRAGEN, ENTPRELLUNG
C02A      4F        MOV C,A           ; EINGABEWERT ZWISCHENSPEICHERN
C02B      06 80      MVI B,80         ; ENTPRELL-ZEITSCHLEIFE
C02D      05        PRELL: DCR B
C02E      C2 2D00   JNZ PRELL-OFF
C031      DB E2      IN PIOC           ; NOCHMALS EINGABE LESEN
C033      B9        CMP C             ; UND VERGLEICHEN
C034      C2 2400   JNZ RAMTST        ; WENN UNGLEICH, NOCHMALS ABFRAGEN
C037      E6 80      ANI 80           ; B7 UNGLEICH 0, DANN TEST DES TIMERS
C039      CA 9200   JZ SERTST-OFF     ; UND DER SERIELLEN SCHNITTSTELLE
C03C
C03C      DB E2      IN PIOC           ; EINGABE ABFRAGEN
C03E      E6 7F      ANI 7F           ; B7 AUSMASKIEREN

```

```

C040 21 0000      LXI H,0000      ; 01 --> TESTE AB ADRESSE 0000
C043 FE 01      CPI 01
C045 CA 6000     JZ WRAM-OFF
C048
C048 26 40      MVI H,40        ; 02 --> TESTE AB ADRESSE 4000
C04A FE 02      CPI 02
C04C CA 6000     JZ WRAM-OFF
C04F
C04F 26 80      MVI H,80        ; 04 --> TESTE AB ADRESSE 8000
C051 FE 04      CPI 04
C053 CA 6000     JZ WRAM-OFF
C056
C056 26 C0      MVI H,0C0       ; 08 --> TESTE AB ADRESSE C000
C058 FE 08      CPI 08
C05A 6000       JZ WRAM-OFF
C05D C3 2400     JMP RAMTST-OFF
C060
C060 16 FF      WRAM: MVI D,OFF      ; OK-AUSGABE: 1111 1111
C062 01 0040    LXI B,04000     ; SPEICHERBLOCK-LÄNGE = 16 K
C065
C065 5E      TSTNXT: MOV E,M      ; ZEICHEN AUS SPEICHER IN E SPEICHERN
C066 3E 55      MVI A,055      ;      0 1 0 1 0 1 0 1
C068 77      MOV M,A      ;      IN SPEICHER SCHREIBEN
C069 BE      CMP B      ;      IST DRIN ?
C06A CA 6F00   JZ OK1-OFF     ;      JA: DANN WEITER
C06D 16 80     MVI D,80      ;      NEIN: ERROR-AUSGABE: 1000 0000
C06F 2F      OK1:  CMA      ;      1 0 1 0 1 0 1 0
C070 77      MOV M,A      ;      IN SPEICHER SCHREIBEN
C071 BE      CMP M      ;      IST DRIN ?
C072 CA 7700   JZ OK2-OFF     ;      JA: DANN WEITER
C075 16 80     MVI D,80      ;      NEIN: ERROR-AUSGABE: 1000 0000
C077 73      OK2:  MOV M,E      ;      ALTEN WERT WIEDER IN SPEICHER ZURÜCK
C078
C078 7A      MOV A,D      ;      AUSGABE (OK ODER ERROR)
C079 D3 E0     OUT P10A
C07B
C07B 23      INX H      ;      NÄCHSTEN SPEICHERPLATZ
C07C 0B      DCX B      ;      SCHON 16 K GETESTET ?
C07D 78      MOV A,B
C07E B1      ORA C
C07F C2 6500   JNZ TSTNXT-OFF ;      NEIN: DANN TESTE NÄCHSTE ADRESSE
COB2
COB2 7A      MOV A,D      ;      STACK-POINTER AUF RAM-TOP SETZEN,
C083 FE FF     CPI OFF      ;      WENN RAM OK
C085 C2 8900   JNZ ZEITR-OFF
C088 F9      SPHL
C089
C089 0B      ZEITR: DCX B      ;      ZEITVERZÖGERUNG FÜR LETZTE AUSGABE
C08A 78      MOV A,B
C08B B1      ORA C
C08C C2 8900   JNZ ZEITR-OFF
C08F
C08F C3 2400   JMP RAMTST-OFF
C092

```



```

C092          ;--- 4. TEST DES TIMERS UND DER SERIELLEN SCHNITTSTELLE
C092
C092 CD 1F01  SERTST: CALL INITS-OFF ; INITIALISIERUNG SERIELLE SCHNITTST.
C095
C095 DB E2    SERT1:  IN PIOC          ; ABFRAGE: WAS SOLL GETESTET WERDEN ?
C097
C097 FE 01    CPI 01                ; 01 --> TESTE TIMER-DEKODIERUNG
C099 CC AE00  CZ TIMEDEK-OFF        ;          (KANAL 0 = 154 KHZ)
C09C
C09C FE 02    CPI 02                ; 02 --> TESTE BILDSCHIRMAUSGABE
C09E CC B800  CZ BLDOUT-OFF
COA1
COA1 FE 04    CPI 04                ; 04 --> TESTE TASTATUR MIT BILDSCHIRM
COA3 CC 1401  CZ TAST-OFF
COA6
COA6 FE 08    CPI 08                ; 08 --> GESAMTEN TEST VON VORN
COA8 CA 0000  JZ TEST-OFF
COAB
COAB C3 9500  JMP SERT1-OFF         ; SONST --> "WAS SOLL ICH TUN ?"
COAE
COAE
COAE          ;--- TESTE TIMER-DEKODIERUNG -----
COAE
COAE 3E 16    TIMDEK: MVI A,16        ; TIMER 0 INITIALISIEREN
COB0 D3 43    OUT TIMEST             ; STEUERWORT: 00 01 011 0
COB2                                     ; (KANAL 0, NUR L-BYTE LADEN, MODE 3
COB2                                     ; BINÄR-ZÄHLER)
COB2 3E 0D    MVI A, 0D             ; L-BYTE = 13 --> OUT0 = 154 KHZ
COB4 D3 40    OUT TIME0
COB6 AF       XRA A                 ; RETURN MIT A = 00
COB7 C9       RET
COB8
COB8
COB8          ;--- BILDSCHIRMAUSGABE -----
COB8
COB8 21 C500  BLDOUT: LXI H,TXTOUT-OFF ; TEXTZEIGER AUF TEXTANFANG
COBB 7E      NXTOUT: MOV A,M          ;          TESTZEICHEN HOLEN
COBC B7      ORA A                   ;          WENN ENDE-ZEICHEN
COBD C8      RZ                      ;          DANN RETURN
COBE CD 3B01  CALL SEROUT-OFF        ;          ZEICHEN ZUM TERMINAL
COC1 23      INX H                   ;          TEXTZEIGER ERHÖHEN
COC2 C3 BB00  JMP NXTOUT-OFF        ;          UND NÄCHSTES ZEICHEN HOLEN
COC5
COC5 0D0A5445  TXTOUT: DB 0D,0A,'TERMINAL-TEST',0D,0A
COC9 524D494E
COC0D 414C2D54
COD1 4553543A
COD5 0D0A
COD7 20544845  DB ' THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS,'
CODB 20515549
CODF 434B2042
COE3 524F574E
COE7 20464F59
COEB 204A554D

```

```

COEF 50454420
COF3 4F564552
COF7 20544845
COFB 204C415A
COFF 5920444F
C103 47532C
C106 20303132      DB ' 0123456789',0D,0A,0
C10A 33343536
C10E 3738390D
C112 0A00
C114      ;--- TASTATUR ABFRAGEN UND BILDSCHIRMAUSGABE ---
C114
C114 DB B1      TAST:  IN SERCTR      ; STATUS LESEN
C116 E6 02      ANI 02      ; RXRDY ?
C118 C8         RZ      ; NEIN: DANN RETURN
C119
C119 DB B0      IN SERDAT      ; JA: DANN ZEICHEN LESEN
C11B CD 3B01    CALL SEROUT-OFF ;      UND ZUM TERMINAL AUSGEBEN
C11E C9         RET
C11F
C11F
C11F      ;--- UNTERPROGRAMME FÜR SERIELLE SCHNITTSTELLE
C11F
C11F      ;--- TIMER UND SERIELLE SCHNITTSTELLE INITIALISIEREN
C11F
C11F 3E 16      INITS: MVI A,16      ; TIMER 0 INITIALISIEREN
C121 D3 43      OUT TIMEST      ; STEUERWORT 00 01 011 0
C123           ; (KANAL 0, NUR L-BYTE LADEN, MODE 3
C123           ; BINÄR-ZÄHLER)
C123 3E 0D      MVI A,0D      ; L-BYTE = 13 --> OUTO = 154 KHZ
C125 D3 40      OUT TIMEO
C127
C127 AF         XRA A      ; 3-MAL 00 AN SER. CONTROL
C128 D3 B1      OUT SERCTR
C12A D3 B1      OUT SERCTR
C12C D3 B1      OUT SERCTR
C12E
C12E 3E 40      MVI A,40      ; UMSCHALTEN AUF BETRIEBSARTENREGISTER
C130 D3 B1      OUT SERCTR
C132 3E CE      MVI A,OCE      ; BETRIEBSARTENWORT: 11 X0 11 10
C134 D3 B1      OUT SERCTR      ; 2 STOPBITS, NO PARITY, 8 BIT, DIV 16
C136 3E 37      MVI A,37      ; KOMMANDOWORT: 01 1 1 0 111
C138 D3 B1      OUT SERCTR      ; STANDARD, RTS, ERROR-RESET, NORMAL,
C13A           ; RECEIVER, DTR, TRANSMITTER
C13A C9         RET      ; RETURN
C13B
C13B
C13B      ;--- SERIELLE AUSGABE ZUM BILDSCHIRM -----
C13B
C13B F5         SEROUT: PUSH PSW      ; AUSGABEWERT RETTEN
C13C DB B1      STATO:  IN SERCTR      ;      SATUS ABFRAGEN (TXRDY)
C13E E6 01      ANI 01      ;      WARTE BIS TXRDY = 1
C140 CA 3C01    JZ STATO-OFF
C143 F1         POP PSW      ; AUSGABEWERT ZURÜCK

```

2.2. ASSEMBLER-LISTING

18

```
C144 D3 B0      OUT SERDAT      ; UND AUSGEBEN
C146 C9        RET          ; RETURN
C147
C147          ZZZ:          ; ENDE-LABEL
C147
```

Kapitel 3

Tabellen und Skizzen

3.1 Schaltungsaufbau

3.1.1 Bauteilliste

Integrierte Schaltungen

U1	EPROM	27128
U2	EPROM	27128
U3	RAM	51256
U4	Adreßlatch	74HCT573
U5	CPU	8085AP δ_1
U6	PIO	82C55A
U7	Timer	8253 γ
U8	Monoflop	74LS122
U9	Pegelumsetzer	MAX232
U10	Decoder	74LS138
U11	USART	8251P γ

Widerstände

R1A...R1F	Widerstand	6 * 4,7 k Ω
R2, R3	Widerstand	2 * 4,7 k Ω
R4, R5, R7	Widerstand	3 * 2,7 k Ω
R6	Widerstand	1 * 1 k Ω
R8	Widerstand	1 * 47 k Ω
R9, R10	Widerstand	2 * 10 k Ω

Kondensatoren

C1, C2, C3	Keramik	3 * 10 nF
C4, C11, C12	Keramik	3 * 100 nF
C6	Keramik	1 * 27 pF
C8	Keramik	1 * 470 nF
C5, C9	Tantal	2 * 4,7 μ F
C7, C10, C13, C14	Tantal	4 * 22 μ F

Dioden

D1, D2, D3, D4, D5	Germanium oder Schottky	BAT 43	0,1A/20V	750 x 5
D6	Silizium 1N4148	707 ✓	2x	
D7	Silizium 1N4007			

Quarze

QZ1 Quarz 4 MHz 1,75

IC-Sockel

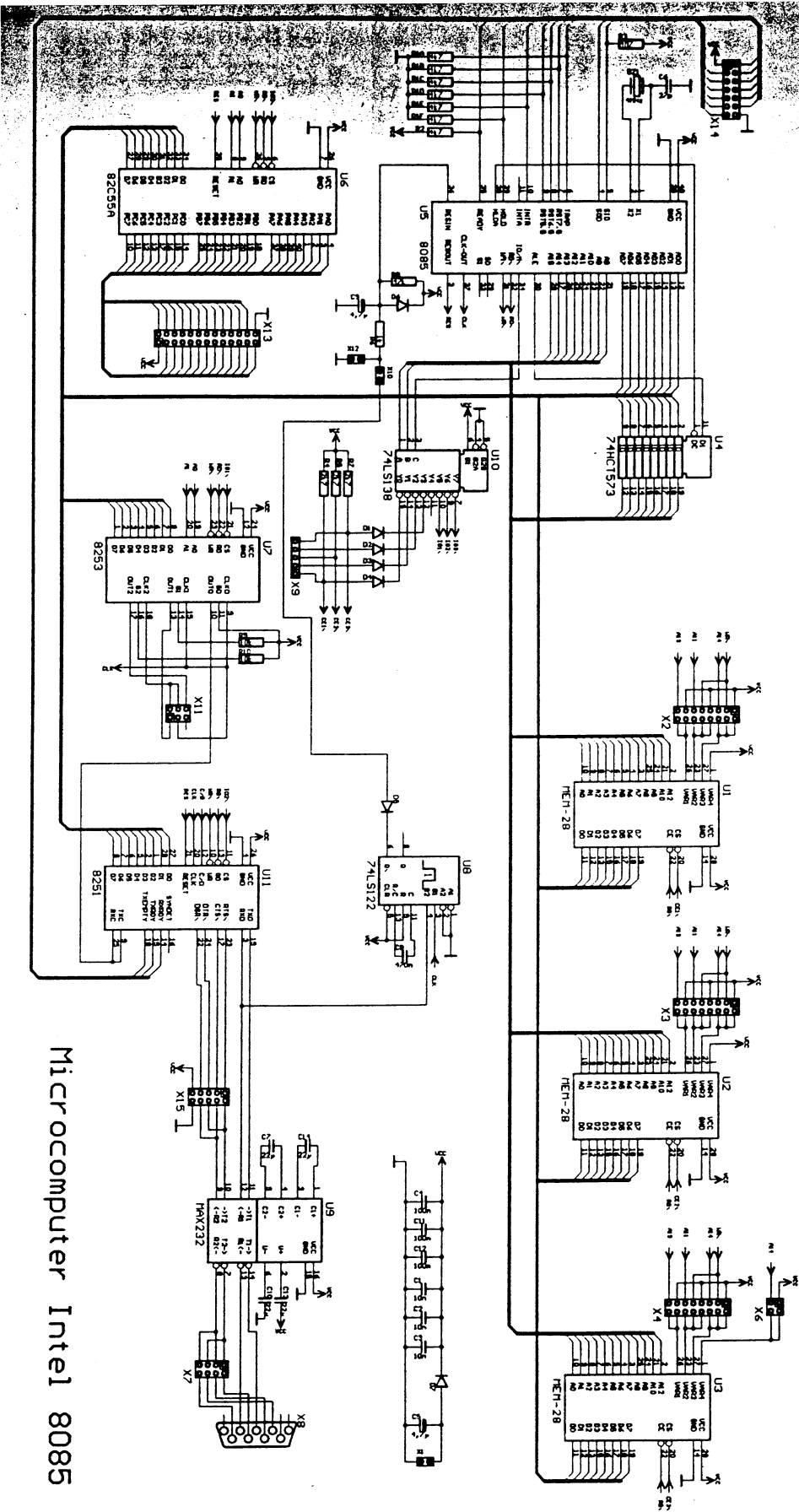
1 * 14-polig	2 * 16-polig	1 * 20-polig
1 * 24-polig	4 * 28-polig	2 * 40-polig
1,75	1,85	1,1
1,45	1,55	1,35

Verbindungselemente

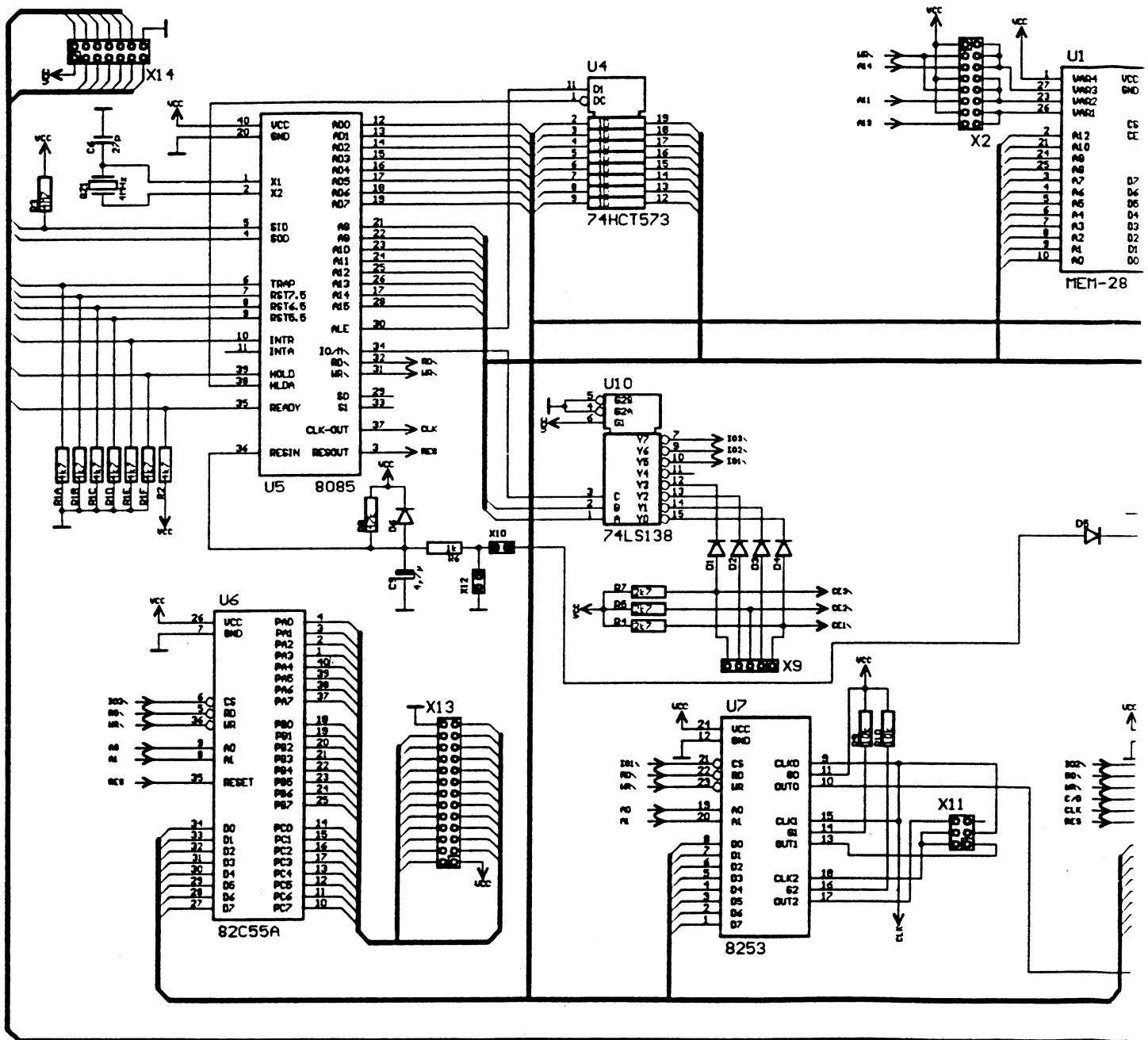
X1	Anschluß-Klemmleiste 2-polig	1,55
X2	Pfostenleiste 2 * 8	-
X3	Pfostenleiste 2 * 8	-
X4	Pfostenleiste 2 * 8	X
X6	Pfostenleiste 2 * 5	X
X7	Pfostenleiste 2 * 4	-
X8	SUB-D-Steckerleiste, einlötlbar, 90°-gewinkelt 9-polig	✓
X9	Pfostenleiste 1 * 5	1,35
X10	Pfostenleiste 2 * 1	-
X11	Pfostenleiste 2 * 3	X
X12	Pfostenleiste 2 * 1	-
X13	Pfostenleiste 2 * 13	X
X14	Pfostenleiste 2 * 7	X
X15	Pfostenleiste 2 * 2	-

3.1. SCHALTUNGSAUFBAU

3.1.2 Schaltplan

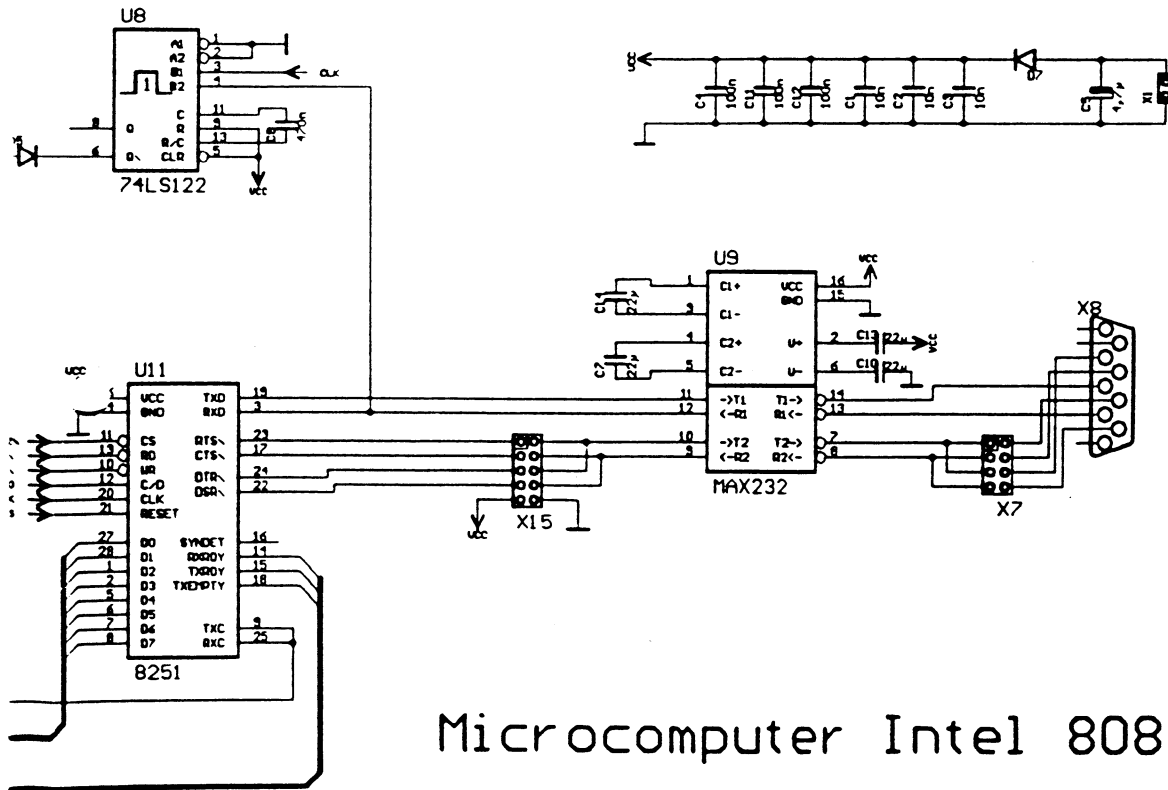
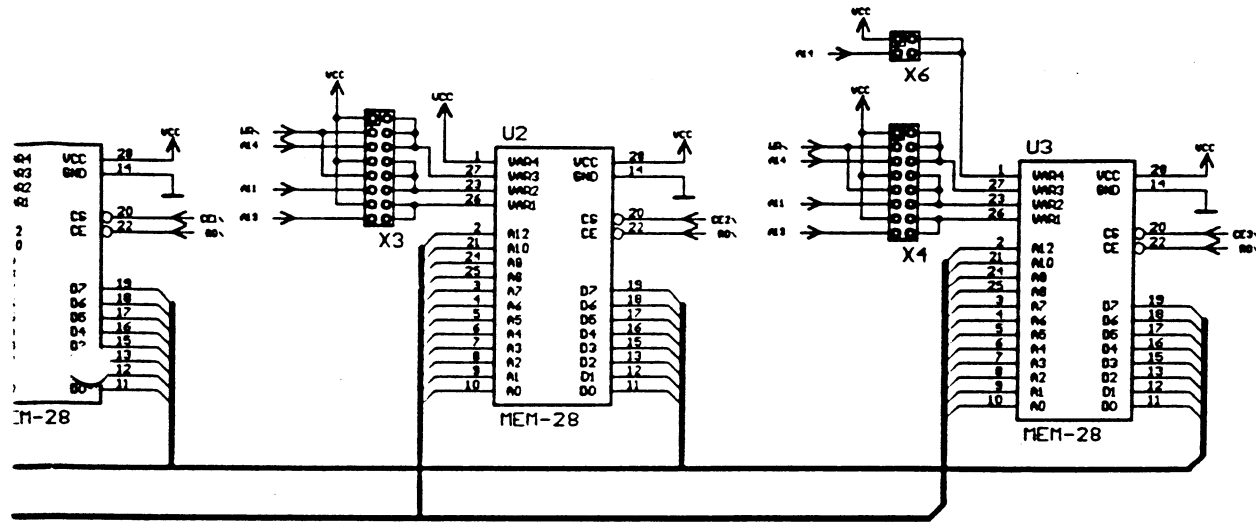


Micromcomputer Intel 8085



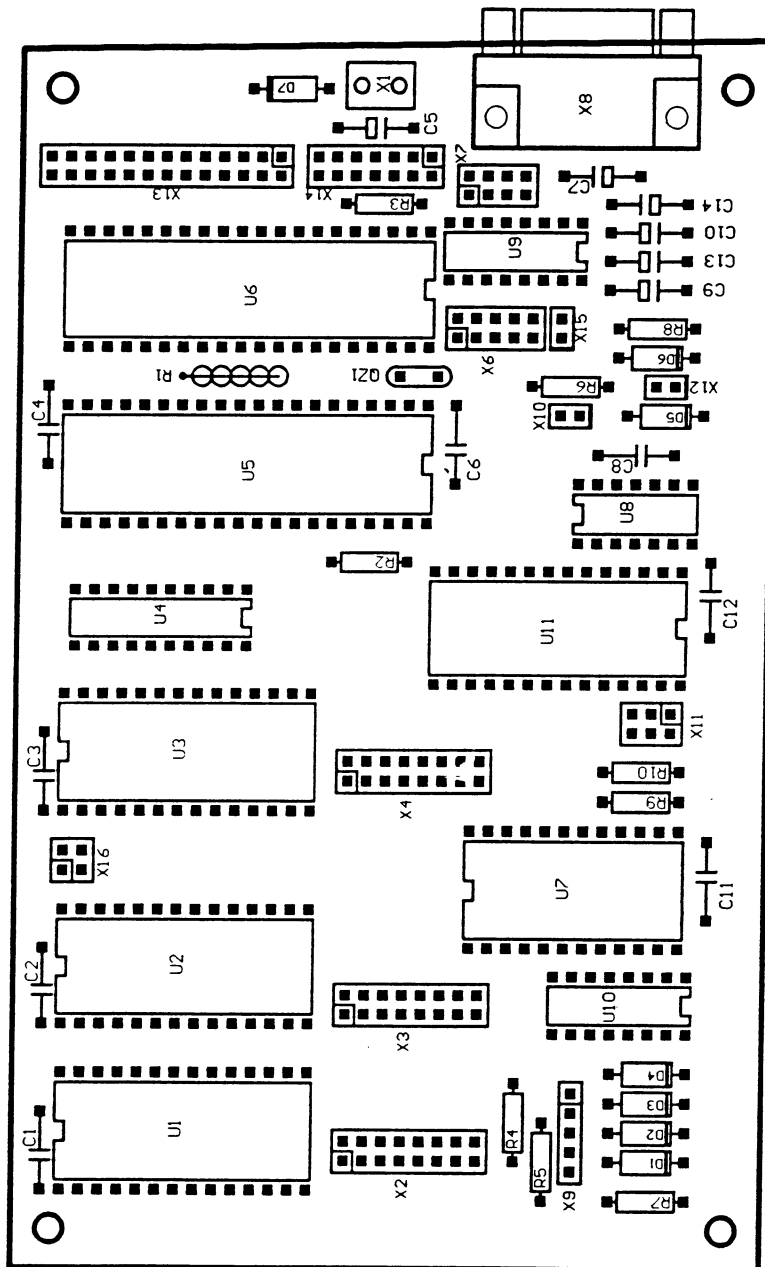
3.1. SCHALTUNGSANFBAU

3.1.2 Schaltplan



Microcomputer Intel 8085

3.1.3 Bestückungsplan



3.2 Übersicht Speicherbausteine

Pinbelegung von unterschiedlichen Speichern

Pin	A	B	C	D	E	F	G	H	J	I	K	L
1	Vpp	Vpp	NC	RFS	Vpp		RFS					
2	A12	A12	A12	A12	A12		NC					
3(1)	A7	A7	A7	A7	A7	A7	A7	A7	A7	A7	A7	A7
4(2)	A6	A6	A6	A6	A6	A6	A6	A6	A6	A6	A6	A6
5(3)	A5	A5	A5	A5	A5	A5	A5	A5	A5	A5	A5	A5
6(4)	A4	A4	A4	A4	A4	A4	A4	A4	A4	A4	A4	A4
7(5)	A3	A3	A3	A3	A3	A3	A3	A3	A3	A3	A3	A3
8(6)	A2	A2	A2	A2	A2	A2	A2	A2	A2	A2	A2	A2
9(7)	A1	A1	A1	A1	A1	A1	A1	A1	A1	A1	A1	A1
10(8)	A0	A0	A0	A0	A0	A0	A0	A0	A0	A0	A0	A0
11(9)	D0	D0	D0	D0	D0	D0	D0	D0	D0	D0	D0	D0
12(10)	D1	D1	D1	D1	D1	D1	D1	D1	D1	D1	D1	D1
13(11)	D2	D2	D2	D2	D2	D2	D2	D2	D2	D2	D2	D2
14(12)	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND
15(13)	D3	D3	D3	D3	D3	D3	D3	D3	D3	D3	D3	D3
16(14)	D4	D4	D4	D4	D4	D4	D4	D4	D4	D4	D4	D4
17(15)	D5	D5	D5	D5	D5	D5	D5	D5	D5	D5	D5	D5
18(16)	D6	D6	D6	D6	D6	D6	D6	D6	D6	D6	D6	D6
19(17)	D7	D7	D7	D7	D7	D7	D7	D7	D7	D7	D7	D7
20(18)	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE	CE
21(19)	A10	A10	A10	A10	A10	A10	A10	A10	A10	A10	A10	L
22(20)	OE	OE	OE	OE	OE	OE	OE	OE	OE	OE	OE	OE
23(21)	A11	A11	A11	A11	A11	A11	NC	Vpp	WE	WE	Upp	WE
24(22)	A9	A9	A9	A9	A9	A9	A9	A9	A9	A9	A9	A9
25(23)	A8	A8	A8	A8	A8	A8	A8	A8	A8	A8	A8	A8
26(24)	A13	A13	CE2	CS	NC	Vcc	CS	Vcc	Vcc	Vcc	Vcc	Vcc
27	A14	PGM	R/W	WE	PGM		WE					
28	Vcc	Vcc	Vcc	Vcc	Vcc		Vcc					

- A EPROM 27256
- B EPROM 27128
- C CRAM 5564
- D PRAM 4864
- E EPROM 2764
- F EPROM 2732
- G PRAM 4816
- H EEPROM 2816
- I SRAM 4802
- J CRAM 6116
- K EPROM 2716
- L SRAM 4118

3.3 8085-Befehlssatz

Datentransport-Befehle

Mnemonic	Code	TZ	By.	MZ	Funktion des Befehls	NZPCH
MOV r1,r2	..	4	1	1	(r1) \leftarrow (r2)	xxxxx
MOV M,r	..	7	1	2	(M) \leftarrow (r)	xxxxx
MOV r,M	..	7	1	2	(r) \leftarrow (M)	xxxxx
MVI r,n	..	7	2	2	(r) \leftarrow n	xxxxx
MVI M,n	36	10	2	3	(M) \leftarrow n	xxxxx
LXI B,m	01	10	3	3	(B/C) \leftarrow m	xxxxx
LXI D,m	11	10	3	3	(D/E) \leftarrow m	xxxxx
LXI H,m	21	10	3	3	(H/L) \leftarrow m	xxxxx
LXI SP,m	31	10	3	3	(SP) \leftarrow m	xxxxx
SPHL	F9	6	1	1	(SP) \leftarrow (H/L)	xxxxx
STAX B	02	7	1	2	(B/C) \leftarrow (A)	xxxxx
STAX D	12	7	1	2	(D/E) \leftarrow (A)	xxxxx
LDAX B	0A	7	1	2	(A) \leftarrow (B/C)	xxxxx
LDAX D	1A	7	1	2	(A) \leftarrow (D/E)	xxxxx
STA m	32	13	3	4	(m) \leftarrow (A)	xxxxx
LDA m	3A	13	3	4	(A) \leftarrow (m)	xxxxx
SHLD m	22	16	3	5	(m) \leftarrow (L) (m+1) \leftarrow (H)	xxxxx
LHLD m	2A	16	3	5	(L) \leftarrow (m) (H) \leftarrow (m+1)	xxxxx
XCHG	EB	4	1	1	(H/L) \leftarrow (D/E)	xxxxx
XTHL	E3	16	1	5	(H/L) \leftarrow (SP+1/SP)	xxxxx

Register Inkrement, Dekrement

Mnemonic	Code	TZ	By.	MZ	Funktion des Befehls	NZPCH
INR r	..	4	1	1	(r) \leftarrow (r)+1	??x?
INR M	34	10	1	3	(M) \leftarrow (M)+1	??x?
DCR r	..	4	1	1	(r) \leftarrow (r)-1	??x?
DCR M	35	10	1	3	(M) \leftarrow (M)-1	??x?
INX B	03	6	1	1	(B/C) \leftarrow (B/C)+1	xxxxx
INX D	13	6	1	1	(D/E) \leftarrow (D/E)+1	xxxxx
INX H	23	6	1	1	(H/L) \leftarrow (H/L)+1	xxxxx
INX SP	33	6	1	1	(SP) \leftarrow (SP)+1	xxxxx
DCX B	0B	6	1	1	(B/C) \leftarrow (B/C)-1	xxxxx
DCX D	1B	6	1	1	(D/E) \leftarrow (D/E)-1	xxxxx
DCX H	2B	6	1	1	(H/L) \leftarrow (H/L)-1	xxxxx
DCX SP	3B	6	1	1	(SP) \leftarrow (SP)-1	xxxxx

Arithmetik, Logik, Vergleich

Mnemonic	Code	TZ	By.	MZ	Funktion des Befehls	NZPCH
ADD r	..	4	1	1	$(A) \leftarrow (A) + (r)$?????
ADD M	86	7	1	2	$(A) \leftarrow (A) + (M)$?????
ADI n	C6	7	2	2	$(A) \leftarrow (A) + n$?????
ADC r	..	4	1	1	$(A) \leftarrow (A) + (r) + C$?????
ADC M	8E	7	1	2	$(A) \leftarrow (A) + (M) + C$?????
ADC n	CE	7	2	2	$(A) \leftarrow (A) + n + C$?????
DAD B	09	10	1	3	$(H/L) \leftarrow (HL) + (BC)$	xxx?x
DAD D	19	10	1	3	$(H/L) \leftarrow (HL) + (DE)$	xxx?x
DAD H	29	10	1	3	$(H/L) \leftarrow (H/L) * 2$	xxx?x
DAD SP	39	10	1	3	$(H/L) \leftarrow (HL) + (SP)$	xxx?x
SUB r	..	4	1	1	$(A) \leftarrow (A) - (r)$?????
SUB M	96	7	1	2	$(A) \leftarrow (A) - (M)$?????
SUI n	D6	7	2	2	$(A) \leftarrow (A) - n$?????
SBB r	..	4	1	1	$(A) \leftarrow (A) - (r) - C$?????
SBB M	9E	7	1	2	$(A) \leftarrow (A) - (M) - C$?????
SBI n	DE	7	2	2	$(A) \leftarrow (A) - n - C$?????
ANA r	..	4	1	1	$(A) \leftarrow (A) \text{AND}(r)$???01
ANA M	A6	7	1	2	$(A) \leftarrow (A) \text{AND}(M)$???01
ANI n	E6	7	2	2	$(A) \leftarrow (A) \text{AND } n$???01
XRA r	..	4	1	1	$(A) \leftarrow (A) \text{XOR}(r)$???00
XRA M	AE	7	1	2	$(A) \leftarrow (A) \text{XOR}(M)$???00
XRI n	EE	7	2	2	$(A) \leftarrow (A) \text{XOR } n$???00
ORA r	..	4	1	1	$(A) \leftarrow (A) \text{OR}(r)$???00
ORA M	B6	7	1	2	$(A) \leftarrow (A) \text{OR}(M)$???00
ORI n	F6	7	2	2	$(A) \leftarrow (A) \text{OR } n$???00
CMP r	..	4	1	1	$(A) - (r)$?????
CMP M	BE	7	1	2	$(A) - (M)$?????
CPI n	FE	7	2	2	$(A) - n$?????

Akku rotieren, beeinflussen, Carry setzen

Mnemonic	Code	TZ	By.	MZ	Funktion des Befehls	NZPCH
RLC	07	4	1	1	Links schieben	xxx?x
RRC	0F	4	1	1	Rechts schieben	xxx?x
RAL	17	4	1	1	Links schieben	xxx?x
RAR	1F	4	1	1	Rechts schieben	xxx?x
CMA	2F	4	1	1	$(A) \leftarrow \text{Not } (A)$	xxxxx
DAA	27	4	1	1	(A) in BCD wandeln	?????
STC	37	4	1	1	$(C) \leftarrow 1$	xxx1x
CMC	3F	4	1	1	$(C) \leftarrow \text{Not } (C)$	xxx?x

Sprungbefehle

Mnemonic	Code	TZ	By.	MZ	Funktion des Befehls	NZPCH
JMP m	C3	10	3	3	(PC) \leftarrow m	xxxxx
PCHL	E9	6	1	1	(PC) \leftarrow (H/L)	xxxxx
JC m	DA	10/7	3	3/2	(C)=1	xxxxx
JNC m	D2	10/7	3	3/2	(C)=0 wenn erfüllt	xxxxx
JZ m	CA	10/7	3	3/2	(Z)=1 (PC) \leftarrow m	xxxxx
JNZ m	C2	10/7	3	3/2	(Z)=0	xxxxx
JP m	F2	10/7	3	3/2	(N)=0 nicht erfüllt	xxxxx
JM m	FA	10/7	3	3/2	(N)=1 (PC) \leftarrow (PC)+3	xxxxx
JPE m	EA	10/7	3	3/2	(P)=1	xxxxx
JPO m	E2	10/7	3	3/2	(P)=0	xxxxx

Unterprogrammaufrufe

Mnemonic	Code	TZ	By.	MZ	Funktion des Befehls	NZPCH
CALL m	CD	18	3	5	(SP-1)(SP-2) \leftarrow (PC+3) (PC) \leftarrow m (SP) \leftarrow (SP)-2	xxxxx
RST n	..	12	1	3	(SP-1)(SP-2) \leftarrow (PC+1) (PC) \leftarrow n*8 (SP) \leftarrow (SP)-2	xxxxx
CC m	DC	18/9	3	5/2	(C)=1	xxxxx
CNC m	D4	18/9	3	5/2	(C)=0 wenn erfüllt	xxxxx
CZ m	CC	18/9	3	5/2	(Z)=1 s. CALL	xxxxx
CNZ m	C4	18/9	3	5/2	(Z)=0	xxxxx
CP m	F4	18/9	3	5/2	(N)=0 nicht erfüllt	xxxxx
CM m	FC	18/9	3	5/2	(N)=1 (PC) \leftarrow (PC)+3	xxxxx
CPE m	EC	18/9	3	5/2	(P)=1	xxxxx
CPO m	E4	18/9	3	5/2	(P)=0	xxxxx

Unterprogramm-Rücksprünge

Mnemonic	Code	TZ	By.	MZ	Funktion des Befehls	NZPCH
RET	C9	10	1	3	(PC) \leftarrow (SP+1) (SP) \leftarrow i-+2	xxxxx
RC m	D8	12/6	1	3/1	(C)=1	xxxxx
RNC m	D0	12/6	1	3/1	(C)=0 wenn erfüllt	xxxxx
RZ m	C8	12/6	1	3/1	(Z)=1 s. RET	xxxxx
RNZ m	C0	12/6	1	3/1	(Z)=0	xxxxx
RP m	F0	12/6	1	3/1	(N)=0 nicht erfüllt	xxxxx
RM m	F8	12/6	1	3/1	(N)=1 (PC) \leftarrow (PC)+1	xxxxx
RPE m	E8	12/6	1	3/1	(P)=1	xxxxx
RPO m	E0	12/6	1	3/1	(P)=0	xxxxx

Ein- Ausgabe, Unterbrech. Steuerung

Mnemonic	Code	TZ	By.	MZ	Funktion des Befehls	NZPCH
IN n	DB	10	2	3	(A) \leftarrow (Port n)	xxxxx
OUT n	D3	10	2	3	(Port n) \leftarrow (A)	xxxxx
EI	F8	4	1	1	(INTE) \leftarrow 1	xxxxx
DI	F3	4	1	1	(INTE) \leftarrow 0	xxxxx

Stapelsteuerung

Mnemonic	Code	TZ	By.	MZ	Funktion des Befehls	NZPCH
PUSH PSW	F5	12	1	3	Akku und Flags auf (SP)	xxxxx
PUSH B	C5	12	1	3	B und C-Reg. auf (SP)	xxxxx
PUSH D	D5	12	1	3	D und E-Reg. auf (SP)	xxxxx
PUSH H	E5	12	1	3	H und L-Reg. auf (SP)	xxxxx
POP PSW	F1	10	1	3	Akku und Flags vom (SP)	?????
POP B	C1	10	1	3	B und C-Reg. vom (SP)	xxxxx
POP D	D1	10	1	3	D und E-Reg. vom (SP)	xxxxx
POP H	E1	10	1	3	H und L-Reg. vom (SP)	xxxxx

Sonstige 8085-Befehle

Mnemonic	Code	TZ	By.	MZ	Funktion des Befehls	NZPCH
HLT	76	5	1	1	(PC) \leftarrow (PC)+1	xxxxx
NOP	00	4	1	1	(PC) \leftarrow (PC)+1	xxxxx
RIM	20	4	1	1	Liest vom ser. Eingang (A)	xxxxx
SIM	30	4	1	1	Schreibt auf ser. Ausgang	xxxxx

3.4 Unterprogramme

Ein-/Ausgabe – Unterprogramme

- RCHAR 0043 Ein Zeichen einlesen (SERIN)
 (0AE0) CALL RCHAR
 (A) = Zeichen

- WCHAR 0052 Ein Zeichen auf Bildschirm ausgeben (SEROUT)
 (0BB6) MVI A,Zeichen
 CALL WCHAR

- WCHARI 0055 Ein Zeichen nach "CALL" ausgeben
 (0BF7) CALL WCHARI
 DB Zeichen

- WAHEX 0058 8-Bit-Wert (A) als 2 Hex-Zeichen ausgeben
 (0B6F)

- WHLHEX 005B 16-Bit-Wert (HL) als 4 Hex ausgeben
 (0C1C)

- WABIN 005E 8-Bit-Wert (A) binär ausgeben
 (0B04)

- WADEZ 0061 8-Bit-Wert (A) dezimal ausgeben
 (0B19) (mit Unterdrückung der führenden Nullen)

- WAFOR 0064 8-Bit-Wert entsprechend Format ausgeben
 (0B4F) MVI A,Wert
 MVI C,Format (0=ASCII,1=BIN,2=DEZ,3=HEX)
 CALL WAFOR

- WBLANK 0067 Ein Leerzeichen ausgeben
 (0B8E)

- WBUF 006A Puffer ←(HL) ausgeben
 (0BA1) Textpuffer="TEXT",0 (0 als Endezeichen)
 (HL) zeigt auf Wert nach dem Endezeichen
 LXI H,Anfangsadresse des Textpuffer
 CALL WBUF

- WBUFI 006D Textpuffer nach "CALL" ausgeben
 (0BB0) CALL WBUFI
 DB "TEXT",0

- WCRLF 0070 Neue Zeile (CR+LF ausgeben)
 (0C01)

- WCRLFI 0073 Text nach "CALL" in neue Zeile ausgeben
 (0C13) (CR+LF und Text nach "CALL" ausgeben)
 CALL WCRLFI
 DB "TEXT",0

Eingabpufferung

- BCLR 0CAA Eingabepuffer löschen
CALL BCLR

- BGETF 0CCD Erstes Zeichen aus Eingabe-Puffer holen
CALL BGETF
JC PufferLeer
JNC Zeichen in (A)

- BPUT 0CE9 Zeichen in Eingabepuffer ablegen
MVI A, Zeichen
CALL BPUT
JC PufferVoll
JNC noch Plätze im Puffer frei

- BREAD 0D0A Zeichen einlesen, kontrollieren und puffern

 - RUBOUT zuletzt eingegebenes Zeichen löschen
 - CR od. SPACE Eingabe normal beenden
 - + od. - Eingabe beenden wenn BCKFLG=1
 - ESC Eingabe abbrechen
 - Kon. Zeichen werden ignoriert
 - MVI B, maximale Zahl der Zeichen
 - MVI C, Format (0=ASCII,1=BIN,2=DEZ,3=HEX)
 - CALL BREAD
 - JC ABRUCH
 - JNC CR,BLANK(od. - od. + ,falls BCKFLG=1)
 - (A) = Abschlusszeichen

- BGETL 0CDB Letztes Zeichen aus Eingabepuffer holen
(sonst siehe BGETF)

Character-Test und -manipulationen

- CHTST 0E84 Testen, ob ASCII-Zeichen angegebenem Format entspricht
 MVI A,ASCII-Zeichen
 MVI C,Typ (0=ASCII,1=BIN,2=DEZ,3=HEX)
 CALL CHTST Carry=1: Falsch ; C=0: O.K.

- CMP2 0EA8 Vergleiche (DE) mit (HL) ; 16 Bit-Vergleich
 LXI D,WERT1
 LXI H,WERT2
 CALL CMP2
 Carry=1: (HL) > (DE)

- GROSS 0EE9 Kleine Buchstaben in GROSSE (A) wandeln
 LDA ASCII-Zeichen
 CALL GROSS

- HEXASC 0EF2 HEX-Zahl in ASCII wandeln

- SUB2 1039 Zwei 16-Bit-Werte subtrahieren
 (HL) = (HL) - (DE)

Hole Tastatureingaben in speziellem Format

- HADR 08DF Hole Adresse max. 4 Zeichen Hexadezimal
LHLD Altwert
CALL HADR
(HL) = Altwert / Neuwert
(A) = Abschlusszeichen (CR,Space,ETX)
Carry= 1 Altwert ; 0 Neuwert

- HDATA 0941 Hole 8-Bit Datenbyte
MVI C,Format (0=ASCII,1=BIN,2=DEZ,3=HEX)
LXI H,Adresse Altwert / Neuwert
CALL HDATA
(D) = Neuer bzw. Alter Datenwert

- HDATAD 096E Hole Datenbyte mit Default
CALL HDATAD
(D) = Neuer bzw. Alter Datenwert

- HEINAUS 097B Hole "EIN / AUS"
LXI H,EIN/AUS-Adresse
CALL HEINAUS
(B) = Alter bzw. Neuer EIN(1)/AUS(0)-Zustand
gleicher Wert unter Adresse in (HL)

- HFORMD 09C5 Hole Zahlenformat
(mit Ausdruck "FORMAT=ALTWERT")
CALL HFORMD
FORMAT=Neues bzw. Altes Format
Variable FORMAT in Adresse FCA1h

- HJANEIN 0A0C Hole "JA / NEIN"
CALL HJANEIN
JZ JA

- HPORTD 0A2C Hole Ein bzw. Ausgabe-Port
Default wird zuvor ausgedruckt
Eingabe: Port-Adresse ; +/-
CALL HPORTD
Port= Alte bzw. Neue Portadresse

- HSTART 0A6D Hole Startadresse (4 Hex-Zeichen)
LHLD Alter Wert
CALL HSTART
(HL) = Neu bzw. Altwert

- HSTOPD 0A9C Hole Stopadresse, mit Default
LHLD Altwert
CALL HSTOPD

- WAASC 0AEE 8-Bit-Wert in ASCII ausgeben
MVI A,8-Bit Wert
CALL WAASC
z.B. .X , .Z , 10 , B4

3.5 Steuerbefehle des Editors

— Verarbeitung —

Ctrl + N	Neustart	Ctrl + S	Speichern/Laden
Ctrl + D	Drucker-Menü	Ctrl + A	Assemblieren

— Zeichen editieren —

Ctrl + R	Zeichen radieren
Ctrl + DEL	Zeichen links löschen
Ctrl + E	Einfügen Ein/Aus

— Zeile editieren —

Ctrl + Z + L	Zeile löschen
Ctrl + Z + E	Zeile einfügen
Ctrl + Z + A	Zeile anhängen

— Cursor-Steuerung —

Ctrl + C + O	An oberen Bildrand
Ctrl + C + U	An unteren Bildrand
Ctrl + C + L	An linken Bildrand
Ctrl + C + R	An rechten Zeilenrand
Ctrl + W	Auf nächstes Wort

— Seiten blättern —

Ctrl + U	Seite unten
Ctrl + O	Seite oben
Ctrl + C + A	Cursor zum Textanfang
Ctrl + C + E	Cursor zum Textende
Ctrl + T	Tabulator

— Block-Befehle —

Ctrl + B + M	Block markieren
Ctrl + B + R	B.-Mark. rücksetzen
Ctrl + B + F	B.-Mark. finden
Ctrl + B + L	Block löschen
Ctrl + B + K	Block kopieren
Ctrl + B + V	Block verschieben
Ctrl + B + S	Block speichern

— Finde / Ersetze —

Ctrl + F + S	Setze Suchwort
Ctrl + F + F	Finde Suchwort
Ctrl + F + E	Ersetze Suchwort

Belegung der speziellen MF2-Sondertasten

Ins	≐	Ctrl + E	Shift + Tab	≐	Ctrl + W
Del	≐	Ctrl + R	Ctrl + Home	≐	Ctrl + C + O
Pos1	≐	Ctrl + C + L	Ctrl + End	≐	Ctrl + C + U
End	≐	Ctrl + C + R	Ctrl + PgUp	≐	Ctrl + C + A
PgUp	≐	Ctrl + O	Ctrl + PgDn	≐	Ctrl + C + E
PgDn	≐	Ctrl + U			