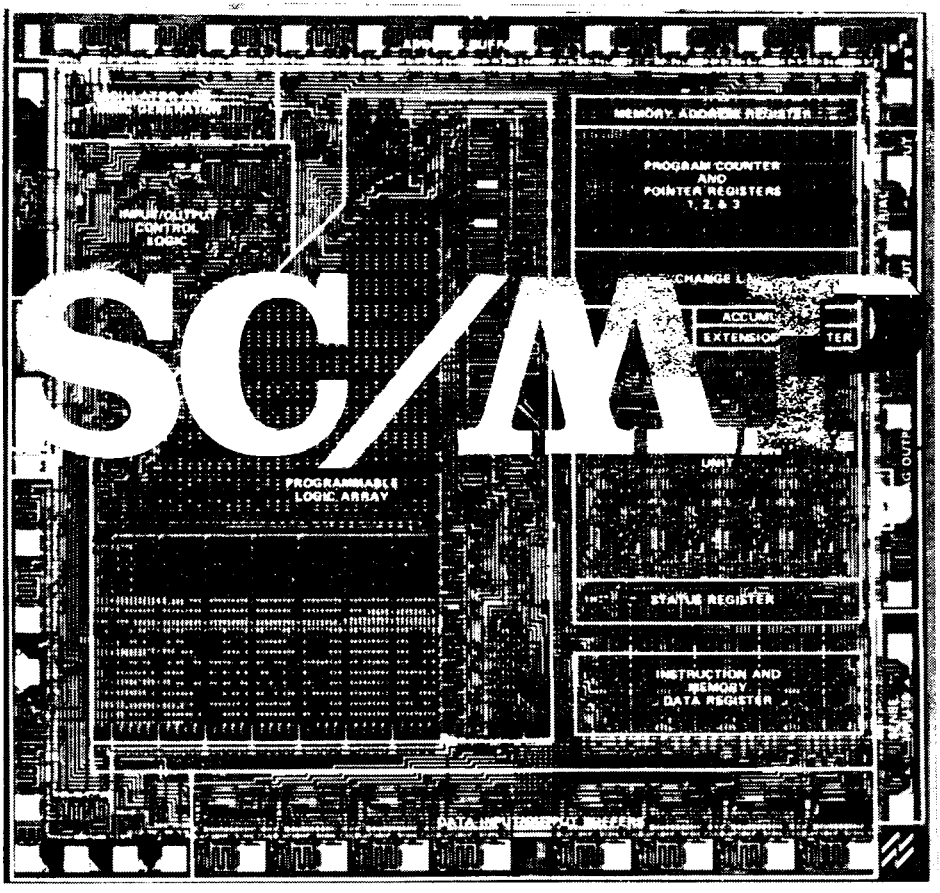
C. LORENZ



MICROCOMPUTER Handbuch

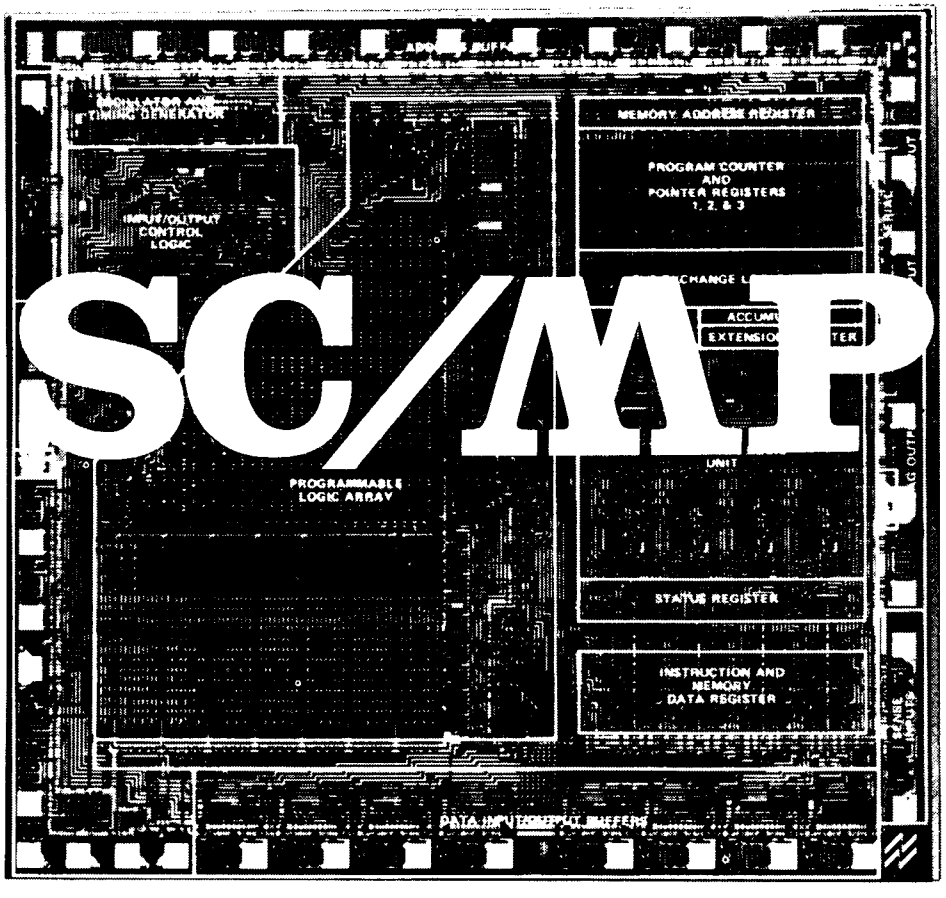
Autoren: H. Nössler
M. Reimer
W. Hofacker
C. Lorenz
H. Jansen
Satz u. Repro: A. Grünwald

Wir danken für die großzügige Unterstützung der Firma Homecomputer Vertriebs GmbH, Flügelstr. 47 in 4000 Düsseldorf 1 und Herrn Manfred Reimer für den Beitrag über TINY BASIC und WILLY.

ISBN 3-921682-42-8

1. Auflage 1980
Gedruckt in der Bundesrepublik Deutschland – Printed in West-Germany – Imprime'en RFA.

C. LORENZ



MICROCOMPUTER Handbuch

Inhaltsverzeichnis

Einführung

Will man in die Microcomputertechnik einsteigen hat jeder Interessent ein bestimmtes Ziel im Auge. Interesse an Hardware ausschließlich dürfte selten vorhanden sein, da der Computer unbedingt Software benötigt um überhaupt zu arbeiten. Software-Interesse alleine ist schon eher möglich, da es ja bereits eine Reihe von kompletten Computersystemen auf dem Markt gibt.

Hardware und Software zusammen braucht wohl der größte Anteil von allen Microcomputer-Benutzern. Welchen Weg der Microcomputeranwender beim Einstieg beschreiten will, ist weitgehend aber auch von seinen Vorkenntnissen und seinen Geldbeutel abhängig.

Erfahrungen haben gezeigt, daß es fast unmöglich ist, sich Microcomputerkenntnisse nur durch Theorie anzueignen. Der Interessent muß auf jeden Fall an einem System Erfahrungen sammeln. Zur Erstellung eines eigenen Systems gibt es folgende Möglichkeiten:

1. Aufbau eines Systems aus Einzelteilen. Diese Möglichkeit ist relativ kostengünstig, benötigt aber gute Kenntnisse. Trotzdem kann bei Problemen die "Geschichte" sehr teuer werden.
2. Kaufen eines Bausatzes oder nach einer Baubeschreibung eines laufenden Systems (Bauanleitungen aus Magazinen, Büchern etc).
3. Anschaffung eines kompletten Systems (eine relativ teure Lösung).

Heute sind auf dem Weltmarkt zwei Arten von Systemen relativ weit verbreitet.

1. Art: Sehr preiswert und einfach, aber nich erweiterbar, oder nur unter größeren Aufwand. Z.B. SC/MP-Introkit und Keyboardkit, KIM-1 (6502) oder MEK 6800 D-1 (6800).
2. Art: Komplette Systeme mit Erweiterungsmöglichkeit bis 64K, Floppy Anschluß, Drucker etc. (diese Art ist relativ teuer, ab ca. DM 6.000,–).

Die Aufgabe des vorliegenden Buches ist:

1. Dem Anwender soll die Möglichkeit gegeben werden ein arbeitendes System nachzubauen. Bei der Erstellung seines Microcomputersystems kann er fertige Leiterplattenvorlagen aus diesen Buche verwenden.
2. Den tausenden von SC/MP-Benutzern ein praktisches Handbuch für Systemerweiterungen oder den Aufbau eines kompletten eigenen Systems zu geben.

Dieses Buch wendet sich an den durchschnittlichen Microcomputer-Anwender, und soll keine theoretische Abhandlung über komplizierte Technologien sein. Es soll ein "Kochbuch" für den SC/MP Anwender sein. Für diejenigen, die mehr über technische Daten und Details wissen wollen, haben wir auch einen Datenblattanhang und Referenzliste angefügt.

Was finden Sie in diesem Buch?

Im ersten Teil besprechen wir den SC/MP Introkit/Keyboardkit mit seinen praktischen Erweiterungsmöglichkeiten. Dann folgt der Aufbau einer neuen CPU-Karte mit residenten NIBL-BASIC (TINY-BASIC). Für die Erstellung größerer Programme geben wir Ihnen dann die Möglichkeit, sich selbst eine 4K oder 8K statische, oder 16K dynamische Speicherverwertung zu bauen.
Dann folgt der Aufbau eines Cassetteninterfaces, sowie eines billigen Datensichtgerätes. Einen breiten Raum nehmen auch die Interfaceschaltungen zur Verbindung mit der Außenwelt ein (A/D D/A Wandler). Computer Musik ist auch ein interessanter Bestandteil.

Allgemeine Einführung

Nachdem der Microprocessor vor einigen Jahren auf den Markt kam und seinen Siegeszug angetreten hat, ist über diesen hochintegrierten Schaltkreis schon vieles geschrieben worden. Diese Systembeschreibung des SC/MP soll keine allgemeine Ausführung über Microprocessoren sein, sondern sich speziell mit diesem Home Computer System befassen. Für diejenigen, die ihre Kenntnisse über den Microprocessor vervollständigen oder aber sich in die Digitaltechnik einerseits und die Datenverarbeitung andererseits einarbeiten wollen, empfehlen wir folgende Bücher:

1. Hobby Computer Handbuch — Best.-Nr. 25, Hofacker Verlag
2. Software Handbuch — Best.-Nr. 27, Hofacker Verlag
3. SC/MP Microcomputer Bücher Teil 1 + 2, — Elektor Verlag
4. SC/MP Programmier + Assembler Handbuch — Best.-Nr. 708, Hofacker Verlag

Selbstverständlich können wir auch hier nicht auf gewisse grundlegende Erläuterungen verzichten.

Einige wichtige Begriffe der Microprocessor-Technik (der Digitaltechnik) sind Bit, Byte, die Logikzustände high, low, tri state. Beim Microprocessor speziell treten dann noch die Begriffe dual, hexadezimal, oktal auf. Diese sollen hier kurz näher erläutert werden.

Das Bit

Das Bit ist die kleinste Einheit bei Digitalschaltungen. Das Bit kann sowohl Null oder Eins sein (nein oder ja). Es ist also die kleinste logische Einheit. Ein normales digitales Schaltnetzwerk kann nur zwischen diesen beiden Zuständen unterscheiden (0 oder 1). Durch Verarbeitung dieser Bits, wenn es sich um mehrere handelt, werden Digitalschaltungen gesteuert.

Das Byte

Die nächst größere Einheit ist das Byte. Es ist zusammengesetzt aus 8 Bit. Man nennt dieses Byte auch ein 8-Bit-Wort.

Definition high, low und hochohmig

Normalerweise versteht man in der Digitaltechnik unter high (logisch "1"), sofern es sich um TTL-Pegel handelt (5 Volt) eben diese 5 Volt bzw. einen Bereich, der in den Datenblättern angegeben wird.

Der Low-Zustand (logisch "0") wird als 0 Volt definiert bzw. der Bereich als logisch 0 angesehen wird.

Erst beim Microprocessor tritt der quasi dritte logische Zustand "hochohmig" auf. In diesem Zustand ist die Leitung praktisch nicht mehr existent, da sie einen relativ hohen Widerstand darstellt, der im M Ohm-Bereich liegt.

Wie oben schon erwähnt, arbeitet die Digitaltechnik mit dem Bit als kleinste Einheit. Da hier eine Stelle (Dualstelle) nur 0 oder 1 sein kann, sieht man ein, daß man hier nicht mit dem Dezimalsystem operieren kann, denn dieses geht von 0 bis 9. Darum wird hier mit dem Dualsystem operiert.

Die Digitaltechnik (überhaupt die Datenverarbeitung) arbeitet immer mit dem Dualsystem. Um aber dem Menschen, dem es sehr schwer fällt, in diesem System zu denken, das Arbeiten zu erleichtern, hat man noch andere Zahlensysteme eingeführt.

Im Binär- oder Dualsystem verwendet man zur Darstellung einer Zahl 2 Komponeten, die 0 und die 1. Der Stellenwert basiert auf den Potenzen der 2. Die erste Vereinfachung des Dualsystems für den Menschen ist das Oktalsystem. Hier werden 3 Bit zu einer Oktalzahl (0 bis 7) zusammengefaßt. Die nächste Vereinfachung, mit der wir später auch den Microprocessor programmieren, ist das Hexadezimalsystem (0 bis 15). Hier werden 4 Bit zu einer hexadezimalen Zahl zusammengefaßt.

Was ist ein Microprocessor?

Der Microprocessor ist eine hochintegrierte Schaltung. Er bildet das Herz in einem Microprocessorsystem. Er ist praktisch das Gehirn. Hier müssen sämtliche Operationen durchgeführt werden. Alle externen Bausteine sind an ihn angeschlossen. Im Prinzip besteht der Microprocessor (CPU = Central Processing Unit) aus folgenden Einheiten: Adreßregister, Programmzähler, Datenregister (evtl. mehrere), Befehlsdecoder, Controlleinheit, Accumulator, arithmetrische und logische Einheit.

Wie zu ersehen ist, hat die CPU im Normalfall nur Steuerungsaufgaben. Um diese Steuerungsaufgaben ausführen zu können, braucht die CPU ein Gedächtnis (Speicher). Innerhalb der CPU gibt es zwar einige Speicher (Register), von denen es im Normalfall höchstens 20 gibt, werden deshalb auch nicht als Speicher, sondern als sogennante Register be-

nutzt. Dieses sind Hilfsstellen, in die die CPU Zwischenergebnisse oder Adressen abspeichern kann. Speicher im eigentlichen Sinne werden über den sogenannten BUS von außen an die CPU angeschlossen. Der BUS ist eine Anzahl von Leitungen, die zwischen CPU und den externen Bauelementen verläuft. Über ihn korrespondiert die CPU mit den Peripheriegeräten.

Speicher

Um Informationen kurz- oder langfristig zu konservieren, benötigen wir Speicherbausteine. Die heutige Technologie unterscheidet zwischen statischen und dynamischen Speichern auf der einen Seite sowie zwischen fest programmierten Speichern (ROMS) und Schreib-/Lesespeichern (RAMS). Ein statischer Speicher besteht aus einem Schalter, der eine binäre Information darstellen kann (Flip Flop).

Der dynamische Speicher besteht im wesentlichen aus einem Kondensator, der für eine bestimmte Zeit die Information festhalten kann, sie dann aber wieder verlieren würde, wenn man sie nicht mit einer speziellen Schaltung immer wieder erneuern (auffrischen) würde.
Wegen dieses Auffrischens der Information kann auch nur während bestimmter Zeiten in diesen Speicher geschrieben oder aus ihm gelesen werden, während man beim statischen Speicher zu gleicher Zeit lesen und schreiben kann. Diese beiden Variationen beziehen sich auf den Schreib-/Lesespeicher.

Solche Schreib-/Leseschreiber verlieren bei Stromabschaltung ihre Information und dienen deshalb im Normalfall als Arbeitsspeicher im Microprocessorsystem.

Völlig anders der Festspeicher. Hier unterscheidet man im wesentlichen zwischen dem maskenprogrammierten ROM, dem PROM und dem EPROM.

Das maskenprogrammierte ROM kann nur vom Hersteller selbst programmiert werden. Da die Herstellung einer solchen Maske sehr teuer ist, kommt dieses kaum für den normalen Anwender in Frage.

Das PROM ist ein elektrisch programmierbares ROM. Es kann vom Anwender selbst elektrisch programmiert werden, aber nur einmal.

Diesen Nachteil, das nur einmal programmieren können, schaltet das EPROM aus. Dieses kann man elektrisch programmieren und durch UV-Strahlen wieder löschen. ROMS behalten ihre Information auch nach Abschalten der Versorgungsspannung.

Dieses Buch ist gegliedert in zwei große Abschnitte:
Hardware
Software

Hardware ist der materielle Teil eines Computers, also: Platinen, aktive und passive Bauelemente etc. während unter Software die Information mit der die Maschine arbeitet verstanden wird.

Hardware

Die beiden wichtigsten Hardwarelemente sind die CPU = central processing unit = zentrale Recheneinheit und die Speicher. Zwischen diesen Baugrauppen werden Informationen ausgetauscht.

Schnittstellen

Zur Korresspondenz mit der Zentraleinheit benötigt man Peripherie. Diese teilt sich auf in Ein- und Ausgabegerät. Über diese werden dem Prozessor Daten eingegeben oder man erhält vom Prozessor Daten. Peripheriegeräte können z.B. sein:

Für die Eingabe: Tastaturen, AD-Wandler oder Lochstreifenleser.
Für die Ausgabe: Lochstreifenstanzer, Drucker, Displays, TV oder DA-Wandler.

Als Ein- und Ausgabegräte werden betrachtet:
Cassetten- und Datenrecorder, Terminals, Floppy-Disks und Fernschreiber.

Diese Geräte können direkt am Bus liegen oder auch über Schnittstellen angeschlossen sein.

Schnittstellen sind allgemein genormte Anpassungsglieder. Die Daten können parallel d.h. alle 8 Bit gleichzeitig über 8 Leitungen oder seriell, d.h. hintereinander über eine Leitung übertragen werden. Für beide Arten von Schnittstellen gibt es integrierte Schaltkreise, die den Datenfluß verwalten und die Verbindung zum System herstellen.

Parallelschnittstellen können in drei Betriebsarten benutzt werden.
1. Ein- oder Ausgabe auf Befehl
2. Handshake / Ein- oder Ausgabe
3. Bidirektional / in beiden Richtungen

Serielle Schnittstellen können in zwei Betriebsarten benutzt werden
1. synchron
2. asynchron

Beide Betriebsarten können sowohl in:
symplex / Richtung schreiben
duplex / Richtung schreiben und lesen
halbduplex / wechselseitig schreiben und lesen
betrieben werden.

Die Bausteine für Parallelschnittstellen werden PIO, PIA und PORT genannt. Für serielle Schnittstellen: UART und USART.

Serielle Schnittstellen sind genormt. Es gibt folgende Normungen: V 24, RS 232, TTY und RTTY.

Im SC/MP ist übrigens eine serielle Schnittstelle bereits integriert. (SI + SO)

Peripherie

Die zur Eingabe verwendeten Tastaturen unterscheidet man in numerische und alphanumerische. Eine hexadezimale Tastatur ist eine numerische Tastatur. Die Tastatur eines Fernschreibers z.B. ist eine alphanumerische. Für die Umrechnung der Buchstaben in Ziffern gibt es verschiedene Codes. Der gebräuchlichste ist der ASCII-Code.

ASCII Code

Even	00	NUL	@	NULL, CTRL SHIFT P, TAPE LEADER
Odd	01	SOH	A	START OF HEADER, SOM
Odd	02	STX	B	START OF TEXT, EOA
Even	03	ETX	C	END OF TEXT, EOM
Odd	04	EOT	D	END OF TRANSMISSION, END
Even	05	ENQ	E	ENQUIRY, WRU, WHO ARE YOU
Even	06	ACK	F	ACKNOWLEDGE, RU, ARE YOU
Odd	07	BEL	G	BELL
Odd	08	BS	H	BACKSPACE, FE0
Even	09	HT	I	HORIZONTAL TAB, TAB
Even	0A	LF	J	LINE FEED, NEW LINE, NL
Odd	0B	VT	K	VERTICAL TAB, VTAB
Even	0C	FF	L	FORM FEED, FORM, PAGE
Odd	0D	CR	M	CARRIAGE RETURN, EOL
Odd	0E	SO	N	SHIFT OUT, RED SHIFT
Even	0F	SI	O	SHIFT IN, BLACK SHIFT
Odd	10	DLE	P	DATA LINK ESCAPE, DC0
Even	11	DC1	Q	XON, READER ON
Even	12	DC2	R	TAPE, PUNCH ON
Odd	13	DC3	S	XOFF, READER OFF
Even	14	DC4	T	TAPE, PUNCH OFF
Odd	15	NAK	U	NEGATIVE ACKNOWLEDGE, ERR
Odd	16	SYN	V	SYNCHRONOUS IDLE, SYNC
Even	17	ETB	W	END OF TEXT BUFFER, LEM
Even	18	CAN	X	CANCEL, CANCL
Odd	19	EM	Y	END OF MEDIUM
Odd	1A	SUB	Z	SUBSTITUTE
Even	1B	ESC	[	ESCAPE, PREFIX
Odd	1C	FS	\	FILE SEPARATOR
Even	1D	GS	]	GROUP SEPARATOR
Even	1E	RS	^	RECORD SEPARATOR
Odd	1F	US	–	UNIT SEPARATOR
Odd	20	SP	SPACE BLANK	
Even	21	!		
Even	22	"		
Odd	23	#		
Even	24	$		
Odd	25	%		
Odd	26	&		
Even	27	'	APOSTROPHI	
Even	28	(		
Odd	29	)		
Odd	2A	*		
Even	2B	+		
Odd	2C	,	COMMA	
Even	2D	-	MINUS	
Even	2E	.		
Odd	2F	/		
Even	30	0	NUMBER ZERO	
Odd	31	1	NUMBER ONE	
Odd	32	2		
Even	33	3		
Odd	34	4		
Even	35	5		
Even	36	6		
Odd	37	7		
Odd	38	8		
Even	39	9		
Even	3A	:		
Odd	3B	;		
Even	3C	<	LESS THAN	
Odd	3D	=		
Odd	3E	>	GREATER THAN	
Even	3F	?		
Odd	40	@	SHIFT P	

Even	41	A	
Even	42	B	
Odd	43	C	
Even	44	D	
Odd	45	E	
Odd	46	F	
Even	47	G	
Even	48	H	
Odd	49	I	LETTER I
Odd	4A	J	
Even	4B	K	
Odd	4C	L	
Even	4D	M	
Even	4E	N	
Odd	4F	O	LETTER O
Even	50	P	
Odd	51	Q	
Odd	52	R	
Even	53	S	
Odd	54	T	
Even	55	U	
Even	56	V	
Odd	57	W	
Odd	58	X	
Even	59	Y	
Even	5A	[	SHIFT K
Even	5C	\	SHIFT L
Odd	5D	]	SHIFT M
Odd	5E	^	, SHIFT N
Even	5F	–	, SHIFT O, UNDERSCORE
Even	60	`	ACCENT GRAVE
Odd	61	a	
Odd	61	B	
Even	63	c	
Odd	64	d	
Even	65	e	
Even	66	f	
Odd	67	g	
Odd	68	h	
Even	69	i	
Even	6A	j	
Odd	6B	k	
Even	6C	l	
Odd	6D	m	
Odd	6E	n	
Even	6F	o	
Odd	70	p	
Even	71	q	
Even	72	r	
Odd	73	s	
Even	74	t	
Odd	75	u	
Odd	76	v	
Even	77	w	
Even	78	x	
Odd	79	y	
Odd	7A	z	
Even	7B	{	
Odd	7C	|	VERTICAL SLASH
Even	7D	}	ALT MODE
Even	7E	~	(ALT MODE)
Odd	7F	DEL	DELETE, RUBOUT

even	= gerade
odd	= ungerade
SOH	Start des Headers
STX	Text Anfang
ETX	Text Ende
EOT	Ende der Übertragung
ENQ	Erkundigung (Wer bist Du?)
ACK	Bestätigung (Ich bin)
BEL	Klingel
BS	Rückschritt
HT	Strich (horizontal)
LF	Zeilenvorschub
VT	Strich (vertikal)
FF	Format wählen
CR	rück z. Zeilenanfang
SYN	gleichzeitig
ETB	Ende Textspeicher
CAN	Aufheben
EM	Ende der Vermittlung
SUB	Ersetzen
ESC	Vorsetzen
FS	Cursor rechts
GS	Cursor abwärts
RS	Cursor links
US	Cursor aufwärts
DEL	Streichen

Mittels eines Lochstreifenlesers können ebenfalls Daten eingegeben werden.

Ein AD-Wandler setzt Spannungen in digitale Signale um, die dann zur Auswertung in das MP-System eingegeben werden können.

Die Verarbeitung der ausgegebenen MP-Daten können von einem Lochstreifenstanzer auf einen Lochstreifen festgehalten werden. Man kann aber auch Daten mittels eines Druckers oder Fernschreibers zu Papier bringen. Eine weitere Version ist die Sichtbarmachung auf einem Display sei es hexadezimal oder alphanumerisch. Endlich eignet sich ein TV-Bildschirm zur Sichtbarmachung der Daten.

Cassetten-Recorder, Datenrecorder oder Floppy-Plattenspeicher dienen als Massenspeicher. Daten können hier abgelegt und wieder in das System eingelesen werden. Im Gegensatz zum Cassetten-Recorder können die beiden letzteren vom System gesteuert werden, so daß auf bestimmte Datenblöcke direkt zugegriffen werden kann.

Das HOME-Computer SC / MP-System

Unter der Vielzahl der Microprozessoren gibt es den SC/MP (Simple Coast Effective Micro-Prozessor) - man spricht Skemp - von National Semiconductor. Dieser Prozessor zeichnet sich durch eine problemlose Hardware und eine ebensolche Software aus. Er wird nicht zu Unrecht als der VW unter den Microprozessoren bezeichnet. Die erste SC/MP-Version der ISP 500 wurde durch die doppelt so schnelle 600er Version ersetzt. Dieser SC/MP 600, jetzt auch 8060 genannt, benötigt nur noch eine Versorgungsspannung, nämlich + 5V.

Die Firma Homecomputer GmbH hat, basierend auf diesem Prozessor, ein komplettes Personalcomputersystem geschaffen, das vom Anwender in Baugruppen selbst aufgebaut werden kann. Der Selbstbau und die Programmierung eines Computers ist, neben der vielfältigen Nutzungsmöglichkeit einer derartigen Maschine, auch ein faszinierendes Hobby.

Es sollte nicht verheimlicht werden, daß zum Aufbau eines solchen Systems Vorkenntnisse sowohl in der Praxis beim Aufbau elektronischer Schaltungen (Bestücken und Löten) als auch zumindest geringe Kenntnisse über allgemeine Elektronik vorhanden sein müssen. Wer derartige Kenntnisse nicht hat, braucht auf das Computer-Hobby nicht zu verzichten, er sollte jedoch seinen Personal-Computer oder dessen Einzelbausätze fertig bestückt und getestet kaufen.

Für den Bastler mit geringer Erfahrung sollen die folgenden allgemeinen Hinweise dienen:

Allgemeine Aufbauhinweise

Computerplatinen sind in den meisten Fällen doppelseitig beschichtete und durchkontaktierte Prints. Dies hat den Vorteil, daß auf einer Platine geringerer Größe eine große Anzahl von Bauteilen untergebracht werden kann und keine Brücken bei Leiterbahnkreuzungen eingesetzt werden müssen. Die auf galvanischem Wege erfolgende Durchkontaktierung erzeugt an den Rändern des Bohrloches eine Metallschicht, die wie eine kleine Buchse im Bohrloch sitzt. Diese feinen Metallfilme

sind sehr empfindlich bei mechanischer Beschädigung, wie sie z.B. beim Einlöten eines Bauteils drohen. Es ist daher erforderlich, bei der Bestückung solcher Platinen darauf zu achten, daß keine Bestückungsfehler auftreten, die ein Wiederauslöten erforderlich machen. Sollte trotzdem ein Teil wieder ausgelötet werden müssen, so muß zunächst mittels eines Lötzinnabsaugers oder einer Sauglitze der Lötzinn entfernt werden. Es ist ratsam, die noch verbleibenden Lötzinnreste - solange sie noch durch das Erhitzen mit dem Lötkolben flüssig sind - mit Kältespray abzuschrecken und somit spröde zu machen. Das Auslöten von integrierten Schaltkreisen verursacht die größten Schwierigkeiten. Aus diesem Grunde ist es in jedem Falle ratsam, die ICs zu sockeln, d.h. statt der Schaltkreise Fassungen einzulöten.

Löten

Verwenden Sie für die feinen Leiterbahnen und Lötpunkte einen Lötkolben mit maximal 30 W. Als Lötzinn sollten Sie einen dünnen Zinn zwischen 0,5 und 0,8 mm benutzen. Der Zinn darf nicht zuviel Lötharz enthalten.
Der Lötkolben sollte eine dünne Spitze haben.

Bestückungsreihenfolge

Bevor Sie mit der Bestückung der Platine beginnen, prüfen Sie anhand der Stückliste die Vollständigkeit des Bausatzes. Legen Sie die Einzelteile in der Reihenfolge der Stückliste auf Ihren Arbeitstisch und beginnen Sie mit der Bestückung bei den niedrigsten Bauelementen. In der Regel sind dies evtl. vorhandene Brücken. Es folgen liegend montierte Widerstände, Kondensatoren usw.

Beim Einlöten der IC Sockel achten Sie bitte darauf, daß alle Pins durchgesteckt sind. Viele IC Sockel enthalten bereits Markierungen für Pin 1 des ICs. Es ist zwar nicht schädlich, diese Sockel verkehrt einzulöten, die richtige Bestückung erspart aber Überlegungen beim Einstecken der ICs.

Achtung: MOS ICs

Bei den Microprozessorbausteinen handelt es sich vorwiegend um MOS-Bausteine, die trotz integrierter Schutzstruktur mit Vorsicht behandelt werden sollten.

Die Bauteile sind meistens in leitenden Schaumstoff oder antistatischen Röhren eingepackt oder aber auf Alufolie aufgesteckt. Es ist besser, die Teile beim Bestücken an den Köpfen und nicht an den Pins anzufassen. Vielfach sind die Beine nicht so abgewinkelt, daß sie in die Sockel passen. In diesem Falle werden sie durch Andrücken auf der Arbeitsplatte in die richtige Form gebracht. Besser ist es allerdings, ein kleines Bestückungswerkzeug, wie es heute im Handel überall zu haben ist, zu verwenden.

Fertig bestückte Platinen, die MOS ICs enthalten, sollten Sie, falls diese von Ihnen verschickt werden, in Alufolie einpacken.

Widerstände

Beachten Sie den Farbcode bei den Widerständen, wo in den Bausätzen Widerstandwerte von ... bis eingesetzt sind, sind die Werte wie z.B. bei den Pull Ups unkritisch. Metallfilmwiderstände haben andere Farbcodierungen wie Kohleschichtwiderstände.
Im Zweifel also messen!

Kondensatoren: Tantal und Elektrolyt-Kondensatoren müssen richtig gepolt eingesetzt werden. Tantal-Kondensatoren sind mit einem kleinen + bedruckt. Elektrolyt-Kondensatoren haben eine Eindellung am positiven Pol, nach dem man sich richten kann, falls kein Aufdruck vorhanden ist.

Dioden

Dioden und Zehnerdioden haben als Markierung einen Strich an der Kathodenseite. Auf dem Bestückungsbild ist die Diode ebenfalls mit einem Strich versehen. Beide müssen übereinstimmen.

Allgemeine Testhinweise

Vor Inbetriebnahme der Karte

1.) Prüfen Sie, ob alle IC's richtig bestückt und in der richtigen Richtung eingesteckt sind.

2.) Überprüfen Sie alle Bauteile auf richtige Plazierung.

3.) Legen Sie ein Amperemeter in die Stromversorgungsleitung. Das Minimalsystem benötigt ca. 2 Ampere.
Werden erheblich mehr gemessen, sofort ausschalten. In diesem Falle liegt ein Kurzschluß vor, ein IC ist defekt oder die Karte ist falsch bestückt. Denselben Vorgang sollten Sie bei den jeweiligen Systemerweiterungen vornehmen.

Nach Inbetriebnahme der Karte

1.) Spannungsversorgung an der Steckerleiste nach Bus-Belegung messen.

2.) Spannungsversorgung an den IC's messen.

Das Minimalsystem

Bei Computer-Kleinsystemen unterscheidet man zwischen Single-Board (Ein-Platinen-Systemen) oder Mehr-Platinen-Systemen.

Das Ein-Platinen-System ist in der Regel nicht ausbaufähig und kann nicht in ein 19 Zoll Einschubgehäuse eingebaut werden.

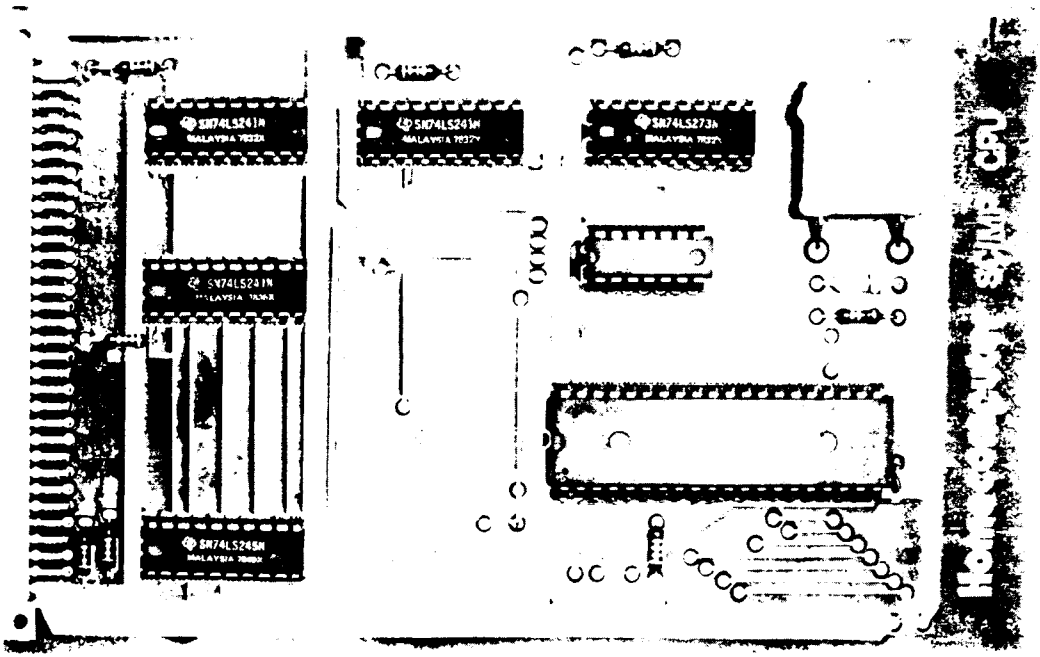
Im Gegensatz zu den in den Vereinigten Staaten üblichen Mehr-Platinen-Systemen mit dem S 100 Bus hat sich auf dem europäischen Markt ein Mehr-Platinen-System auf dem 64 pol Bus im Europa-Kartenformat durchgesetzt. Die einzelnen System-Platinen werden auf eine Bus-Platine (Motherboard) aufgesteckt. Hierauf befinden sich die Daten-, Adreß- und Steuerleitungen sowie die Stromversorgung. Die kleinstmögliche Ausbaustufe eines funktionsfähigen Systems bezeichnet man mit Minimal-System.

Das hier vorgestellte SC/MP Minimal-System muß aus folgenden Komponenten bestehen:

1. Bus-Platine
2. CPU-Platine
3. Speicher-Platine
4. Betriebsfirmware
5. Hexadezimale Ein- und Ausgabeeinheit
   und eine Stromversorgung.

Diese genannten Einzelbausätze werden Ihnen jetzt näher vorgestellt.

CPU-Platine



SC/MP CPU-Karte

Die CPU-Karte ist eine doppelseitig beschichtete und durchkontaktierte Karte im Europaformat, mit einer 64 pol. Steckerleiste nach DIN 41612.

Als CPU wird nur der SC/MP 600 (8060) eingesetzt.

Dies ist die schnelle SC/MP Version, die lediglich eine Spannung (5 V) benötigt. Die CPU enthält den Clockgenerator und wird mit einem Quarz von 2 MHz betrieben. Es kann allerdings auch die Beschaltung mit einem Kondensator erfolgen, falls eine geringe Frequenz erwünscht ist.

Alle CPU-Signale werden über Bustreiber zum System-Bus geführt. Da auch alle unsere weiteren Karten über solche Bustreiber verfügen, wird der Bus pro Karte nur mit einer LS-Last belastet. Dadurch können ca. 50 Platinen an das System angeschlossen werden, ohne daß eine Überlastung der CPU-Karte zu befürchten ist.

Die Bustreiber bestehen aus TTL's in LS-Ausführung (74 LS 245, 74 LS 241).

Stückliste

CPU neu

1 Platine
1 Steckerleiste 64 Pol

Integrierte Schaltkreise

IC 1	8060
IC 2, 3 und 6	74 LS 241
IC 4	74 LS 273
IC 5	74 LS 00
IC 7	74 LS 245

Widerstände

R 1 - 5	3 K 3 - 4 K 7
R 6 und R 7	47 K
R 8	1 K
R 9	100 K

Tantalkondensatoren

C 1, 3 und 5	22 my 16 V

Kondensatoren

C 2 nicht bestückt
C 4 56 p

Fassungen

1 DIL 40
5 DIL 20
1 DIL 14

1 Quarz 2 MHz

Erklärung CPU:

Als Bustreiber wird für die in eine Richtung gehenden Signale das IC 74LS241 eingesetzt. Dies ist ein 8 Bit-Tri-State-Leitungstreiber. Der max. Ausgangsstrom pro Bit beträgt 24mA.

Als Datentreiber wird der bidirektionale 8 Bit Tri-State-Leitungstreiber 74LS245 eingesetzt. Auch er hat ein fan out von 24mA. Der Bustreiber für die Daten ist im Normalfall immer zum Bus hin durchgeschaltet, erst mit dem NRDS (Lesesignal - negativ Read Data Strobe -) wird die Richtung, also zur CPU hin, für die Daten geändert.

Die während des NADS (negativ Adress Data Strobe) auf dem Datenbus liegenden Informationen werden durch eben diesen in ein Register (74LS273) übernommen.

Dies ist ein 8 Bit-Register mit CLEAR.

Aus Bit 7 dieses Registers wird die Halt Information gewonnen und durch eine LED auf der HEX I/O-Platine zur Anzeige gebracht.

Über einen Inverter kann dieses Signal zur CPU an den CONT-Eingang gelangen, wenn die Brücke entsprechend eingelötet worden ist. Gegebenfalls kann das Signal auch auf den System-Bus geschaltet werden.

Beim Minimal-System muß das Signal an den CONT-Eingang der CPU

gelangen, weil an diesem Eingang das Erkennen der "Halt Information" an die CPU gemeldet wird.

Wird der CONT-Eingang high, erkennt die CPU, daß der Halt-Befehl ausgeführt worden ist; er veranlaßt einen Software-Halt.

Dieser kann durch Betätigen der Halt-Taste wieder aufgehoben werden. Wie aus dem Schaltbild zu ersehen ist, schaltet die Halt-Taste nicht direkt gegen Masse. Es muß gewährleistet sein, daß auch zwei hintereinanderstehende Halts richtig erkannt werden.

Würde die Halt-Taste direkt gegen Masse schalten, wäre das Register für die Dauer des Tastendruckes gecleart (gelöscht), evtl. vorkommende Halt-Befehle würden nicht erkannt werden.

Deshalb darf der Clear-Eingang des Registers nicht statisch auf Masse geschaltet werden, sondern es muß beim Betätigen der Halt-Taste ein relativ kurzer Impuls erzeugt werden, der das Register löscht. Dies wird durch ein RC-Glied erreicht. Wird die Halt-Taste betätigt, gelangt über den vorgeschalteten Pull Up Widerstand 5 V an den Kondensator. Da dieser im ersten Moment noch nicht aufgeladen ist, ist für einen kurzen Augenblick Nullpotential am Clear-Eingang (Register wird gecleart). Dann lädt der Kondensator sich auf (High am Clear-Eingang). Wird die Halt-Taste wieder losgelassen, entlädt sich dieser relativ schnell über den Parallel-Widerstand.

Alle weiteren Daten des 8 Bit-Registers sind über Bus-Treiber auf den Bus geführt.

Für normalen Betrieb wird eine Brücke von A nach X eingelötet (Halt nach CONT).

Brücke von A nach Y (CONT der CPU auf System Bus).

Der auf der Platine befindliche 74LS00 dient als Inverter für das oben beschreibene CONT-Signal sowie als Inverter für den NADS und als Verknüpfungsglied für NRDS und NWDS.

Die Widerstände sind Pull Ups für die Leitungstreiber und zwei CPU-Eingänge (NBRQ und NHOLD).

Aufbauhinweise

Die Karte sollte unter Beachtung der allgemeinen Aufbauhinweise bestückt werden.

Testhinweise

Prüfen Sie die

1. Spannungsversorgung:
   + 5 V an 1. des Steckers
   Masse an 4, 16 und 32
   hiervon ist der Massepol 32 erforderlich.
2. der ICs.

Testen Sie den Taktgenerator. Die Schwingung messen Sie mit dem Oszilugraphen an den beiden Anschlüssen des Quarzes. An beiden Polen liegt eine Sinusschwingung gleicher Frequenz mit einer Amplitude von ca. 4 Vss.

Beim Drücken der N Taste auf der Hex I/O muß Pin 7 vom SC/MP von high auf low gehen. Pin 5 und 6 müssen auf high liegen, wobei an Pin 5 - wenn der Prozessor arbeitet - eine Schwingung anliegt.

Pin 8 muß auf high liegen (wenn auf low, steht der Prozessor und die LED auf der Hex I/O leuchtet). Pin 1 am 74LS273 muß auf high liegen.

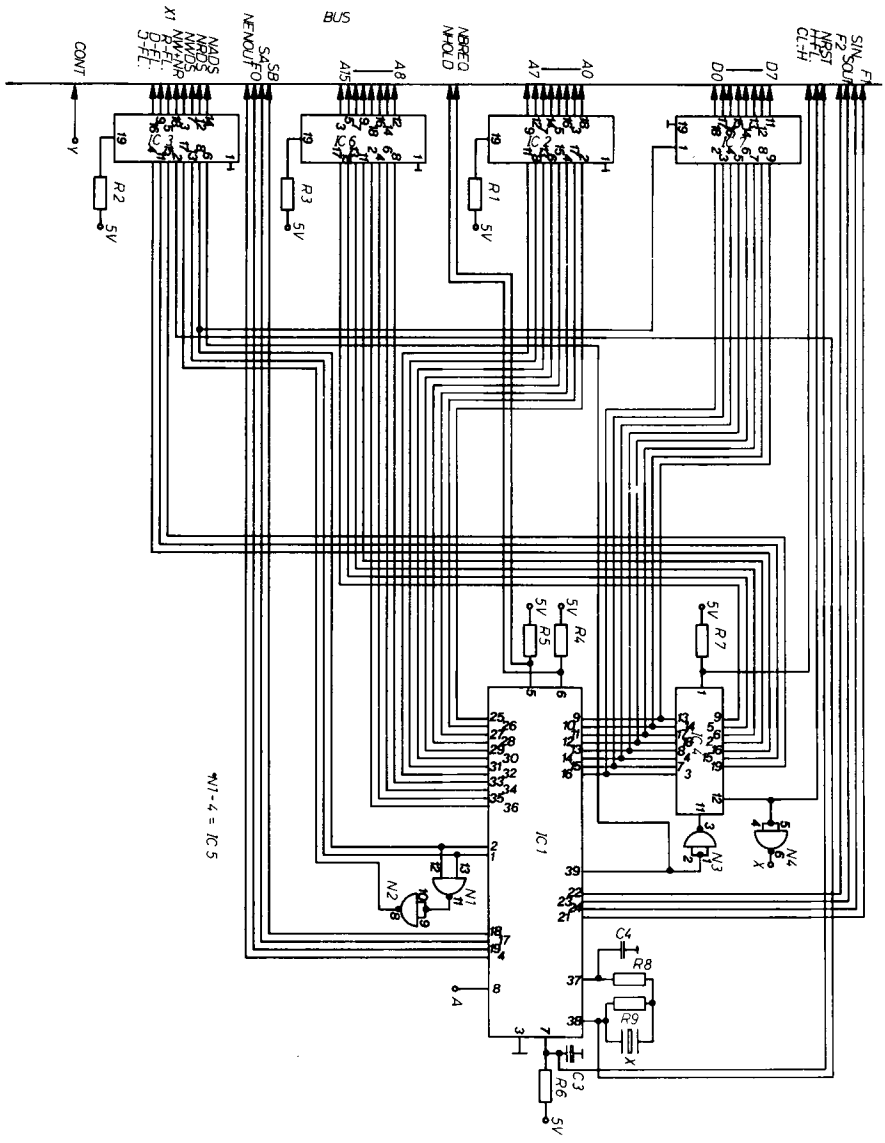
Arbeitet die CPU Karte nicht, ist eine Messung am Stecker auf evtl. vorhandene Kurzschlüsse erforderlich. Es wird jeder Pin gegen jeden Pin gemessen (beachten Sie die Busbelegung). Gleichartige Messungen sind sodann an den SC/MP Anschlüssen erforderlich. Keiner der Pins darf

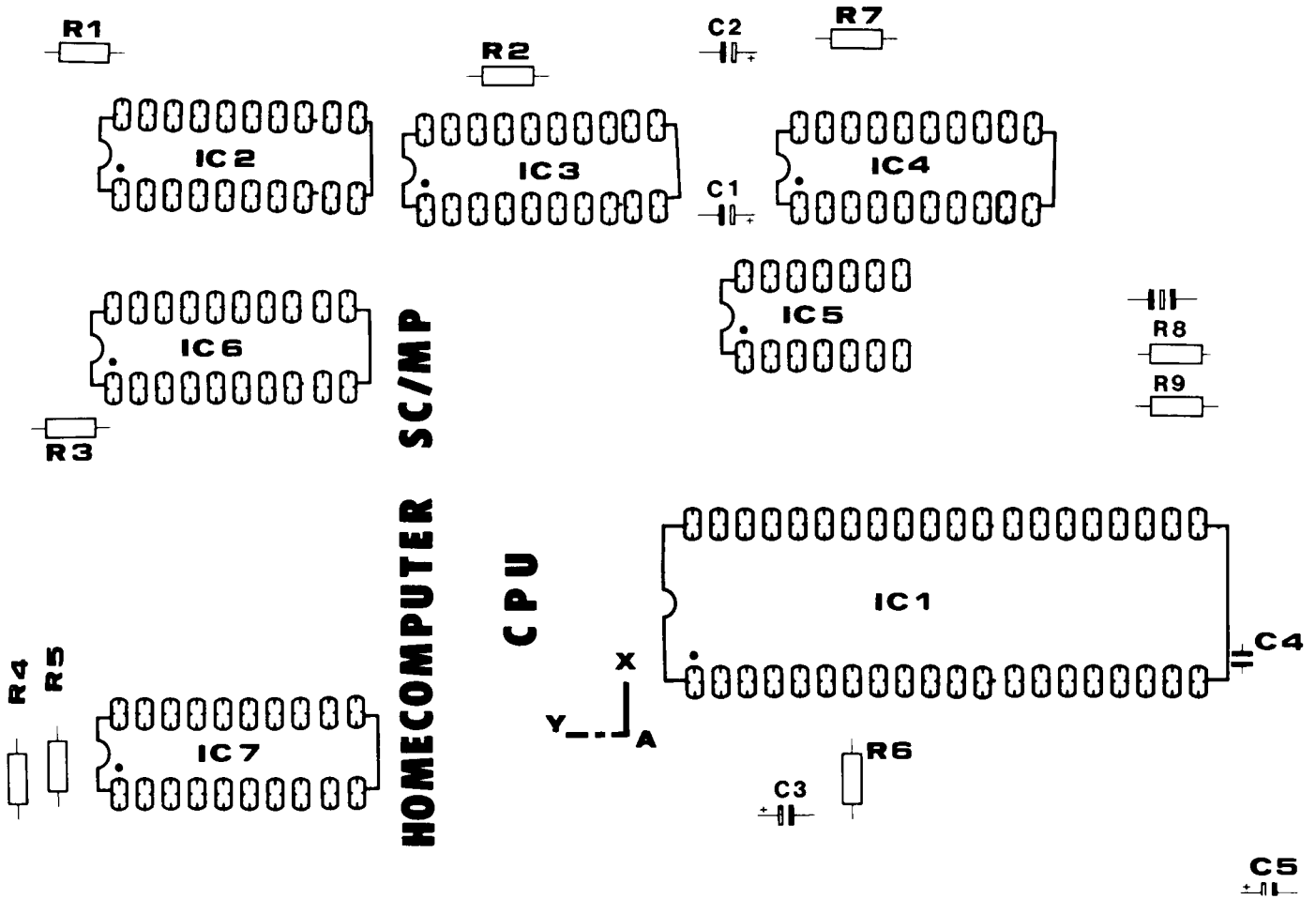
gegen einen anderen einen Kurzschluß aufweisen.

Die Leitungstreiber 74 LS 241 müssen ein high an Pin 19 haben. LS 245 Masse an Pin 19. Ebenso muß eine Masse an Pin 1 bei den 74 LS 241 vorhanden sein.

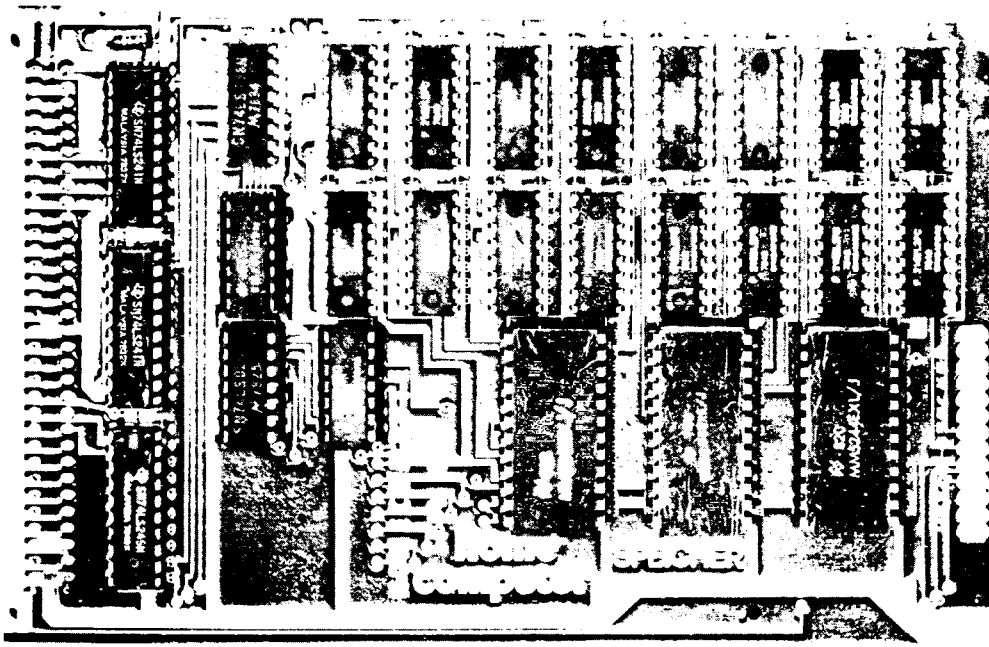
Adressen		Speicher	Übersicht
0000	-	05FF	Elbug PROMS
06X0	-	06X7	Mux Eingänge (nur bei Elektor System)
06XB	-	06XF	Prioritäts-Encoder (nur bei Elektor System)
07X0	-	07X7	HEX Output (7Segm. Display)
07XB	-	07XF	HEX Input / Tasten
0800	-	0BFF	1 K RAM (nur bei Homecomputer System)
0C00	-	0FFF	1 K RAM
1000	-	1FFF	Page 1 (muß für Interpreter bestückt sein)
2000	-	2FFF	und folgend 3000 bis 3FFF Seiten 2 - 7 können für Interpreter bestückt sein.
80XX			USART Karte
81XX			48 I/O Lines
8200	-	82FF	Prommer
8F00	-	8FFF	Nuber cruncher
9000	-	9FFF	frei
A000	-	B8E2	HSA-Assembler
C000	-	D3FF	HSB Interpreter
D400	-	EFFF	frei für Optionen
F000			Printer
FC00	-	FFFF	TV-Interface

Schaltbild und Bestückungsplan




R1
R2
C2
R7
IC2
IC3
IC4
C1
IC5
R8
R9
IC6
R3
HOMECOMPUTER SC/MP
CPU
IC1
C4
X
Y
A
R4
R5
IC7
R6
C3
C5

SC/MP ROM und RAM - Speicherkarte



Speicher-Karte

Die Speicher Karte für das SC/MP-System ist im Europaformat doppelseitig und durchkontaktiert und mit einer 64 pol. Steckerleiste (nach DIN 41612) versehen.

In der Minimalkonfiguration ist sie bestückt mit 1K RAM (2102 oder 21 L 02) dem ELBUG Betriebssystem in 3 ROMs von je 1/2 K-Byte vom Typ 5244 (kompatibel zum 5204). Als Option kann ein weiteres K-Byte RAM auf der Karte eingesteckt werden. Zum ELBUG-Betriebssystem siehe gesonderte Ausführung.

Die ROM Steckplätze können selbstverständlich auch mit anwendereigenen Programmen auf EPROM 5204 bestückt werden.

Stückliste

Bausatz Speicher neu

1 Platine
1 Steckerleiste 64 Pol
1 Wire-Wrap-Pfosten 26 Pol

Integrierte Schaltkreise

IC 1	74 LS 245
IC 2 und 3	74 LS 241
IC 4	74 LS 02
IC 5	74 LS 08
IC 6	74 LS 138
IC 7	74 LS 00
IC 8 - IC 15	2102 oder 2102L

(IC 16 - 23 2102 oder 2102 L - Option)
IC 24 ELBUG J
IC 25 ELBUG K
IC 26 ELBUG L

Widerstand

R 1	3 K 3 - 4 K 7

Tantalkondensatoren

C 1	Tantal 22 my 16 V

Fassungen

3 DIL 24
3 DIL 20
9 DIL 16
(8DIL 16 Option)
3 DIL 14

Als Option ist erhältlich:
Bestückung mit 2 K 21 L 02 und 8 weiteren
IC Sockeln.

Ebenfalls als Option ist das mit zwei 16 Pol Pfostenverbindern konfektionierte Kabel erhältlich.

Auf dieser Platine befindet sich ebenfalls die gesamte Adreßdecodierung des Minimal-Systems, einschließlich HEX I/O Adressen. Natürlich sind wieder alle auf der Karte benötigten Signale gepuffert und alle TTL's LS-Ausführung.

Am vorderen Ende der Platine befindet sich ein 26 poliger Wire-Wrap-Stützpunkt, an den unter Zuhilfenahme eines 26 poligen Pfostenverbinders mit Flachbandkabel die HEX I/O angeschlossen wird.

Adressenbelegung des Minimalsystems

0000	-	05FF	ELBUG
0700	-	0707	DISPLAY
0708	-	070F	HEX-TAST.
0800	-	0BFF	1 K RAM (Option)
0C00	-	0FFF	1 K RAM

Erklärung:

Auch hier werden als Bustreiber die IC's 74LS241, 245 eingesetzt. Die Adreßdecodierung übernimmt ein 74LS138, dies ist ein 1 aus 8 Decoder. Zur Decodierung werden die Adressen 9, 10, 11 herangezogen. Das IC besitzt zwei Enabel (Freigabe-) Eingänge, einer (G 1) aktiv high, der zweite (G 2 a, b) aktiv low. An G 1 ist die Verknüpfung der Adressen 12, 13, 14, 15 angeschlossen. Dies bewirkt, daß nur bei Seite Null freigegeben (enabelt) wird. Der endgültige CS (Chip selekt low = negative IC-Freigabe) wird erst durch den LS 138 und die Adressen 3 für das Display, 8 für die Tastatur gewonnen.

Die Umschaltung der Datenbuffer erfolgt, wenn die Seite Null angesprochen und das NRDS Signal kommt.

Auf den 26 pol. Pfosten sind Daten, Adressen 0, 1 + 2, NWDS, die beiden CS's, NRST, Halt-Taste und der Anschluß für das Halt LED herausgeführt. Dann ist die HEX I/O auch über Puffer mit dem System Bus verbunden.

Aufbauhinweise

Die Karte sollte unter Beachtung der allgemeinen Aufbauhinweise bestückt werden.

IC 1, 2 und 3 sind gegen die Richtung der anderen ICs bestückt. Der 26 pol. Wire Wrap Pfosten gehört auf die Seite , auf der sich die ICs befinden (Bestückungsseite).

Die ELBUG ROMs sind gekennzeichnet mit J, K und L und L IC 24 ist J, IC 25 ist K, IC 26 ist L.

Testhinweise

Prüfen Sie die Spannungsversorgung:

1. am Stecker
   an 1 liegt + 5 V
   an 32 liegt Masse
   an 3 liegt - 12 V
2. an den ICs
3. am 26 poligen Stützpunkt gemäß Anschlußbelegung.

Anschluß Pin 19 an IC 1, 2 und 3 müssen auf high liegen. Pin 1 liegt auf Masse.

Sollte die Karte nicht funktionieren, messen Sie Steckerleiste und Daten und Adressenleitungen hinter den Bustreibern auf Kurzschlüsse durch.

Zum Testen der RAM setzen Sie die einzelenen Datenbits, und zwar:

Hexadezimal	01	für Datenbit 0
	02	für Datenbit 1
	04	für Datenbit 2
	08	für Datenbit 3
	10	für Datenbit 4
	20	für Datenbit 5
	40	für Datenbit 6
	80	für Datenbit 7

Z.B. MO0800 01 jetzt muß auf den beiden rechten Datendisplays 01 stehen. Ist dies nicht der Fall, ist das IC 15 defekt oder es bestehen Kurzschlüsse zwischen Daten und Adressen. Ein Kurzschluß kann ausgeschlossen werden, wenn die anderen Datenbits richtig ausgelesen werden.

Spezielle Testhinweise für Speicherkarten

Alle Speicherkarten könen entweder über Brücken oder über Schalter auf eine bestimmte Page festgelegt werden. Im SC/MP-System ist die Page 0 bereits belegt, die Verwendung dieser Seite ist also unzulässig. Es ist ebenso nicht möglich, zwei Speicherkarten auf die gleiche Seite zu legen, da diese sich gegenseitig stören würden. Prüfen Sie also vor Inbetriebnahme einer Speicherkarte, daß diese auf eine bisher unbelegte Seite eingestellt ist.

Prüfen mit Modify

Der gewählte Speicherbereich muß über Modify angesprochen werden können. Bei RAM-Karten muß unter der gewählten Adresse eine Datenänderung möglich sein. Bei ROM-Karten kann der Adresseninhalt ausgelesen werden. Beim Überschreiben bleibt die Date erhalten.

CHIP-Select-Test

Programm A für RAM-, Ausgabe und Eingabekarten

Programm B für ROM-Karten

Nach dem Start des Programms wird auf der Karte in einem dauernden Zyklus geschrieben oder gelesen. Dies erzeugt ein Signal (Cip-Select) für die im Programm gewählten Adressen. Die Verfolgung dieses Signals ersehen Sie bitte aus dem jeweiligen Schaltplan.

Programm A

```
C4      XX
        31
C4      YY
        35
        C9-00
90      FD
```

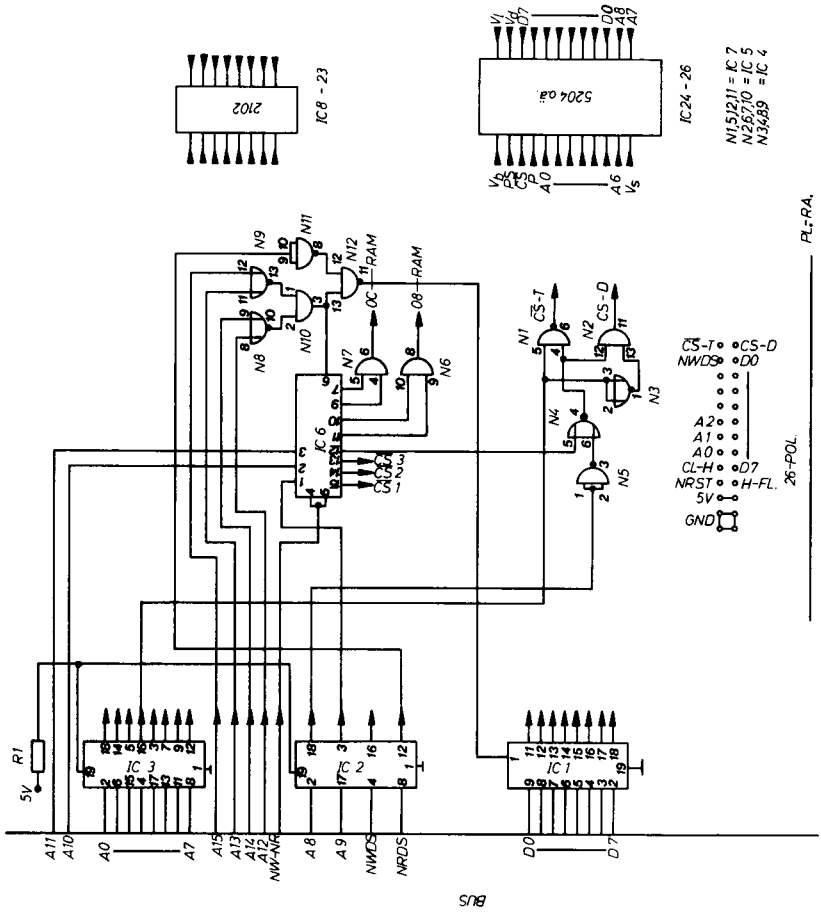
Programm B

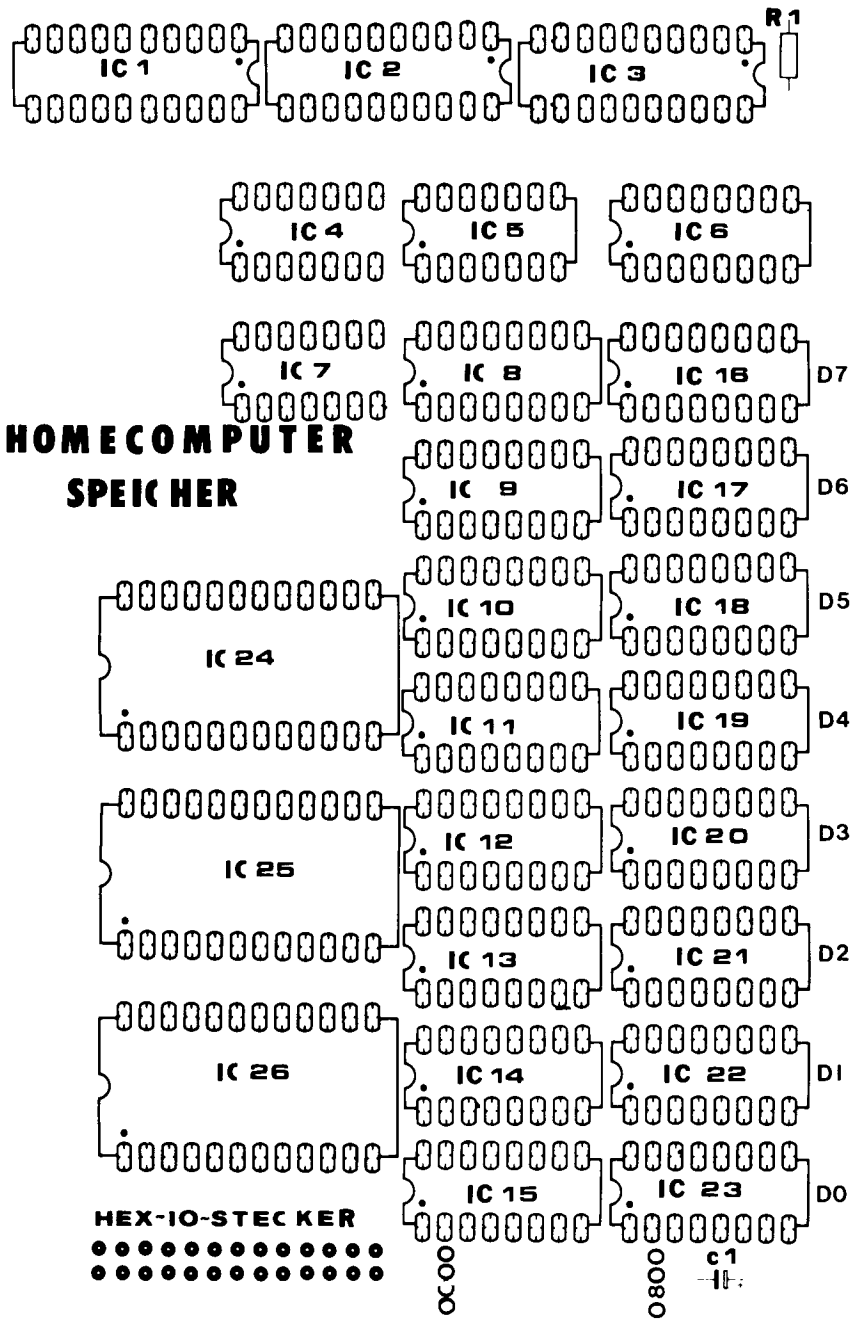
```
C4      XX
        31
C4      YY
        35
        C1 - 00
90      FD
```

XX = lower Byte
YY = higher Byte

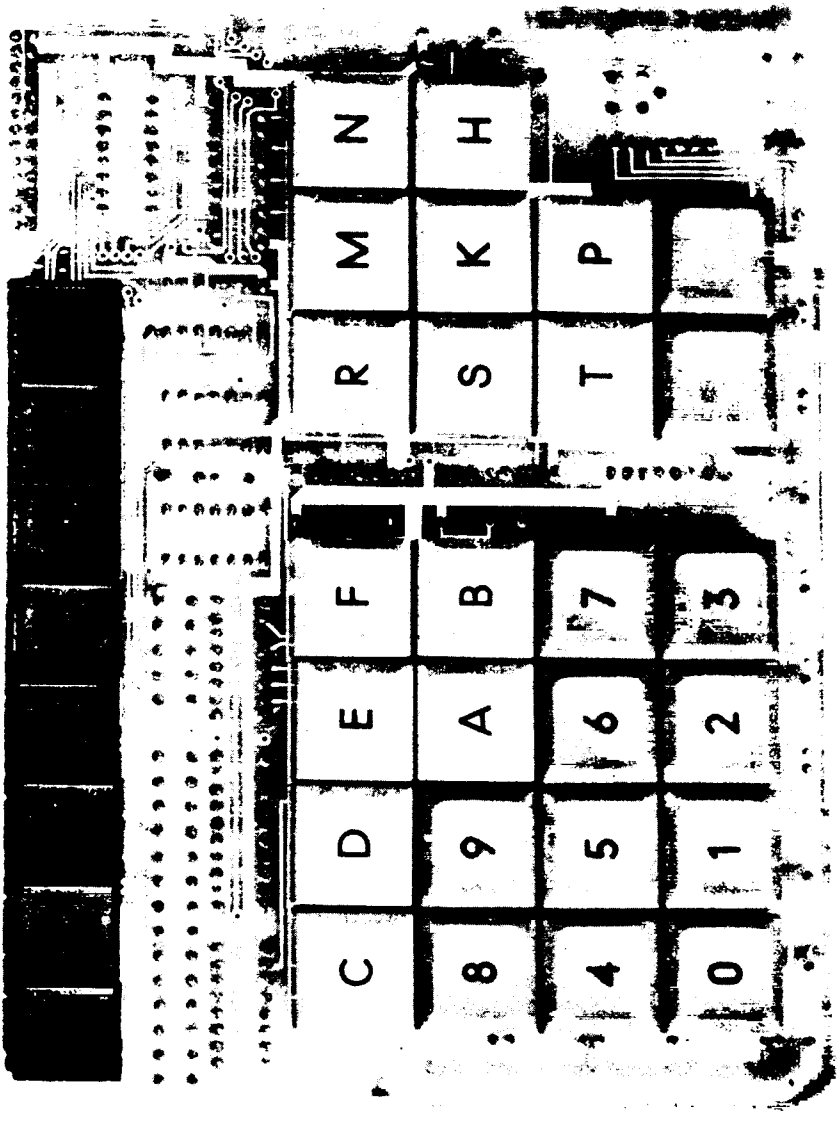
Inbetriebnahme des Minimalsystems

Nachdem die Platinen sorgfältig aufgebaut wurden, wird die Stromversorgung am BUS angeschlossen. Das Minimalsystem benötigt + 5 V 1,5 A und - 12 V 200 mA. Die CPU und die Speicherkarte werden auf den BUS aufgesteckt. Die HEX I/O wird mit dem Flachkabel mittels der Pfostenverbinder mit der Speicherkarte verbunden. Nach dem Einschalten erscheint auf dem Display eine Anzeige zufälliger Konstellation. Das Halt-LED leuchtet. Nach Drücken der Taste "N" setzt der Prozessor alle Register zurück und beginnt bei 0000 zu arbeiten. Jetzt drücken Sie die Taste "H", das Halt-LED erlischt und auf dem Display erscheint "ELBUG"; ist dies nicht der Fall, beachten Sie die Testhinweise für Einzelbausätze. Erscheint ELBUG, entnehmen Sie die weitere Bedienung den Ausführungen zum Betriebssystem oder den Software-Paketen.


2102
IC8 - 23
5204 o.ä.
IC24 - 26
N1,5,12,11 = IC 7
N2,6,7,10 = IC 5
N3,4,8,9 = IC 4
PL-RA.
26-POL.
RAM
OC
CS-T
CS-D
NWDS
D0
A2
A1
A0
CL-H
D7
NRST
H-FL.
5V
GND
IC 3
IC 2
IC 1
R1
BUS


R1
IC 1
IC 2
IC 3
IC 4
IC 5
IC 6
IC 7
IC 8
IC 16
D7
HOMECOMPUTER
SPEICHER
IC 9
IC 17
D6
IC 10
IC 18
D5
IC 24
IC 11
IC 19
D4
IC 12
IC 20
D3
IC 25
IC 13
IC 21
D2
IC 26
IC 14
IC 22
D1
IC 15
IC 23
D0
HEX-IO-STECKER
C1

SC/MP Hexadezimale Ein- und Ausgabe



HEX I/O

Die hexadezimale Ein-/Ausgabeeinheit für das SC/MP Minimal-System befindet sich auf einer doppelseitigen, durchkontaktierten Platine mit den Maßen 136 x 178,5 und ist mit einem 26 pol. Wire Wrap Stützpunkt versehen. Mittels eines mit einem 26 pol. Pfostenverbinders und Flachbandkabel wird diese Karte an die Speicherplatine angeschlossen. Als Eingabe werden normale Schließer-Tasten verwendet. Diese sind sehr flach, haben einen kurzen Hub, sind leicht zu betätigen und werden geprägt geliefert. Als Ausgabe dienen 8 Stck. FND 507, 7 Segment Displays.

Alle auf der Karte befindlichen TTL's sind in LS-Ausführung.

Stückliste

Hex I/O neu

1 Platine
1 Wire-Wrap Pfosten 26 Pol

Integrierte Schaltkreise

IC 1	74 LS 138
IC 2 und 7	74 LS 241
IC 3	74 LS 157
IC 4	74 LS 93
IC 5	74 LS 14
IC 6 und 8	7489
IC 9	74 LS 00
IC 10, 11 und 12	74 LS 148

Transistoren

T 1 - T 8	TUP
T 9	TUN

Widerstände

R 1 - R 27 und R 48 - R 52 und R 61 - R 64 und R 30	3 K 3 - 4 K 7
R 28	180
R 29	47 K
R 31, 33, 35, 37, 39, 41, 43, 45	820
R 32, 34, 36, 38, 40, 42, 44, 46,	470
R 47	330
R 53 - R 60	47 - 82 (Helligkeit
R 65	100 der Displays)

Tantalkondensatoren

C 1 und 2	22 my 16 V

Kondensatoren

C 3	10 p
C 4	1 n
C 5	100 n

Fassungen

2 DIL 20
7 DIL 16
3 DIL 14
8 Displays FND 507 o.ä.
26 geprägte Tasten
1 LED

Erklärung:

Jede Taste ist mit einem PULL UP Widerstand versehen (also ist in Ruhestellung der Anschluß "high"). Ist eine Taste gedrückt, wird dieser "low". Die Codierung der Tasten erfolgt über einen 74LS00 und über 3 Stck. 74LS148. Dies sind binäre 8 zu 3 Bit Prioritätsencoder, welche 8 Eingänge besitzen. Wird einer der Eingänge auf "low" gelegt, gibt das IC eine 3 Bit-Information heraus. Außerdem können mehrere caskadiert (hintereinander) geschaltet werden. Dies ist auch hier der Fall. Die Caskadierung erfolgt so, daß bei jedem Tastendruck eine 8 Bit Information

entsteht. Die 8 Bit Information gelangt auf einen Tri-State Puffer (LS 241), welcher durch den Tasten-CS auf der Speicherkarte freigegeben wird.

Ausgabe:

Die acht 7-Segment-Displays werden im Multiplexverfahren angesteuert. Ein Oszillator, aufgebaut mit einem Widerstand, einem Kondensator und einem Inverter, läßt einen Zähler (74LS93) immer bis 8 zählen. Der Ausgang des Zählers ist eine 3 Bit Information. Diese gelangt einerseits zu einem 1 aus 8 Decoder (LS 138) und andererseits zu einem Multiplexer. In Abhängigkeit von der 3 Bit Information wird einer der 8 Ausgänge des 74LS 138 low, und einer der 8 Transistoren steuert das entsprechende Display "auf". Über den Multiplexer gelangt die 3 Bit Information an die RAM's (7489), welche die gespeicherte 7-Segment-Information an die Displays geben. Als Segmenttreiber dient ein Leitungstreiber (LS 241).

Wird in die RAM's eine neue Information eingeschrieben, schaltet der Multiplexer, gesteuert durch den Display Chip Selekt, um. Nun gelangen die Adressen 0,1 + 2 vom Systembus an die Adresseingänge der RAM's. Mit dem NWDS-Signal kommen die Daten in die entsprechenden Speicherzellen. Nachdem die neuen Daten in die RAM's übernommen worden sind, schaltet der Multiplexer wieder in Normalstellung umundder Inhalt der RAM's wird zur Anzeige gebracht.

Diese Vorgänge laufen so schnell ab, daß sie für das Auge nicht wahrzunehmen sind, unter der Voraussetzung, daß die Multiplex-Frequenz einige kHz beträgt. Ist sie zu klein, flimmert die Anzeige.

Aufbauhinweise

Die Hex I/O ist von zwei Seiten bestückt. Von der einen Seite (B Seite) werden alle Rs Cs und IC Fassungen und auch der 26 Pol Wire Wrap Stützpunkt bestückt. Von der anderen Seite werden die Displays und die Tasten eingesetzt.

Beginnen Sie mit den liegenden Widerständen R 1 - R 29. Die restlichen

Widerstände werden stehend eingelötet. Wenn diese Seite fertig bestückt und gelötet ist, werden die Tasten von der anderen Seite eingesetzt. Setzen Sie erst alle Tasten ein, richten Sie diese aus und löten erst dann. Neben den Tasten wird die LED eingesetzt, und zwar Anode nach unten (an 5 V). Die abgeflachte Seite des LED Gehäuses ist die Kathode.

Die Displays können auf 24 Pol Fassungen (nicht im Bausatz enthalten), welche quer eingesetzt werden, aufgesteckt werden.

Testhinweise

Prüfen Sie die Spannungsversorgung am Stecker (siehe Steckerbelegung) und an allen ICs.

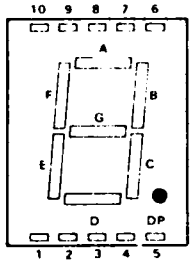
Die erste Prüfung findet am Oszillator IC 5 Pin 14 statt. Hier muß ein Signal von einigen kHz anliegen. Dieses Signal verfolgen am Ausgang Pin 1, 8 und 9 des gleichen ICs. Hier liegen drei verschiedene Signale an. Diese gehen nach IC 7 und IC 8. An den Ausgängen des IC 7 (Pin 7, 9, 10, 11, 12, 13, 14, und 15) liegen die zur Aufsteuerung der Displays dienenden Rechtecksignale. An den Ausgängen des IC 8 (Pin 4, 4 und 9) liegen die Adressen für die RAMs 7489. An den Ausgängen der RAMs, welche über Pull Ups an high liegen, müssen die ausgelesenen Dateninformationen der RAMs liegen. Über den LS 241 gelangen die Daten an die Displays.

Prüfen Sie Datenein- und Ausgänge, ob Kurzschlüsse vorliegen. Jede Taste muß in Ruhestellung an high liegen. Bei Tastendruck muß an den Eingängen des LS 241 eine 8 Bit Information anliegen (IC 2).

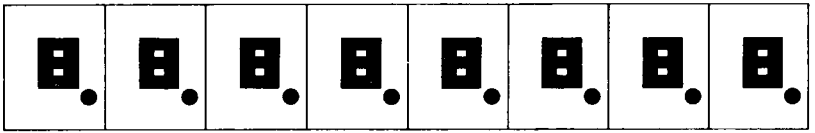
Anschlußbelegung des Displays

•FND507•

PIN CONNECTIONS
(FRONT VIEW)



PIN	FND507/567	FND500/560
1	Segment E	Segment E
2	Segment D	Segment D
3	Comm-Anode	Comm-Cathode
4	Segment C	Segment C
5	Decimal Point	Decimal Point
6	Segment B	Segment B
7	Segment A	Segment A
8	Comm-Anode	Comm-Cathode
9	Segment F	Segment F
10	Segment G	Segment G

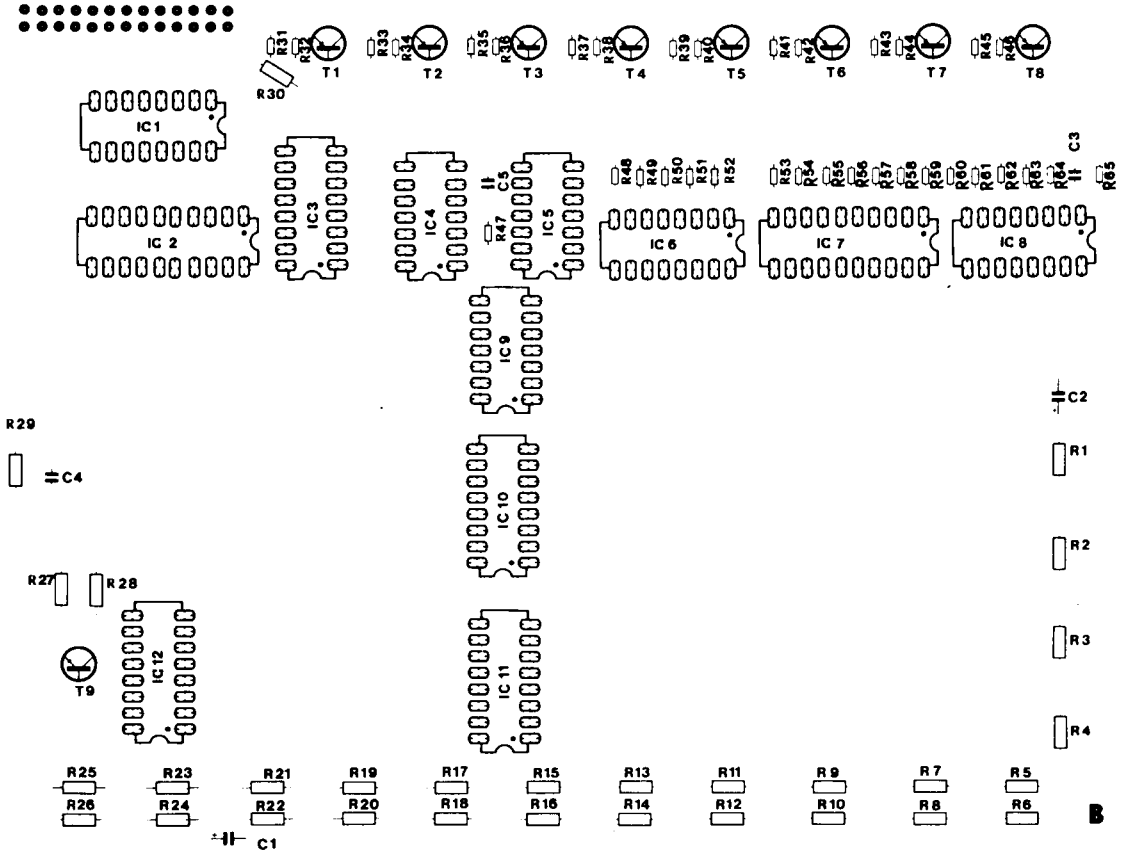


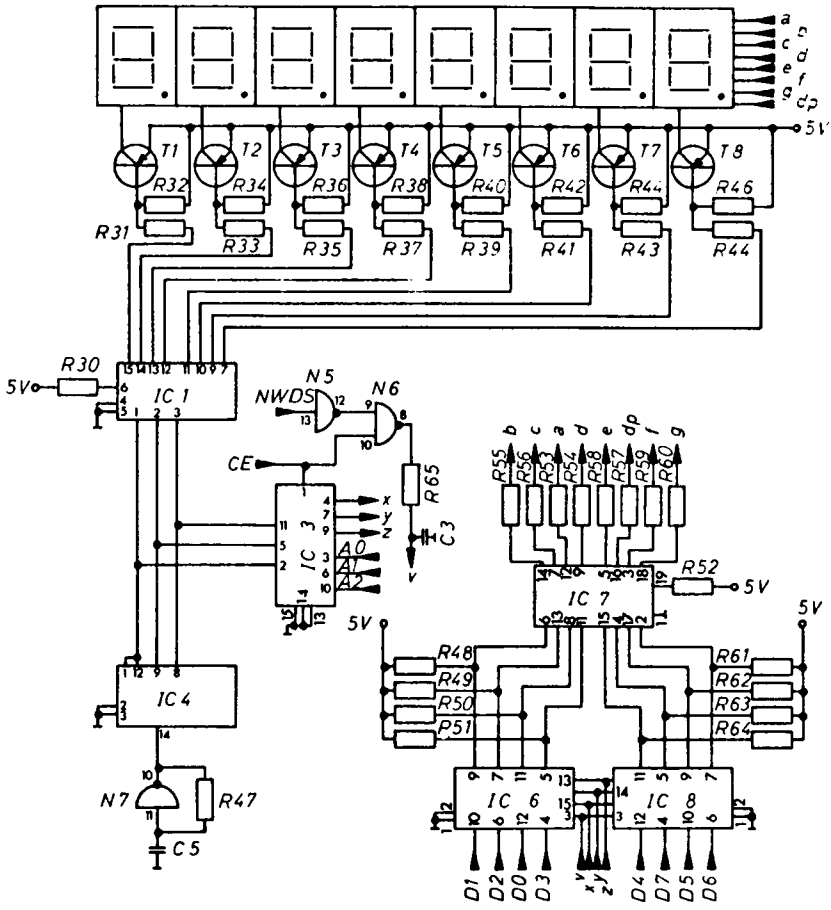
HOMEC OMPUTER HEX--I-O

C	D	E	F
8	9	A	B
4	5	6	7
0	1	2	3

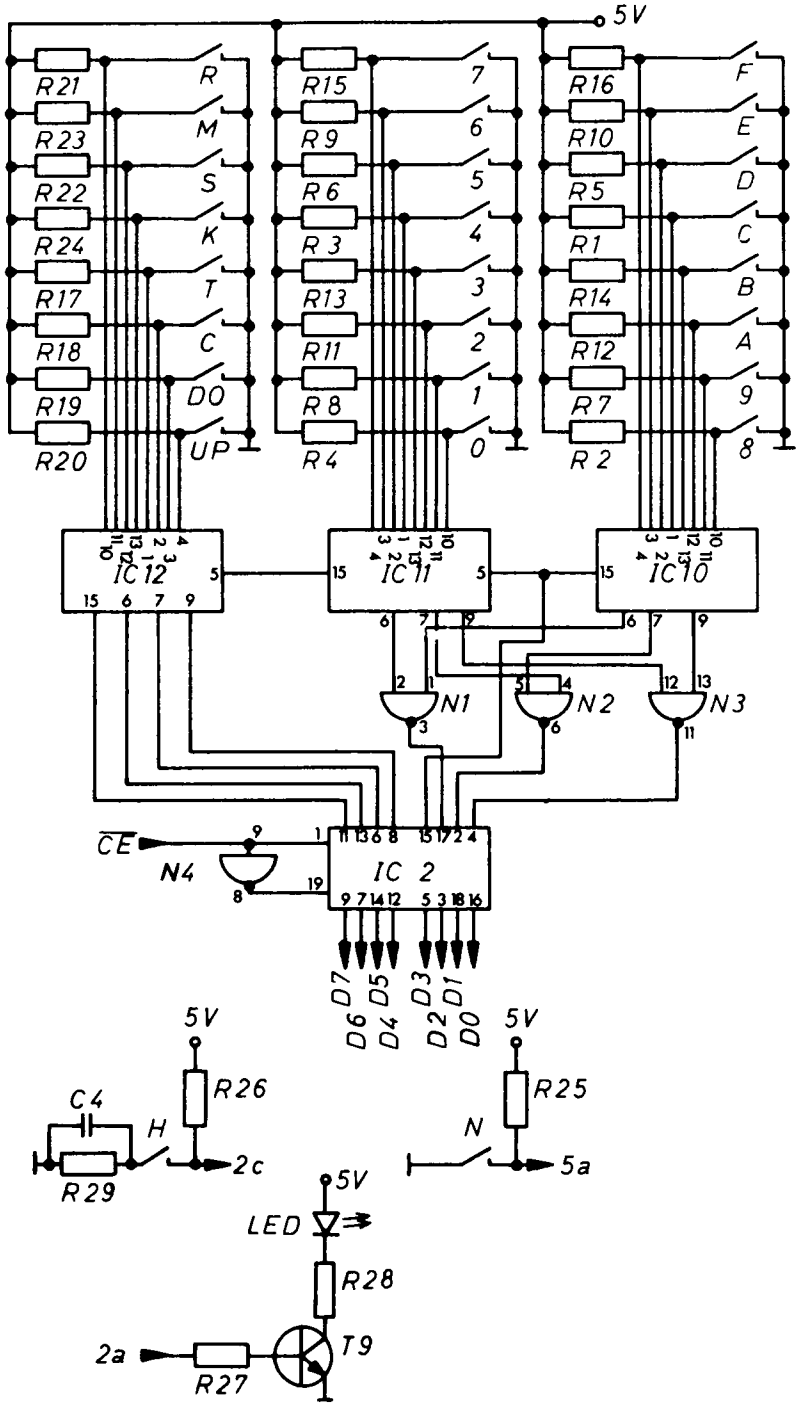
R	M	N
S	K	H
T	C	
V	Λ	

A


R31
R32
T1
R33
R34
T2
R35
R36
T3
R37
R38
T4
R39
R40
T5
R41
R42
T6
R43
R44
T7
R45
R46
T8
R30
IC1
IC 2
IC3
IC4
C6
R47
IC5
R48
R49
R50
R51
R52
IC 6
R53
R54
R55
R56
R57
R58
R59
R60
R61
R62
R63
R64
C3
R65
IC 7
IC 8
IC9
C2
R29
C4
R1
IC10
R2
R27
R28
IC12
T9
IC 11
R3
R4
R25
R23
R21
R19
R17
R15
R13
R11
R9
R7
R5
R26
R24
R22
R20
R18
R16
R14
R12
R10
R8
R6
C1
B



N1 - N3, N6 = IC 9
N4, N5, N7 = IC 5


5V
R21
R23
R22
R24
R17
R18
R19
R20
R
M
S
K
T
C
DO
UP
R15
R9
R6
R3
R13
R11
R8
R4
7
6
5
4
3
2
1
0
R16
R10
R5
R1
R14
R12
R7
R2
F
E
D
C
B
A
9
8
IC 12
IC 11
IC 10
N1
N2
N3
CE
N4
IC 2
D7
D6
D5
D4
D3
D2
D1
D0
C4
R29
H
R26
2c
N
R25
5a
LED
R28
T9
2a
R27

BUS

Die bisher beschriebenen drei Bausätze stellen das SC/MP Minimalsystem dar. CPU und Speicher werden über einen BUS miteinander verbunden. Bei Bestellung eines Minimalsystems wird ein kleiner BUS mit drei Steckmöglichkeiten (bestückt mit 2 Buchsenleisten) geliefert. Dieser reicht für kleine Anwendungen aus. Er kann noch mit ein bis zwei weiteren Buchsenleisten bestückt werden (z.B. Cassetten-Interface und 4 K-Speicherkarten). Abb. a

Neben diesem kleinen BUS gibt es noch den großen (Abb. b) HC-BUS mit 10 Steckmöglichkeiten. Beide Systeme können (Abb. c) über Stecker und Adapterleiste miteinander verbunden werden (Abb. d), so daß eine Erweiterung des Systems leicht möglich ist. Gleichfalls ist eine Verbindung über 64 pol. Litzenleiter konfektioniert mit zwei 64 pol. Pfostenverbindern möglich (Abb. e). Die gewünschte Länge des Kabels muß bei Bestellung angegeben werden.

Die Adaptierung muß durch Aufstecken auf die Wire-Wrap Pins der Buchsenleisten auf der Rückseite der Platine erfolgen. Beachten Sie die richtige Verbindung 1 mit 1 und 32 mit 32. A und C ist dann in jedem Falle richtig belegt.

Selbstverständlich kann auf den kleinen BUS verzichtet und auch für das Minimalsystem bereits ein großer BUS verwendet werden.

Der HC-10-pol.-BUS

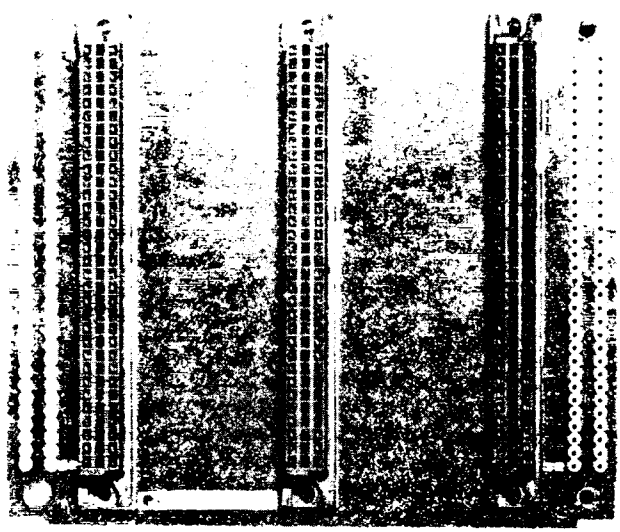
Die Platine hat die Maße 270 x 115. Die Abstände der Steckplätze betragen 1 Zoll. Somit kann die Platine leicht in 19 Zoll Gehäuse einge-

baut werden. Die erforderlichen Bohrungen sind auf der Platine markiert.

Neben den 10 Steckplätzen hat diese BUS-Platine Anschlußmöglichkeiten für Tantalkondensatoren. Die Rückseite ist die gemeinsame Masse gleichzeitig als Abschirmung , die an jeweils zwei Steckplätze gelegt werden kann. Die Durchkontaktierung erfolgt durch Einlöten eines Drahtes.

An der einen Seite des Prints sind die 64 Anschlüsse zusätzlich herausgeführt zum Bestücken mit einer Steckerleiste. Die andere Seite hat verdrehte A + C Leisten, damit hier eine Adapterleiste eingelötet werden kann. Falls die BUS-Platine nicht in ein Gehäuse eingebaut werden soll, ist es möglich, die Euro-Karten in sogenannte Raks einzuschieben (Abb. f). In diesem Falle ist bei der Bestückung darauf zu achten, daß die Buchsenleiste zuerst an das Rak angeschraubt und dann mit der Platine verlötet wird. Dies ist erforderlich, damit der für das Rak benötigte Abstand vom Print eingehalten wird. Gegebenenfalls kann das Rak nach dem Einlöten der Buchse wieder entfernt werden.

Abb. a



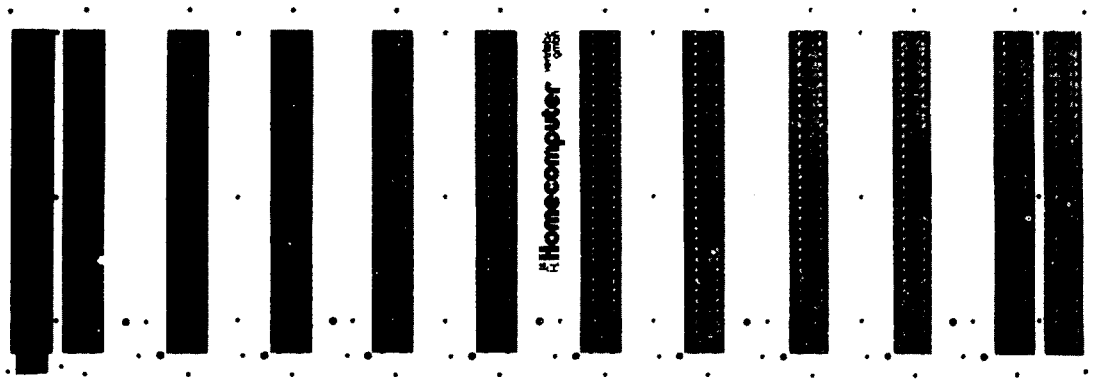


Abb. c

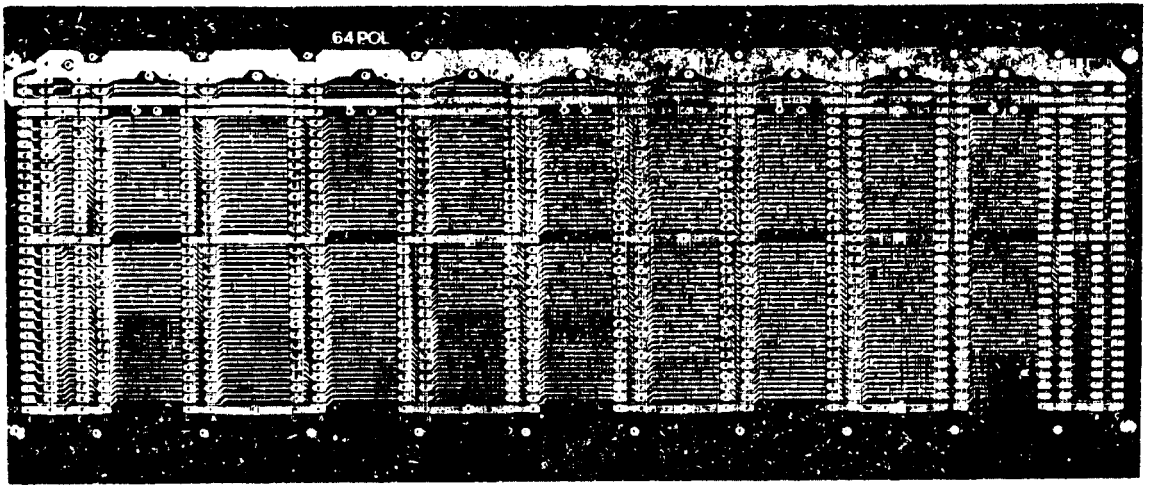


Abb. b

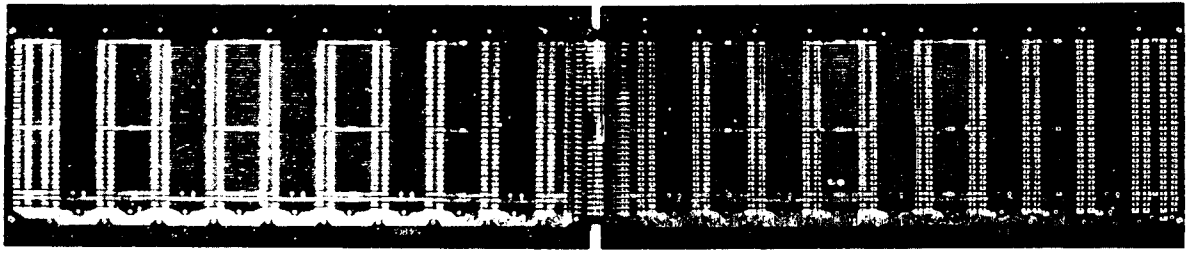
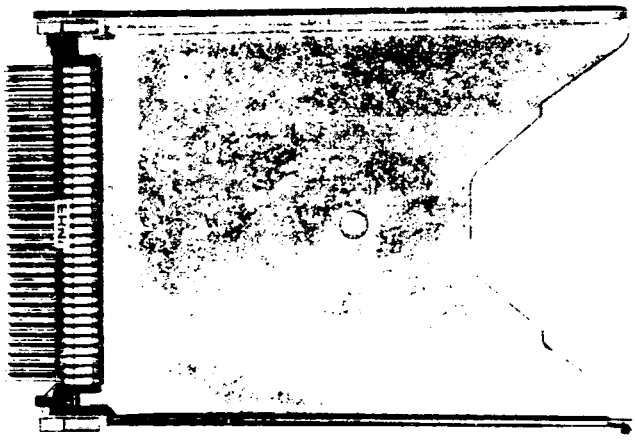


Abb. d

Abb. e

Abb. f



BUS-Belegung

A		C
+5V	1	+5V
H-Flag	2	Clear Halt
- 12V	3	- 12V
GND	4	GND
NRST	5	NHOLD
NBREQ	6	NC
DB 1	7	DB 0
DB 3	8	DB 2
DB 5	9	DB 4
DB 7	10	DB 6
NC	11	CONT
SB	12	SA
SOUT	13	SIN
F 1	14	F 0
D-Flag	15	F 2
GND	16	GND
+12V	17	+12V
NC	18	NENOUT
AD 15	19	AD 14
AD 13	20	AD 12
AD 11	21	AD 10
AD 9	22	AD 8
AD 7	23	AD 6
AD 5	24	AD 4

AD 3	25	AD 2
AD 1	26	AD 0
X 1	27	NWDS+NRDS
NC	28	I-Flag
R-Flag	29	NC
NADS	30	NC
NWDS	31	NRDS
GND	32	GND

Adapter

Die Verbindung von Computer-Platinen und Peripheriegeräten ist in der Regel vielpolig. Das Verlöten einzelner Drähte erfordert nicht nur einen erheblichen Arbeitsaufwand sondern verursacht auch bedingt durch Kapazitäten und Induktivitäten meistens Störungen.

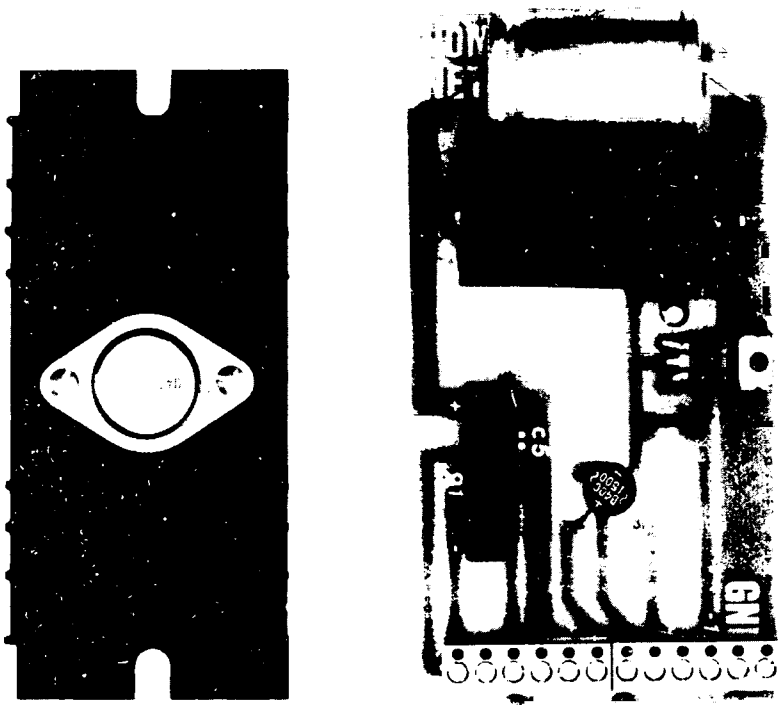
Die erforderlichen Verbindungen bedingen neue Verfahren die dem Bastler mit herkömmlichen Schaltungen nicht vertraut sind. Für die Verbindung stehen Fachkabel mit verschiedener Polzahl sogenannte Litzenleiter zur Verfügung.

Auf diese Litzenleiter können Pfostenverbinder aufgepresst werden. Diese sind so konstruiert, daß kleine Messerchen in die Kabel einschneiden und einen guten Kontakt herbeiführen. Litzenleiter und Pfostenverbinder gibt es in unterschiedlicher Polzahl wobei in der Regel solche mit 2 reihigen Kontakten verwandt werden.

Für die Steckerleisten einer 64 poligen Platinenausführung sind 64 polige Pfostenverbinder erforderlich.
Eine weitere Verbindungsmöglichkeit besteht, wenn ein Pfostenverbinder auf eine eingelötete Steckerleiste sog. Wrapstützpunkte aufgesteckt wird.
Dies ist in der Regel bei 26 poligen Verbindungen der Fall. Zum Dritten können Litzenleiter mit einem aufgepressten Leiterbahnstecker versehen werden. Leiterbahnstecker gibt es in DIL 14, 16, 24 und 40 Ausführung. Diese passen in die Norm-IC-Fassungen.

Zum Betrieb dieses Minimalsystems ist ein MP-Netzteil erforderlich. Für denjenigen, der beim Minimalsystem bleiben will, genügt ein kleines Netzteil. Wer jedoch Erweiterungen vorsieht, sollte gleich eine umfangreichere Stromversorgung einplanen.

Netzteil K



Beschreibung

Das Netzteil befindet sich auf einer einseitig beschichteten Karte mit den Maßen 130 mm x 70 mm.

Ausgelegt ist es für 5V/3A und - 12V/800mA, also genau richtig für ein Mikroprozessor-Minimalsystem, plus ein oder zwei weitere Karten.

Auf einer Seite der Karte wird eine Schraubklemmleiste eingelötet. Von hier können alle erforderlichen Verbindungen zum MP-System erfolgen.

Der 5 V Stabi befindet sich auf einem externen Kühlkörper, der mitgeliefert wird. Des weiteren befinden sich auf der Karte Entstörkonden-

satoren, die das Schwingen der Stabis verhindern.

Erklärung

Die Gleichrichtung und Siebung ist in konventioneller Weise aufgebaut. Der Stabi für die - 12 V ist ein 7912, der zusätzlich durch zwei Kondensatoren gegen Schwingneigung abgeblockt ist.

Der 5 V Stabi ist ein 78 H 05, der, wie gesagt, außerhalb der Platine auf einen Kühlkörper montiert ist. Auch er erhält einen Entstörkondensator auf der Karte.

Aufbauhinweise

Beachten Sie die allgemeinen Aufbauhinweise.

Testhinweise

Sollte der 78H05 trotz der schon vorgenommenen Maßnahmen schwingen, so ist direkt am IC der Eingang mit 100n gegen Masse abzublocken.

Stückliste

1 Kühlkörper (gebohrt)
1 Platine
2 6 pol. Schraubklemmen

Integrierte Schaltkreise

IC 1	7912
IC 2	78 H 05

Gleichrichter

G 1	B40/C3000
G 2	B40/C1500

Elkos

C 1	4700my/25V
C 2	2200my
C 3	2,2my

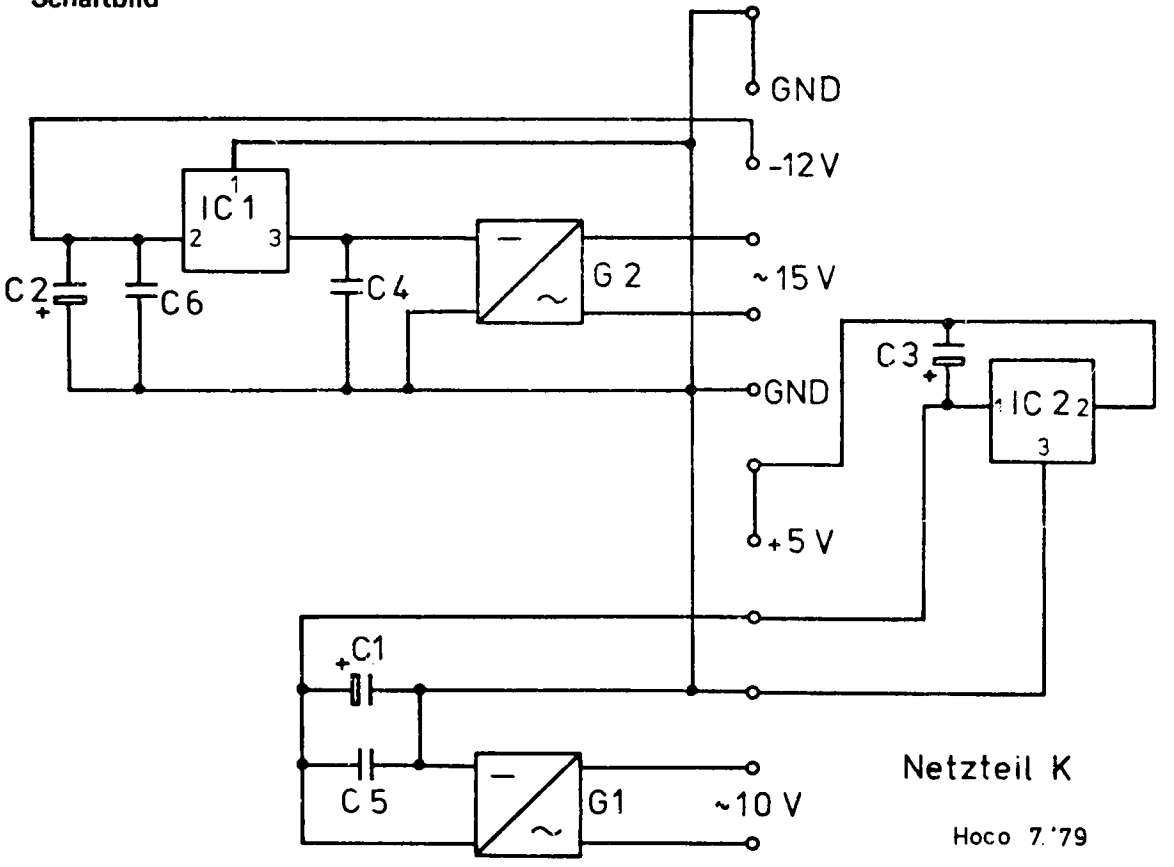
Kondensatoren

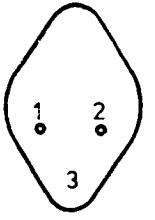
C 4,5	100n

Option

Trafo 9V/3A
15V/1A

Schaltbild

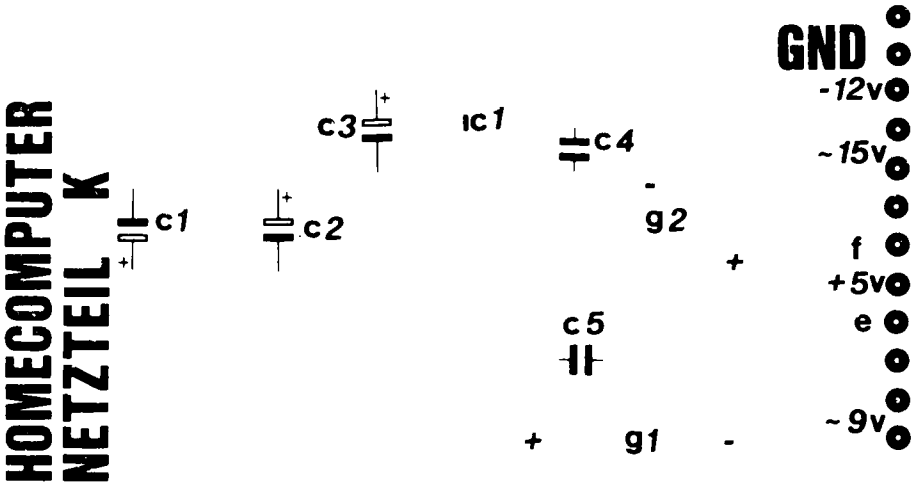




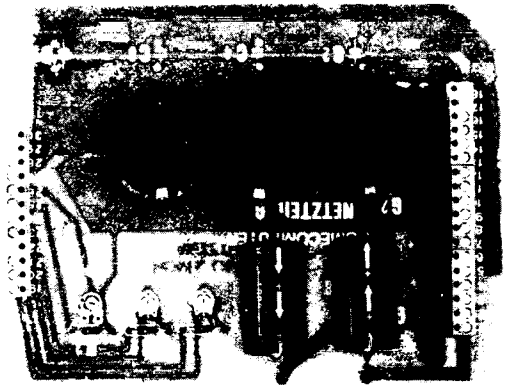
Netzteil K

Hoco 7.'79

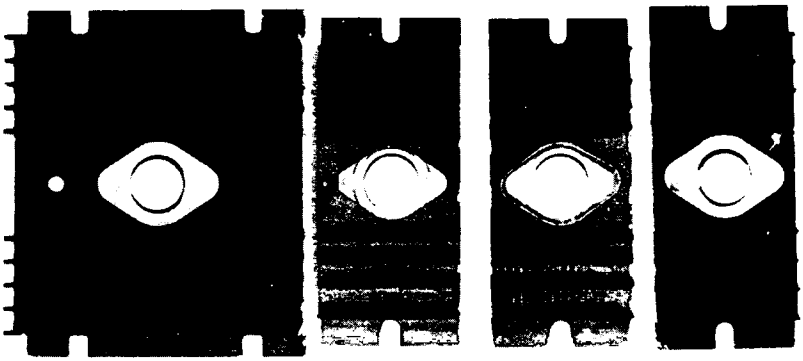
Bestückungsplan



Netzteil G



10 A Ausführung



Beschreibung

Das HC Netzteil G befindet sich auf einer einseitig beschichteten Karte mit den Maßen 180 mm x 140 mm.

Es liefert in der Basisversion folgende Spannungen und Ströme: +5V/10A, - 5V/1A, +12V/1,5A und - 12V/1,5A, wobei - 5V und - 12V aus einer gemeinsamen Wicklung der Trafos gespeist werden.

Durch einfaches Hinzufügen eines weiteren 78 P 05 (mit Zubehör) kann die +5V Belastbarkeit auf 20A erhöht werden.

Auf der Platine befinden sich Potis, mit denen die Spannungen genau eingestellt werden können.

Erklärung

Alle stabilisierten Gleichspannungen werden durch integrierte Spannungsregler erzeugt. Diese sind kurzschlußfest und gegen thermische Überlastung geschützt.

+5V

Der Ringkerntrafo besitzt 2 Wicklungen von 9V/10A, welche parallel zum 25A Gleichrichter geführt werden. Die ungesiebte Gleichspannung wird von 3 (7) Spezial-Elkos-mit einer Kapazität von je 15.000myF - geglättet. Diese geglättete Spannung liegt am Eingang des Reglers an, welcher zwischen seinem Ausgang und Masse eine Spannung von 5 V einstellt. Durch "Hochlegen" der Masse kann am Ausgang des Reglers auch eine Spannung eingestellt werden, die größer als 5V ist. Dies dient dazu, einen Spannungsabfall auf den Leitungen zum Mikroprozessorsystem auszugleichen.

- 5V

Diese Spannung wird in konventioneller Weise mit einem 7905 erzeugt.

- 12V/+12V

Nach Glättung und Siebung der beiden 15V Wechselspannungen des Trafos gelangen die Gleichspannungen an die Eingänge der Regler 78HG und 79HG. Diese IC hat noch 3 weitere Anschlüsse, den Ausgang, Masse und einen Steuereingang. Ein Teil der Ausgangsspannung gelangt über einen Spannungsteiler an den Steuereingang. Das IC ist nun bestrebt, zwischen Masse und diesem Steuereingang eine Spannung von 5V einzustellen. Über ein Poti ist es nun möglich, das Verhältnis des Spannungsteilers und damit die Ausgangsspannung des Reglers einzustellen.

Bei allen Reglern kann der Aus- und Eingang durch einen 1my Tantal direkt am IC überbrückt werden, falls der Regler schwingen sollte.

Aufbauhinweise

Zunächst sind alle Stabis, unter Zuhilfenahme des mitgelieferten Isoliermaterials, auf die Kühlkörper zu montieren (den 78P05 auf großen Kühlkörper). Für die beiden 12V Regler müssen noch jeweils 2 Löcher in die Kühlkörper gebohrt werden. Der 25A Gleichrichter wird ebenfalls auf den großen Kühlkörper montiert. Anhand der Anschlußbilder sind Tantals und Widerstände direkt an die Stabis zu löten. Anschließend ist die Platine nach Bestückungsplan aufzubauen. Nun werden folgende Verbindungen zwischen Platine, Reglern und Trafo hergestellt.

Achten Sie auf die ordnungsgemäße Verdrahtung der Stabis, da falsches Anschließen zur sofortigen Zerstörung des Reglers führt.

Ebenso muß darauf geachtet werden, daß genügend große Kabelquerschnitte verwendet werden.

Anschluß Schraubklemme

1,2	an Eingang 7905, 79HG
3,4	Wechselspg. 15V/2,5A von Trafo (blau/blau)
5,6,7,11 - 16	Masse 79HG (1), 78HG (4), 7905 (3), 24AGL (-)
8	an Eingang Stabi 78HG
9,10	Wechselspg. 15V/1,5A von Trafo (weiß/weiß)
17,18	von 25 A Gleichrichter (+Pol) und an 78 Po5 Eingang
19	an Anschluß 2 79HG
20	an Anschluß 3 79HG
21	Ausgang - 12V
22	an Anschluß 3 78HG
23	an Anschluß 2 78HG
24	Ausgang + 12V
25	an Ausgang 78 P 05
26	Ausgang +5V/bis 3A (größere Ströme direkt am Stabi entnehmen)

27	an Masseanschluß (beider 78 P 05
28	unstab. +5V
29	an Ausgang 7905
30	Ausgang - 5V

In die 220 V Versorgungsspannungsleitung des Ringkerntrafos sollte eine träge Sicherung von 2 A eingebaut werden.

Anschluß Trafo

220 V	gelb/gelb
15V/2,5A	blau/blau
15V/1,5A	weiß/weiß
2x9V/10A	(grün/rot) x 2

Testhinweise

Bevor Sie das Netzteil einschalten, überprüfen Sie noch einmal gründlich die Bestückung der Platine (pol. der Elkos) und die Verdrahtung mit den Stabis.
Schalten Sie das Netzteil nach diesen sorgfältig ausgeführten Kontrollen zunächst ohne Verbraucher ein.

Achten Sie auf die 220 V Versorgungsspannung des Gerätes.

Messen Sie nun die einzelnen Ausgangsspannungen an den angegebenen Punkten. Sollte eine Spannung nicht vorhanden sein, schalten Sie das Gerät sofort aus.

+5V

Legen Sie Ihr Meßgerät an die +5V Spannung und stellen Sie diese mit dem Poti P3 auf genau 5V ein. Danach verfahren Sie ebenso mit den Spannungen +12V (P2), - 12V (P1).

Sollte eine der Spannungen sich nicht einstellen lassen, schalten Sie das Gerät sofort aus. Messen Sie nun noch die - 5V Spannung.

Schalten Sie das Gerät nun aus. Bitte, beachten Sie die nicht ungefährliche Ladung der Elkos, die auch nach einiger Zeit nach dem Aus-

schalten noch vorhanden ist.

Schließen Sie einen Verbraucher an die verschiedenen Spannungen an. Messen Sie die Versorgungsspannung direkt am Verbraucher und stellen die Sollwerte ein, um eventuelle Spannungsabfälle auf den Zuleitungen auszugleichen.

Sollte das Netzgerät nicht ordnungsgemäß arbeiten, liegt ein Verdrahtungs- oder Bestückungsfehler vor, der bei den Stabis sofort zur Zerstörung führt.

Stückliste

1 Platine
5 6 pol. Schraubklemmen

Integrierte Schaltkreise

IC 1	78 P 05
IC 3	78 HG
IC 4	79 HG
IC 5	7905

Halbleiter

G 3	Gleichrichter 25A
G 2	Gleichrichter 3,2A
G 1	Gleichrichter 5A

Widerstände

R 1 - 4	6,8K
R 5	2,2K
P 1,2	4,7K
P3	100

Kondensatoren

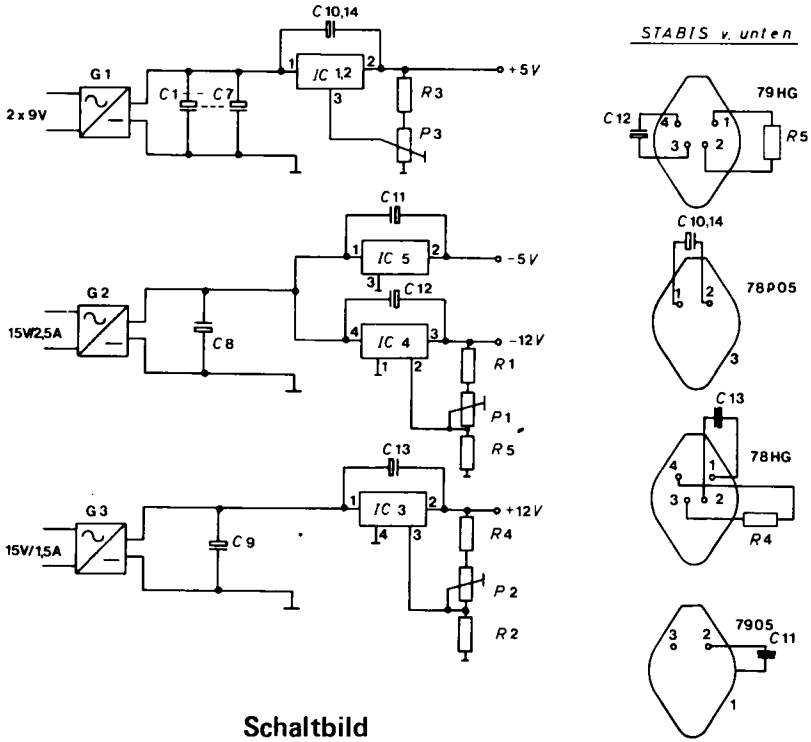
C 1 - 3	15.000 my/16V
C 8,9	2.200 my
C 10 - 13	1my Tantal

Sonstiges

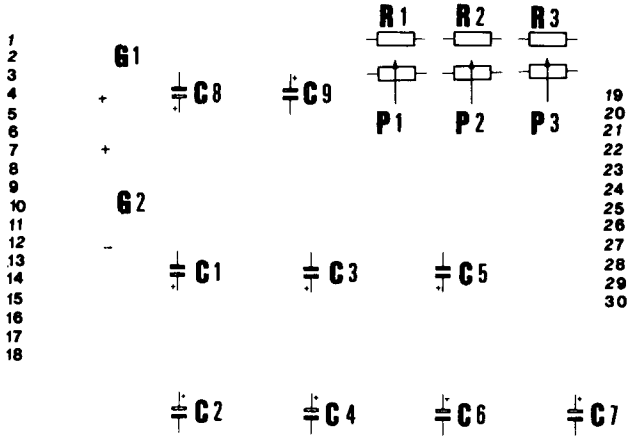
4 Kühlkörper
4 Isoliermaterial

Option

1 Kühlkörper
IC 2 78 P 05
1 Isoliermaterial
C 14 1my Tantal
C 4 - 7 15.000/16V
1 Ringkerntrafo 2x9V/10A, 1x15V/2,5A, 1x15V/1,5A
1 Befestigungsmaterial für Trafo



Schaltbild



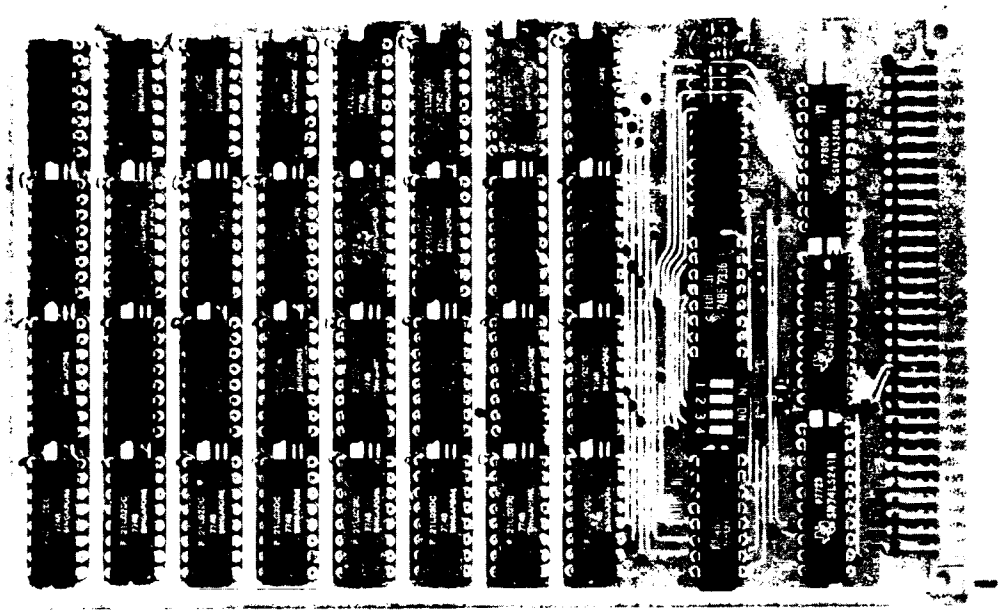
Die erste erforderliche Erweiterung ist eine Speicherkarte.

Hier stehen statische oder dynamische Speicher zur Verfügung.

Diese Speicherkarten dienen zur Arbeitsspeicher-Erweiterung (hier wird das Programm abgelegt). Der Processor schreibt unter einer Adresse eine Date ein, die jederzeit wieder ausgelesen werden kann. Mit dem SC/MP können 64K Byte = 65535 Adressen angesprochen werden. Beim Abschalten der Spannung verlieren diese RAMs ihre Information.

4K RAM-Speicher

für 2102 oder 21 LO 2



4K Platine

Die 4K RAM Karte ist im Europaformat und mit einer 64 pol. Steckerleiste versehen, passend zum SC/MP-System.

Sie ist bestückt mit dem RAM 2102, welches zu 1024 x 1 BIT organisiert ist. Somit kann die Karte K-weise bestückt werden.

Mittels eines DIL 4 Schalters kann die Adresse der Karte eingestellt werden (um Page 1 einzustellen, wird A 12 auf off (high) belassen, die anderen 3 Adressen auf on (low) geschaltet).

Natürlich sind - wie bei allen unseren Karten - alle auf der Karte benötigten Signale gepuffert und alle TTL's in LS-Ausführung.

Auch die RAM's können in LS-Ausfürung mit 450ns geliefert werden, so daß sie dann nur noch ca. halb soviel Strom benötigt wie bei normalen 4K Karten.
Stromaufnahme der Karte in Normalausführung: 1,25 A in LS Ausführung 0,75 A.

Stückliste

4K RAM Karte 2102 oder 2102 L

1 Platine
1 Steckerleiste 64 Pol

Integrierte Schaltkreise

IC 1 - IC 32	2101 oder 2102 L
IC 33	74 LS 138
IC 34	74 LS 85
IC 35	74 LS 00
IC 36 und 37	74 LS 241
IC 38	74 LS 245

Widerstände

R 1 - 6	3 K 3 - 4 K 7

Tantalkondensatoren

C 1 - C 8	22 my 16 V

Fassungen

3 DIL 20
34 DIL 16
1 DIL 14
1 x 4 Bit-Schalter

Erklärung

Als Bustreiber werden auch hier die IC's 74 LS 241, 245 eingesetzt. Der 2102 ist zu 1024 x 1 Bit organisiert; dies hat zur Folge, daß nur 4 Chip-Selekts benötigt werden. Diese werden wieder durch den 1 aus 8 Decoder (74 LS 138) decodiert.

Die vier höherwertigen Adressen und ein 4 Bit DIL Schalter werden über einen 4 Bit Vergleicher (74 LS 85) verknüpft, der bei Gleichheit (Adresse, die vom Bus "anliegt", und eingestellte Adresse am DIL 4 Schalter) ein high Signal liefert. Dies bewirkt ein enablen des 1 aus 8 Decoders, der dann seinerseits einen seiner 8 Ausgänge auf low schaltet. Erst wenn die Karte angesprochen wird, also wenn der 4 Bit Vergleicher ein Gleich erkennt und ein NWDS oder NRDS kommt, wird der Datenpuffer "niederohmig", sonst ist er TRI-STATE. Die Umschaltung des Datenbuffers erfolgt direkt durch das NWDS Signal.

Aufbauhinweise

Zum Aufbau siehe allgemeine Aufbauhinweise.

Testhinweise

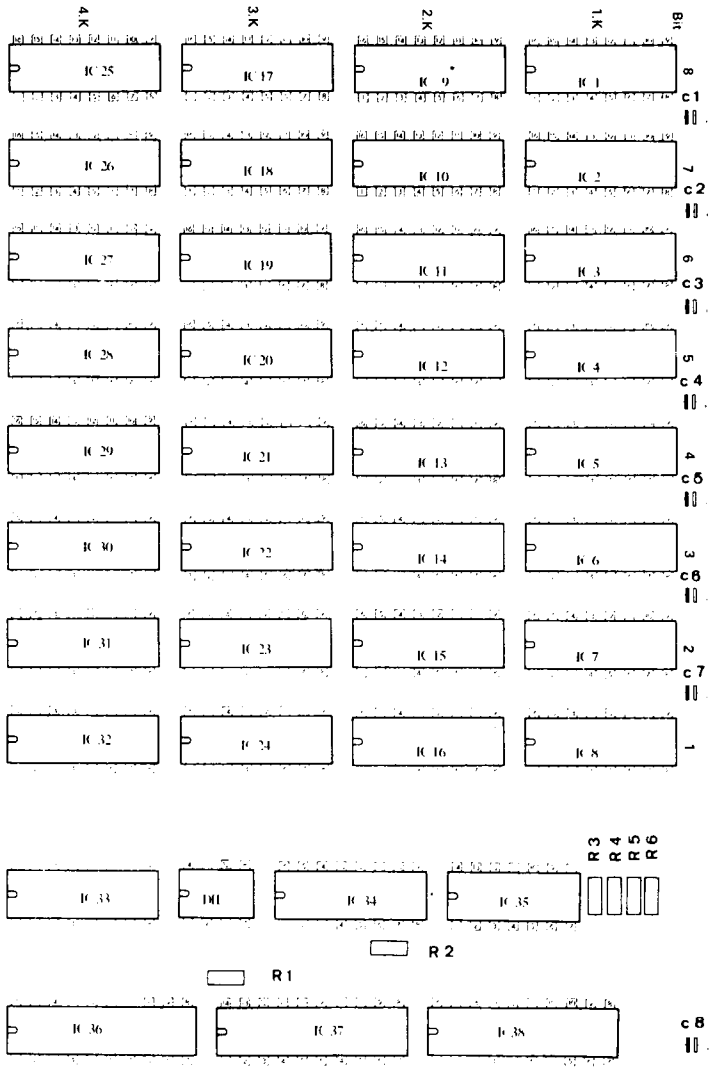
Zum Test siehe allgemeine Testhinweise für Speicherkarten.

Zur Prüfung der Karte über Modify = M.
Page 0 = alle 4 Bit auf On (unzulässig)
Page 1 = A 12 auf 1
Page 2 = A 13 auf 1
Page 3 = A 12 + A 13 auf 1 und fortlaufen im BCD-Code. Über Modify "M" kann ein erster Test erfolgen. Drücken der Taste "M" Adresseneingabe ist X000.

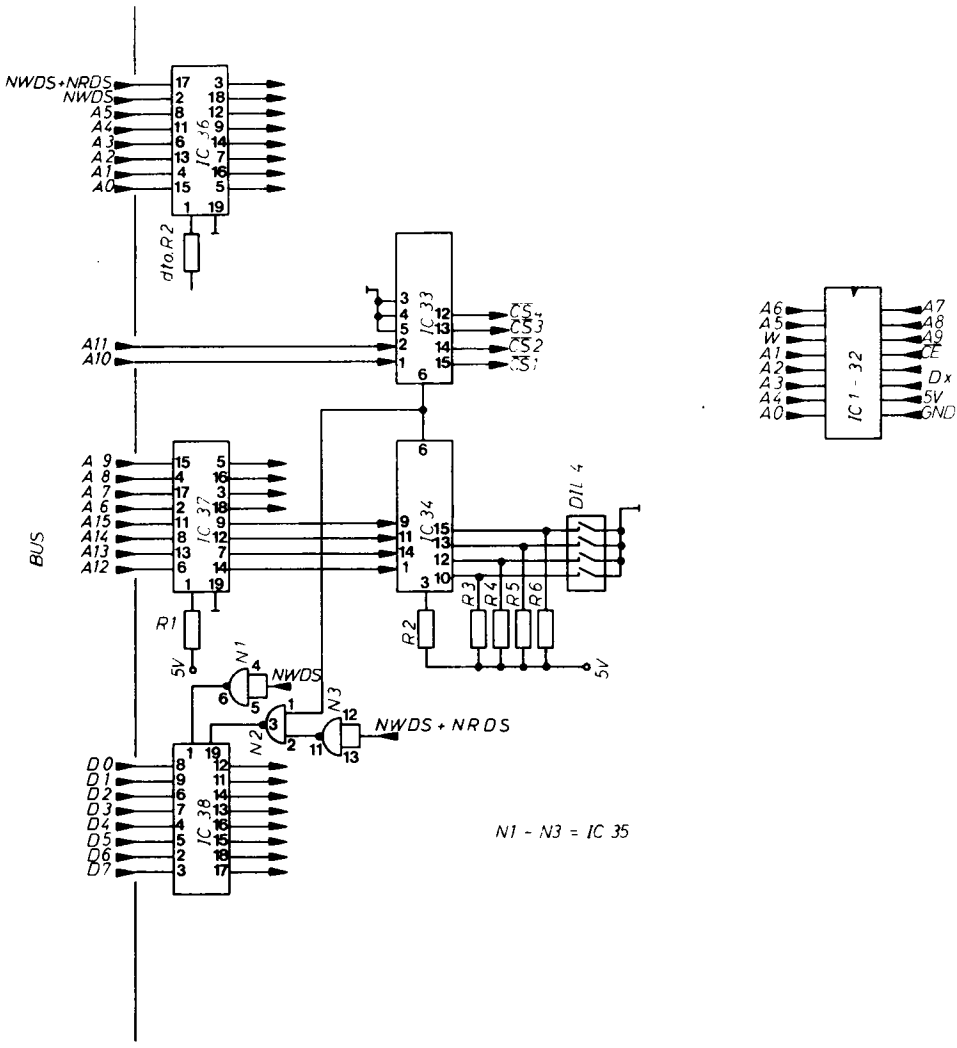
Date ändern

Das Display muß jetzt X000 XX zeigen. Falls die gewünschte Änderung nicht möglich, wird auf die erwähnten Testhinweise für Speicherkarten verwiesen. Das Durchtesten einer 4K-Karte per Modify wäre eine umständliche und langwierige Arbeit.
Hierfür gibt es das 4 K Diagnoseprogramm

BESTÜCKUNGSPLAN



SCHALTPLAN


NWDS+NRDS
NWDS
A5
A4
A3
A2
A1
A0
IC 36
dto.R2
IC 33
CS4
CS3
CS2
CS1
A11
A10
A9
A8
A7
A6
A5
A4
A3
A2
A1
A0
IC 37
IC 34
DIL 4
BUS
R1
R2
R3
R4
R5
R6
5V
N1
N2
N3
NWDS
NWDS + NRDS
D0
D1
D2
D3
D4
D5
D6
D7
IC 38
N1 - N3 = IC 35
A6
A5
W
A1
A2
A3
A4
A0
IC 1 - 32
A7
A8
A9
CE
Dx
5V
GND

16K dynamische RAM

Die 16K dynamische RAM-Karte ist im Europaformat und mit einer 64 pol Steckerleiste versehen passend zum HC MP-System. Ein zweiter 64 pol Steckerleistenplatz ermöglicht die leichte Adaption an andere Busbelegungen.
Sie ist bestückt mit dem RAM 4116,welches zu 16K x 1 Bit organisiert ist.
Das Refreshm der RAMs erfolgt auf der Karte unabhängig von externen Steuersignalen.
Mittels eines 4 Bit-DILSchalters kann die Anfangsadresse der 16K in 4K-Abständen eingestellt werden.
Die RAMs benötigen folgende Spannungen:+5V, +12V -5V.
Die Stromaufnahme für 16K Byte RAMs beträgt: +5V/300mA, +12V/ 350mA, -5V/70mA.
Alle TTLs in LS-Ausführung. Der Bus ist nur mit einer LS, Mos-Last belastet.

Erklärung

Der 4116 ist 16K x 1 organisiert, deshalb wird nur ein Shipselect benötigt. Um die Adressen in 4K Abständen wählen zu können, müssen vier Adressen verglichen werden (A12 - A15), obwohl für die RAMs schon A12 - A13 verwendet werden. Das Problem wurde mit einem 4 Bit Volladdierer(74 LS 83) gelöst. Die 4 Bit-Information vom Bus und die Information vom 4 Bit-DIL-Schalter werden addiert, zgl. eines Carry-Bits, wobei die Information des DIL Schalters invertiert eingestellt werden muß. (≙ 1 werden geschaltet).
Die vereinfachte boolsche Darstellung der Shipselectauswahl ist:
A-B = A+B+1.
Durch Verknüpfen der vier Sumationsausgänge werden die Adressen 12 - 13 und der Shipselect gewonnen.
Das Timing der Karte, sowohl Refresh als auch Schreib/Lesen wird von dem IC 8202 übernommen. Das Refreshm wird in diesem IC intern ausgelöst. Wird die Karte in einem Refreshzyklus angesprochen, erhöht die CPU einen HALT-Befehl vom 8202. Bei einem Lesebefehl werden die Informationen der RAMs erst in 1 Latch 74 LS 373 übernommen, welcher nur beim Lesen auf den Bus gelegt wird, sonst ist er hochohmig. Die Datenausgänge der RAMs liegen direkt am Bus.

Nach Spezifikation der RAMs sind alle Spannungsversorgungen an den RAMs gesondert abgeblockt.

Aufbauhinweise

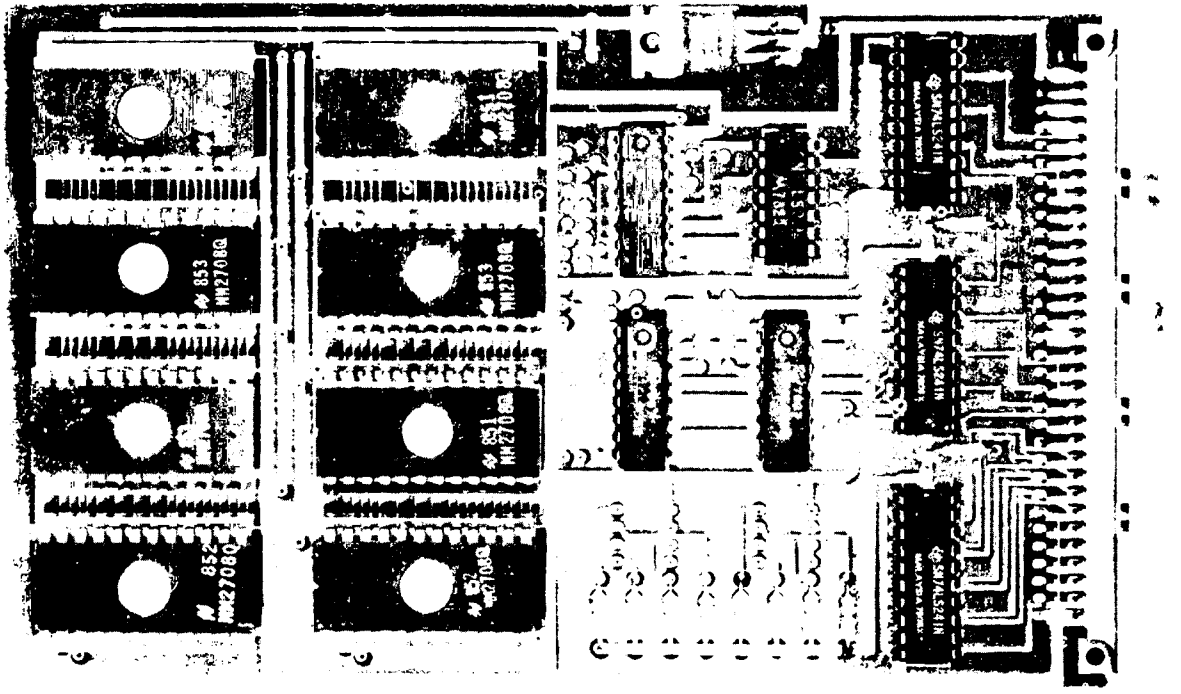
Beachten Sie die allgemeinen Aufbauhinweise.

Testhinweise

Zunächst ist sicherzustellen, daß alle Spannungen keine hochfrequenten Schwingungen aufweisen. Ggf. RC-Glieder nachrüsten. Jetzt kann mit den Testhinweisen für Speicherkarten ein erster Test erfolgen.
Kontrollieren Sie die Funktion der BAIT-Schaltung, d.h. ob beim Ansprechen der Karte die CPU für die Dauer des Refreshzyklusses angehalten wird.

Oft benötigte Programme wie Betriebssysteme, Interpreter, Compiler oder Assembler sollten tunlichst in einem Festspeicher abgelegt werden. Dieser verliert beim Abschalten der Spannung seine Information nicht. Ein solcher Festspeicher ist z.B. ein EPROM. Die gängigen EPROMs unterscheiden sich in der Organisation. Als Standard haben sich durchgesetzt: 512, 1024, 2048 x 8 Bit. Weitere Merkmale sind die benötigten Spannungen und die Zugriffszeiten. Hierunter versteht man die Zeit, die vergehen muß vom Anlegen der Adresse bis zum Auslesen der Daten.

8/16K EPROM - Karte 2708/16



EPROM 2708 Karte

Beschreibung

Die EPROM Karte für den 2708/16 befindet sich auf einer doppelseitigen, durchkontaktierten Europa-Karte und ist mit einer 64 pol. Steckerleiste nach DIN 41612 versehen.

Eingesetzt werden können 8 Stück 2708 oder 8 Stück 2716 (3 Spannungen). Beide IC-Typen benötigen 3 Spannungen (+ 12 V, - 5 V, + 12 V). Die - 5 V werden mit dem auf der Karte befindlichen Stabi 7905 aus den auf dem Bus liegenden - 12 V gewonnen. Die + 12 V müssen (s. Bus-Belegung) auf den Bus gelegt werden.

Die Karte ist bei Lieferung für den 2708 ausgelegt.
Über zwei 4-Bit-DIL Schalter läßt sich für jeweils 4 ICs die Adresse seitenweise einstellen.

Soll die Karte für den 2716 ausgelegt werden, müssen einige Änderungen vorgenommen werden (s. gesonderte Beschreibung). Die Adresse kann dann für 2 x 4 ICs mit 3 Bit eingestellt werden.

Alle auf der Karte benötigten Signale sind gepuffert und alle TTL's in LS-Ausführung.

EPROM Karte 2708

Stückliste

1 Platine
1 Steckerleiste 64 pol.

Integrierte Schaltkreise

IC 1 - IC 3	74 LS 241
IC 4	74 LS 02
IC 5, 7	74 LS 85
IC 6	74 LS 138
IC 8 - IC 15	EPROM Plätze
Stabi 7905	

Widerstände

R 1 - R 12	3K 3 - 4K 7

Kondensatoren

C 1	100 n

Fassungen

8 DIL 24
3 DIL 20
3 DIL 16
1 DIL 14
2 DIL 4 Schalter

Erklärung

Als Bustreiber wird das IC 74 LS 241 eingesetzt. In der 2708 Ausführung der Karte werden die 4 höherwertigen Adressen und die Information eines 4 Bit DIL-Schalters durch einen 4 Bit-Vergleicher verknüpft. Erkennt dieser ein "Gleich", gibt er ein "high" aus, enabelt einen 1 aus 8-Decoder (74 LS 138), welcher, wenn das NRDS-Signal auf dem Bus liegt, das gewählte EPROM freigibt.

Für jeweils 4 EPROMs wird ein 4 Bit-Vergleicher verwendet, welcher, über ein "Oder" verknüpft, den 1 aus 8-Decoder freigibt. Ist die Karte für den 2716 (3 Spannungen) ausgelegt worden, wird die Adresse 10 noch zusätzlich an die EPROMs gelegt und die 4 Bit-Vergleicher erhalten nur noch die 3 höherwertigen Adressen (die vierte liegt auf low, deshalb muß am DIL-4-Schalter Adresse 12 auch auf low geschaltet werden (Stellung on).

Die Umschaltung der Datenpuffer erfolgt, wenn einer der beiden Vergleicher ein "high" ausgibt und das NRDS-Signal kommt.

Änderungen für 2716 (3 Spannungen)

In Bild A ist ein Ausschnitt des Layouts zu sehen. Folgende Änderungen müssen nun vorgenommen werden:

Die Verbindungen: ab, ce, fg, no, pq, rs, tu, vw, xy, z2, 23, 34, 45, 67, 89, ij, !-, - +, üö, müssen unterbrochen werden.

Dann folgende Verbindungen herstellen:

cd, gh, jk, nm, rO, t-, v+, xz, ü4, 85, 72, 69öywusqo.

Die Karte ist jetzt für den 2716 (3 Spannungen) ausgelegt. Diese Änderungen waren erforderlich, da die beiden EPROMs den Chip-Select nicht an gleicher Stelle haben.

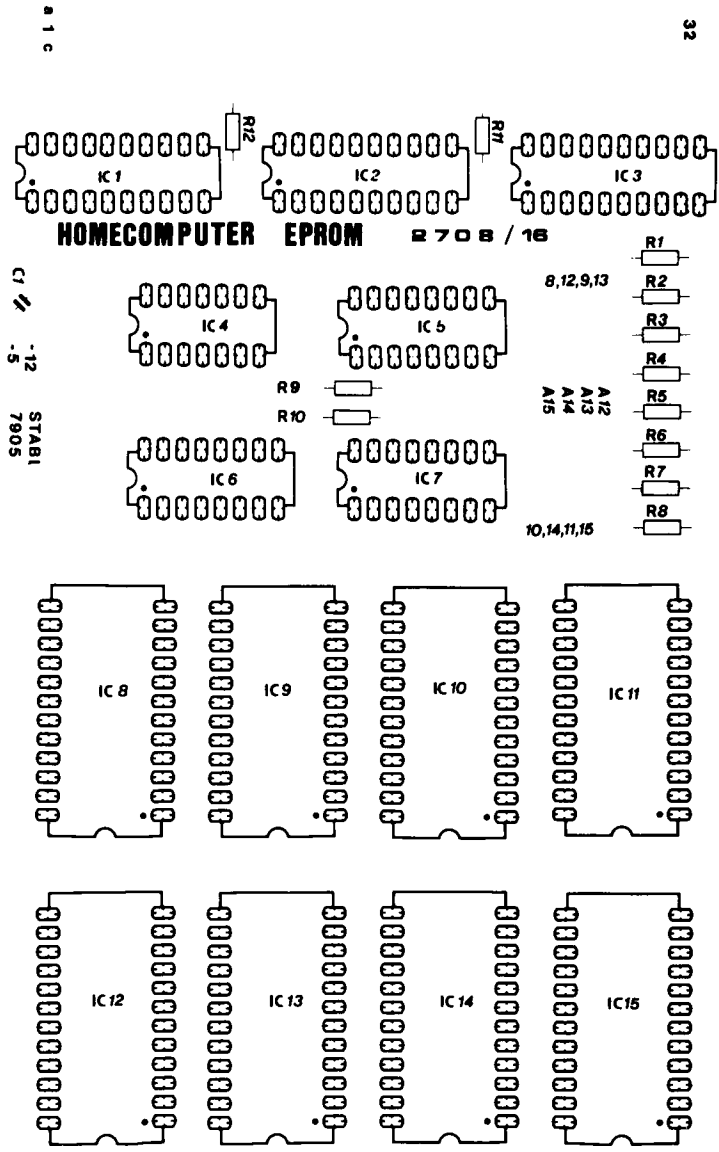
Aufbauhinweise

Den Stabi so einlöten, daß die Metallseite zur Platine zeigt. Spannung + 12 V nach Bus-Belegung verdrahten.

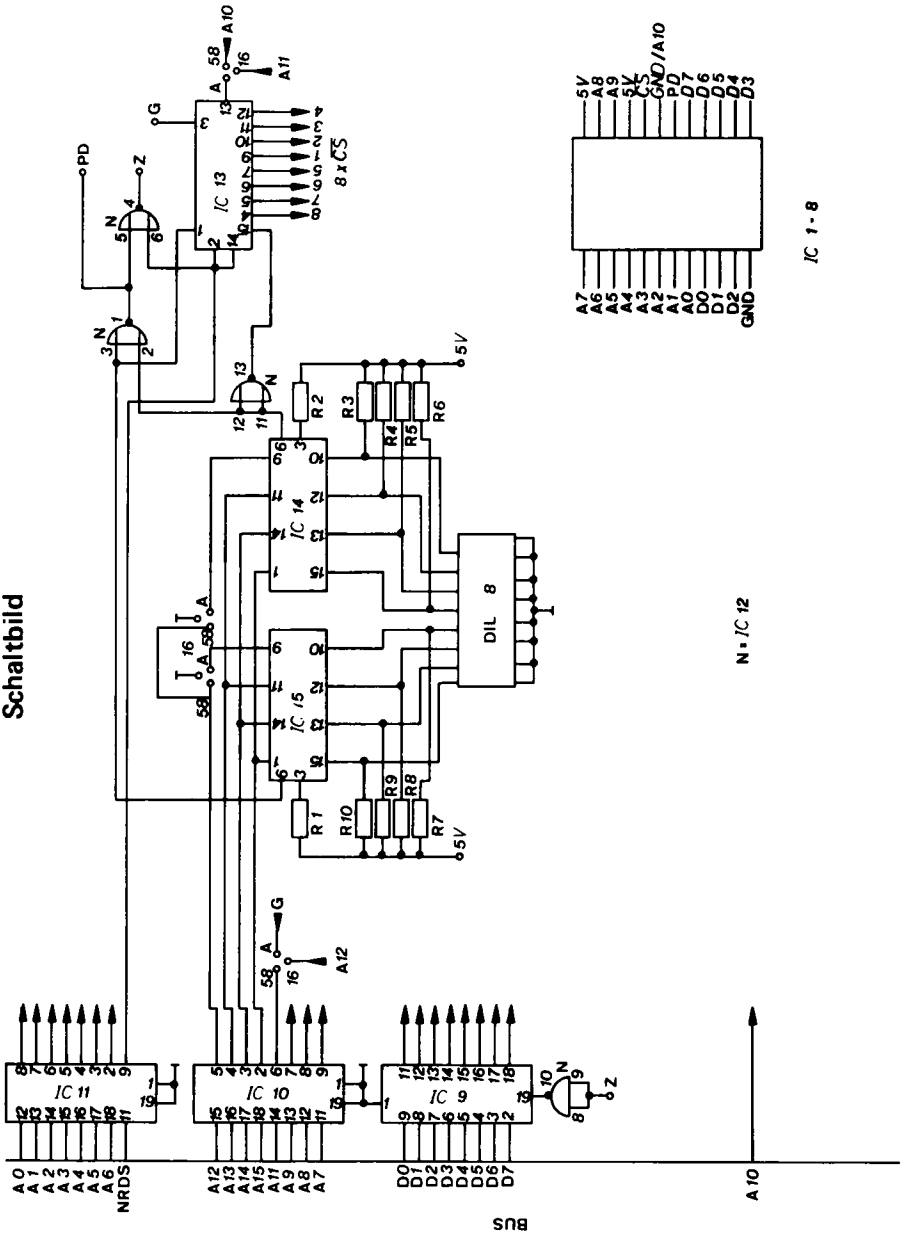
Testhinweise

Beachten Sie die speziellen Testhinweise der EPROM-Karten.

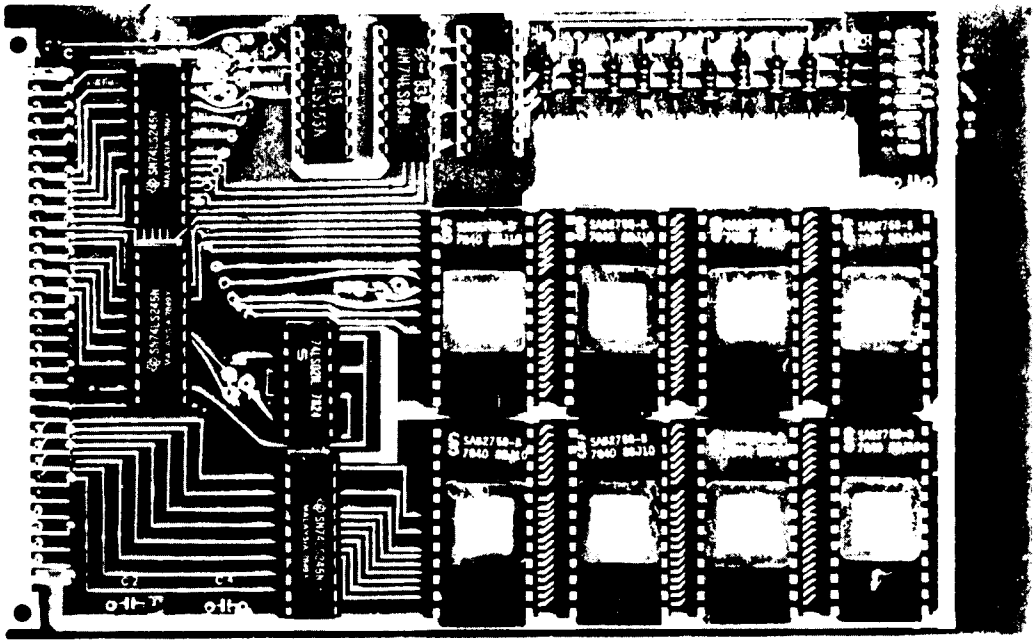
Bestückungsplan



Schaltbild


IC 11
IC 10
IC 9
IC 13
IC 14
IC 15
IC 16
DIL 8
R1
R2
R3
R4
R5
R6
R7
R8
R9
R10
5V
PD
G
Z
A10
A11
A12
8 x CS
N = IC 12
IC 1 - 8
A0 A1 A2 A3 A4 A5 A6 NRDS
A12 A13 A14 A15 A11 A9 A8 A7
D0 D1 D2 D3 D4 D5 D6 D7
A10
BUS
5V A8 A9 5V CS GND/A10 D7 D6 D5 D4 D3
A7 A6 A5 A4 A3 A2 A1 A0 D0 D1 D2 GND

8/16K EPROM-Karte 2758/16



8 K/16 K EPROM Karte für 2758 oder 2716 (5 V Typen)

Die EPROM-Karte ist doppelseitig und durchkontaktiert im Europaformat, mit einer 64 pol. Steckerleiste versehen, passend zum SC/MP System.

Über 4 Lötbrücken wird festgelegt, ob die Karte mit dem EPROM 2758, organisiert zu 1 K x 8 Bit, oder mit dem EPROM 2716, organisiert zu 2 K x 8 Bit, bestückt werden soll.

Gehen die 4 Brücken von den Knotenpunkten A nach 58, ist die Karte für den 2758 ausgelegt. Gehen die 4 Brücken von den Knotenpunkten A nach 16, ist die Karte für 2716 ausgelegt. Somit stehen in der ersten Version auf einer Europakarte 8 K Byte und in der zweiten sogar 16 K Byte zur Verfügung.

Die Karte kann K-weise (2758) bzw. 2 K-weise (2716) bestückt werden. Mit dem 8 poligen DIL-Schalter lassen sich zwei Seiten getrennt einstellen (2758). Bei der 16 K Ausführung kann die Adresse der Karte in zwei seitenweise (für 2 getrennte 8 K-Blöcke) eingestellt werden.

(DIL-Schalter On = 0, Off = 1), Adressen-Belegung des DIL-Schalters siehe Bestückungsplan Abdruck bzw. Aufdruck.
Außerdem wird die Karte, wenn sie nicht angesprochen wird, im "Power Down" betrieben. Das bedeutet, daß sie dann nur ca. 1/5 des aktiven Stromes benötigt.

Die Zugriffszeit ist bei beiden EPROM's 450 ns.

Natürlich sind wie immer bei unseren Platinen alle auf der Karte benötigten Signale gepuffert und alle IC's in LS-Ausführung.

Stückliste

8K/16K EPROM Karte 2716 bzw. 2758

1 Platine
1 Steckerleiste 64 Pol

Integrierte Schaltkreise

IC 1 - IC 8 Plätze für die EPROMS	
IC 9, 10 und 11	74 LS 245
IC 12	74 LS 02
IC 13	74 LS 155
IC 14 und 15	74 LS 85

Widerstände

R 1 - 10	3 K 3 - 4 K 7

Tantal-Kondensatoren

C 1 - C 4	22 my/16 V

Fassungen

8 DIL 24
3 DIL 20
3 DIL 16
1 DIL 14
1 8 Bit Schalter

Erklärung 2758/2716 EPROM-Karte

Als Bustreiber wird wieder das IC 74 LS 245 eingesetzt. In der 2758 Ausführung der Karte werden die 4 höherwertigen Adressen und die Information eines 4 Bit Schalters durch einen 4 Bit Vergleicher verknüpft. Erkennt dieser ein "Gleich" (zwischen der Information des 4 Bit Schalters und der Adresse vom Bus) gibt er ein high aus, welches zum Freigeben eines Demultiplexers (74 LS 155) benutzt wird. Der Demultiplexer wählt unter Zuhilfenahme der Adressen 10, 11 und des NRDS Signals den entsprechenden EPROM Platz aus.

Es befinden sich 2 Vierbit-Vergleicher und ein 8 Bit Schalter auf der Karte, so daß zwei Seiten getrennt eingestellt werden können. Wird die Karte für den 2716 (5 V Typ) ausgelegt, wird zusätzlich eine Adresse (A10) an die EPROM Plätze gelegt und die Adressen des 4 Bit Vergleichers und des Demultiplexers werden entsprechend angepasst.

Adresse 12 der beiden 4 Bit Schalter (8 Bit-Schalter) müssen auf "On" geschaltet werden, da jetzt dieser Eingang des 4 Bit Vergleichers auf Masse liegt.

Der Demultiplexer (74 LS 155) wählt nun das EPROM unter Zuhilfenahme der Adressen 11 und 12 aus.

Die Umschaltung des Datenbuffers erfolgt, wenn die Karte angesprochen (einer der 4 Bit Vergleicher erkennt "Gleich") und das NRDS Signal anliegt.

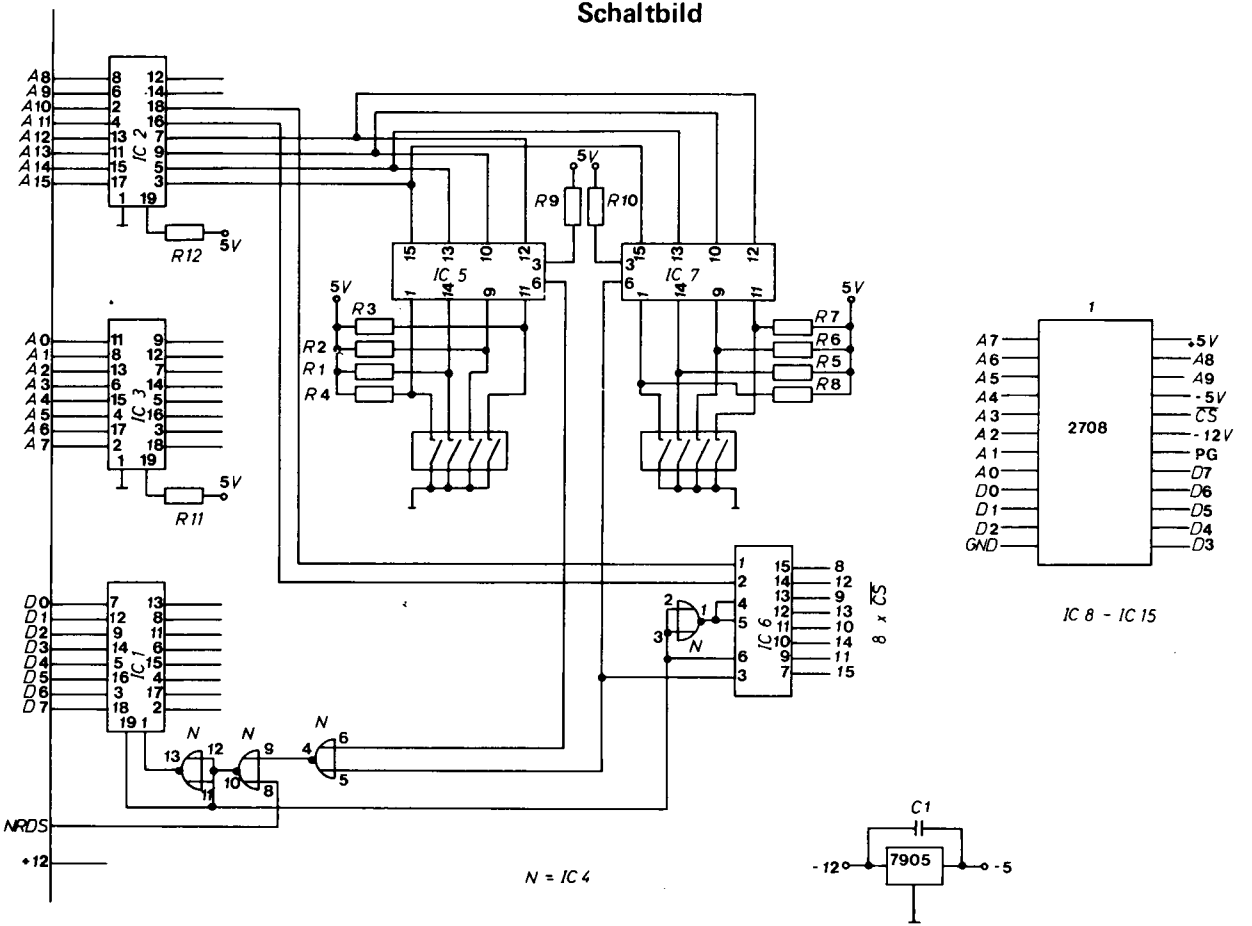
Aufbauhinweise

IC 10 und 11 werden entgegengesezt zu den anderen IC's eingesteckt. Brücken für die Wahl der EPROM nicht vergessen.

Testhinweise

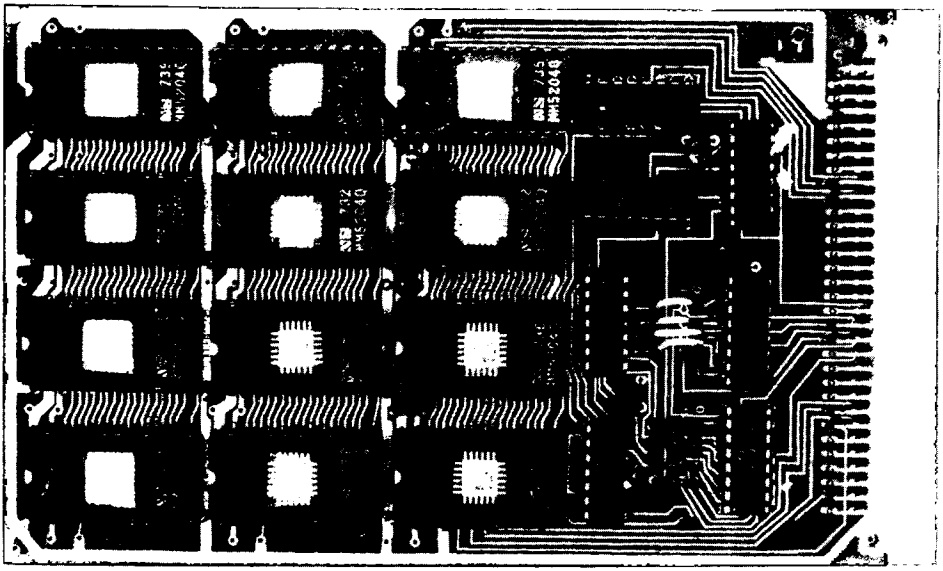
Beachten Sie die Testhinweise für Speicherkarten.

Schaltbild


IC 2
IC 3
IC 1
IC 5
IC 7
IC 6
R12
R11
R9
R10
R3
R2
R1
R4
R7
R6
R5
R8
5V
NRDS
+12
N
N = IC 4
8 x CS
2708
IC 8 - IC 15
C1
7905
-12
-5

Bestückungsplan

6K EPROM-Karte für 5204



6 K EPROM Karte für 5204

Die EPROM-Karte ist eine doppelseitige und durchkontaktierte Karte im Europaformat. Sie ist versehen mit einer 64 Pol Steckerleiste und paßt auf den SC/MP Bus.

Sie wird bestückt mit dem EPROM 5204 (5244), welches 512 x 8 Bit organisiert ist; somit kann die Karte 1/2 K-weise bestückt werden.

Die Seitenauswahl erfolgt mittels Brücken zwischen den Punkten 11 - 15 einerseits und den Punkten A oder B andererseits. Die Adressen 12 - 15 dienen der Adressierung einer ganzen Seite, die Adressen 11 - 15 der Adressierung einer halben Seite.

Z.B.: 12, 13, 14, 15 verbunden mit Punkt A = Seite 0.
Z.B.: 11 mit Punkt B und alle anderen mit Punkt A = 0800.

Alle TTL sind low power Ausführung.

Stückliste

6 K EPROM Karte für 5204

1 Platine
1 Steckerleiste 64 Pol

Integrierte Schaltkreise

IC 1 - IC 12 sind die Plätze für die EPROMs

IC 13	CD 4012
IC 14	CD 4050
IC 15 und 16	CD 4049
IC 17 und 18	74 LS 155
IC 19	74 LS 02

Tantalkondensatoren

Ca 1	22 my 16 V

Fassungen
12 DIL 24
5 DIL 16
2 DIL 14

Erklärung:

Als Adressenbustreiber dienen hier invertierende und nicht invertierende C-MOS Gatter (4049 + 4050).

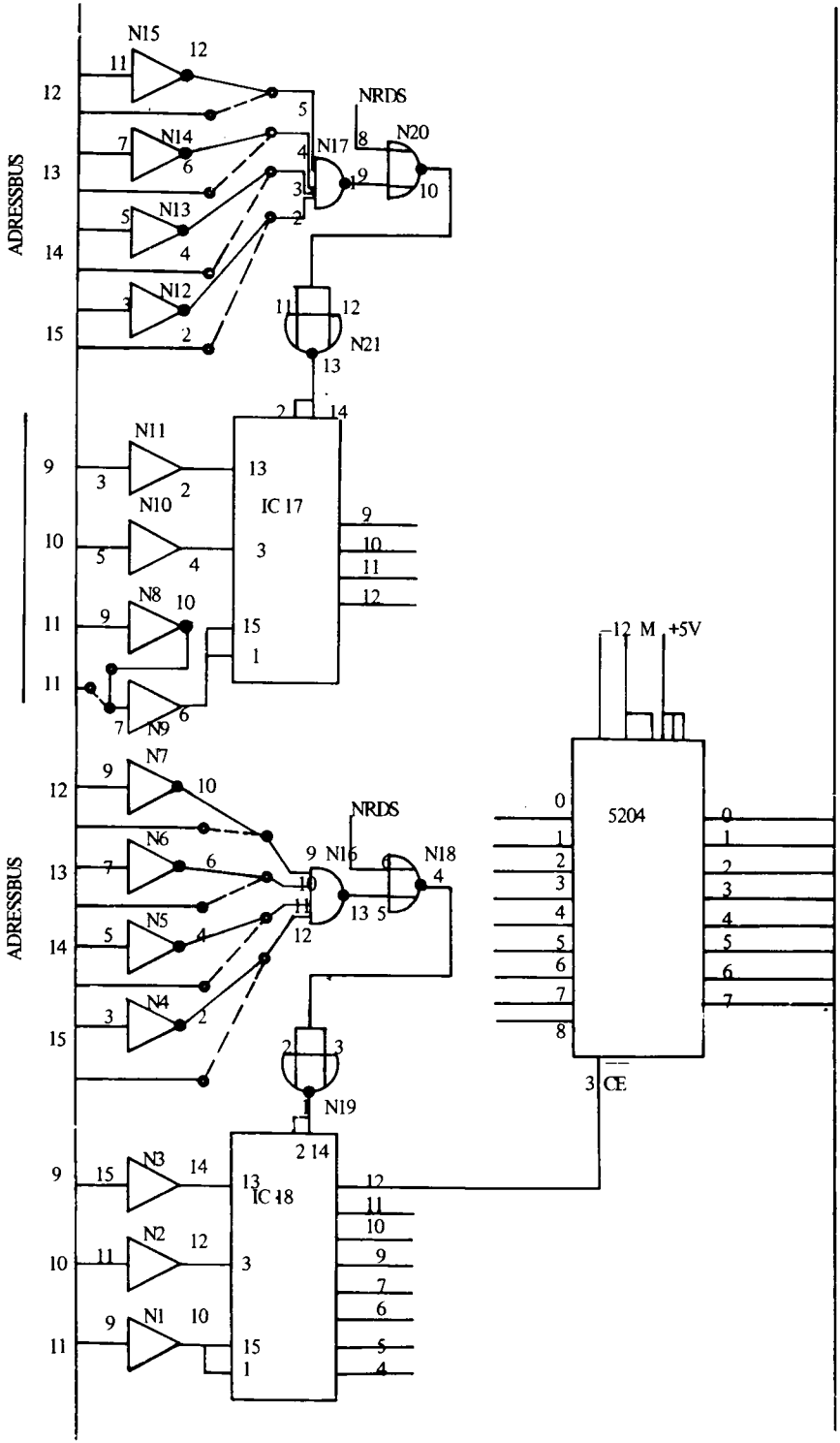
Die Adreßdekodierung übernimmt ein Demultiplexer (74 LS 155). Ein und eine halbe Seite können getrennt eingestellt werden. Somit sind zwei Demultiplexer vorhanden, welche getrennt druch die höherwertigen Adressen und das NRDS Signal freigegeben werden.

Aufbauhinweise

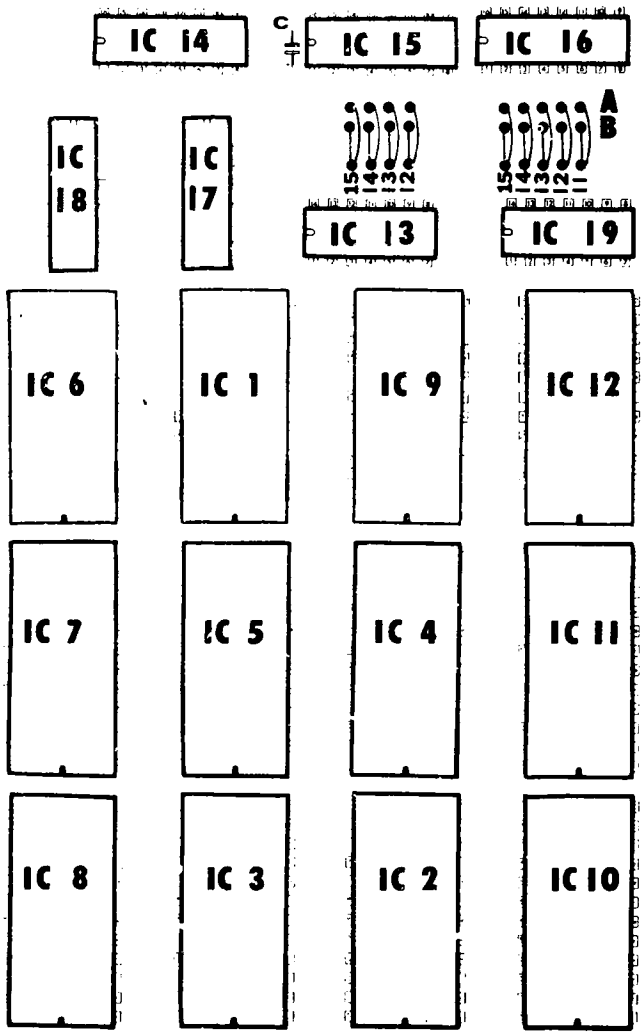
Beachten Sie bitte die allgemeinen Aufbauhinweise.
IC 17 - 18 Pin 1 nach oben zur Steckerleiste weisend,
IC 1 - 12 (EPROM's) Pin 1 nach unten weisend.

Testhinweise

Zum Test beachten Sie bitte die allgemeinen Hinweise für Speicherkarten.

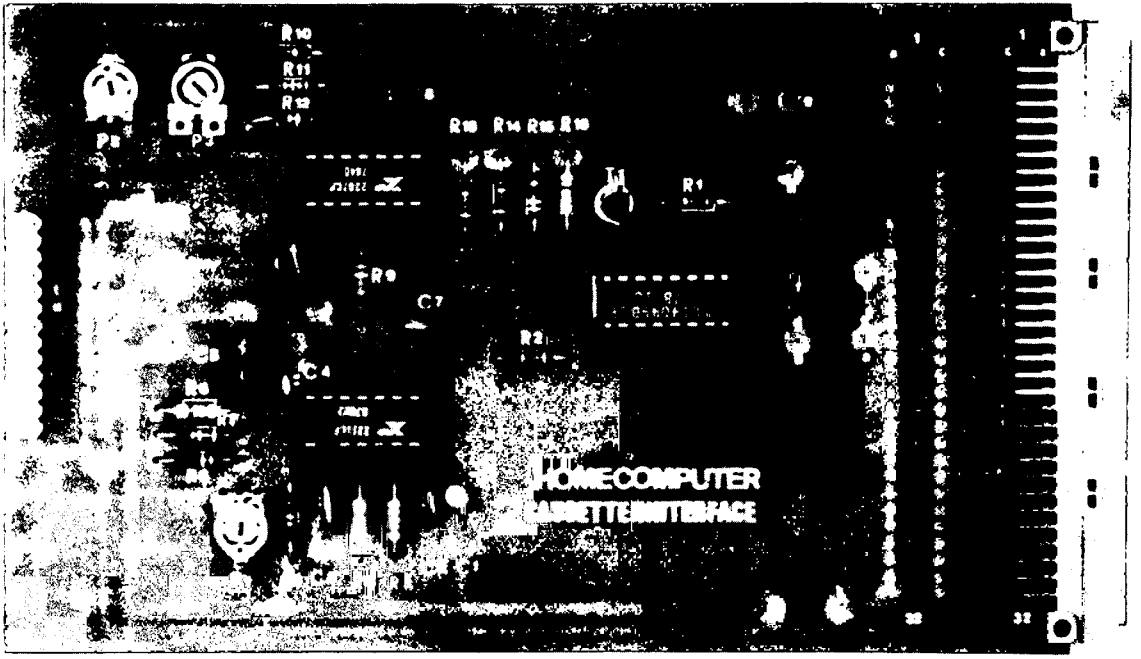

ADRESSBUS
N15
N14
N13
N12
NRDS
N17
N20
N21
IC 17
N11
N10
N8
N9
ADRESSBUS
N7
N6
N5
N4
NRDS
N16
N18
N19
IC 18
N3
N2
N1
DATENBUS
-12 M +5V
5204
CE

HOMECOMPUTER 6K-EPROM-5204



Für Programme, welche nicht im EPROM abgelegt werden sollen, bedient man sich eines Massenspeicners, dessen preiswerteste Version der Cassettenrecorder ist. Dieser wird über ein Cassetten-Interface (Interface = Bindegleid) mit dem Computer verbunden. Das Interface übersetzt die Signale des Computers (0 + 1) in Signale für den Cassetten-Recorder (2 verschiedene Töne). Dieses gebräuchlichste Verfahren nennt man FKS.

Cassetteninterface 2400 Baud



Beschreibung

Das HC Cassetteninterface befindet sich auf einer Europakarte, ist mit einer 64 pol. Steckerleiste und einem 26 pol. Wire-Wrap Stützpunkt versehen.

Das Interface kann an jedes 64 pol. Bussystem (a - c) angeschlossen werden. Außerdem sind alle benötigten Signale und Spannungen auf einen 26 pol. Stecker herausgeführt. Datenein- und Ausgang sind TTL kompatibel.

Das Interface arbeitet nach dem FSK-Verfahren. Baudraten sind bis zu 2.400 Bd möglich.

Der NF-Eingang hat eine Empfindlichkeit von 50mV. Eingangsspannungen bis über 10 V_{ss} werden einwandfrei verarbeitet.

Erklärung

Der FSK Modulator arbeitet mit dem IC XR 2207. Dieses IC, mit Beschaltung, erzeugt die benötigten FSK-Frequenzen, in diesem Fall 3200 Hz und 4800 Hz Rechteck. Diese beiden Frequenzen werden mit Potis eingestellt. Der TTL-Eingang wird über einen Inverter (4049) auf einen Pegelumsetzer (+ 12 V) geführt, da das IC nicht TTL kompatibel ist.

Der FSK-Demodulator arbeitet mit dem IC XR 2211. Der NF Eingang wird direkt über einen Entkopplungskondensator auf den Eingang des IC's geführt. Über einen Inverter (4049) gelangt das Ausgangssignal (TTL kompatibel) zum 64 pol. Bus oder/und zum 26 pol. Stecker.

Aufbauhinweise

Nachdem die aktiven und passiven Bauelemente eingelötet worden sind, wird über Drahtbrücken die Bus-bzw. Steckerbelegung wie folgt festgelegt:

Brückenbelegung für HC SC/MP-System
Verbindungen

a - b
c - d
e - f
g - h
k - l

Brückenbelegung für HC Z 80 System
Verbindungen

k - l
g - h
a - b
s - t
q - u

Falls das Cassetteninterface beim HC SC/MP System mit der HC ASCII, USART' TIMER-Karte betrieben werden soll, ist die gleiche Verdrahtung wie beim HC Z 80 System zu verwenden.

Belegung für andere Systeme

Alle Stecker, sowohl 64 po. als auch 26 pol. Stecker, sind doppelt ausgeführt; somit ist es möglich, durch Brücken alle Belegungen anzupassen.

Definition für alle benötigten Signale

b, n	+ 12V Versorgungsspannung
d, s	Data-out
f, q	Data-in
h, p	+ 5V Versorgungsspannung
l, w	Masse

Diese Punkte müssen nach Ihren Erfordernissen an die doppelt ausgeführten Stecker verdrahtet werden.

Der Cassettenrecorder wird mit seinem Eingang an Punkt z, seinen NF Ausgang an Punkt x und seiner Masse an Punkt y angeschlossen.

Testhinweise

Modulator

Versorgungsspannung anschließen und den Punkt f oder q auf Masse legen. Legen Sie einen Frequenzmesser an Ausgang Punkt z (NF out). Mit dem Poti P2 wird die Frequenz auf 3.200 Hz eingestellt. Anschliessend Punkt f, q auf TTL log. 1 und mit P3 4.800 Hz einstellen. Einstellung nur in dieser Reihenfolge vornehmen. Der Modulator ist nun abgeglichen.

FSK-Demodulator

Mit offenem NF Eingang die Anschlüsse 2 und 10 des XR 2211 kurzschließen. Schließen Sie an Anschluß 3 des gleichen IC's einen Frequenzmesser an und stellen mit Poti 1 eine Frequenz von 4 KHz ein.

Das Cassetteninterface ist jetzt abgeglichen und betriebsbereit.

Stückliste

1 Platine
1 Steckerleiste 64 pol.
1 Wrap Stützpunkt 26 pol.

Integrierte Schaltkreise

IC 1	4049
IC 2	XR 2211
IC 3	XR 2207

Halbleiter

T 1	TUN

Widerstände

R 1, 2, 13, 14	4,7K
R 3	8,2K
R 4	33K
R 5	15K

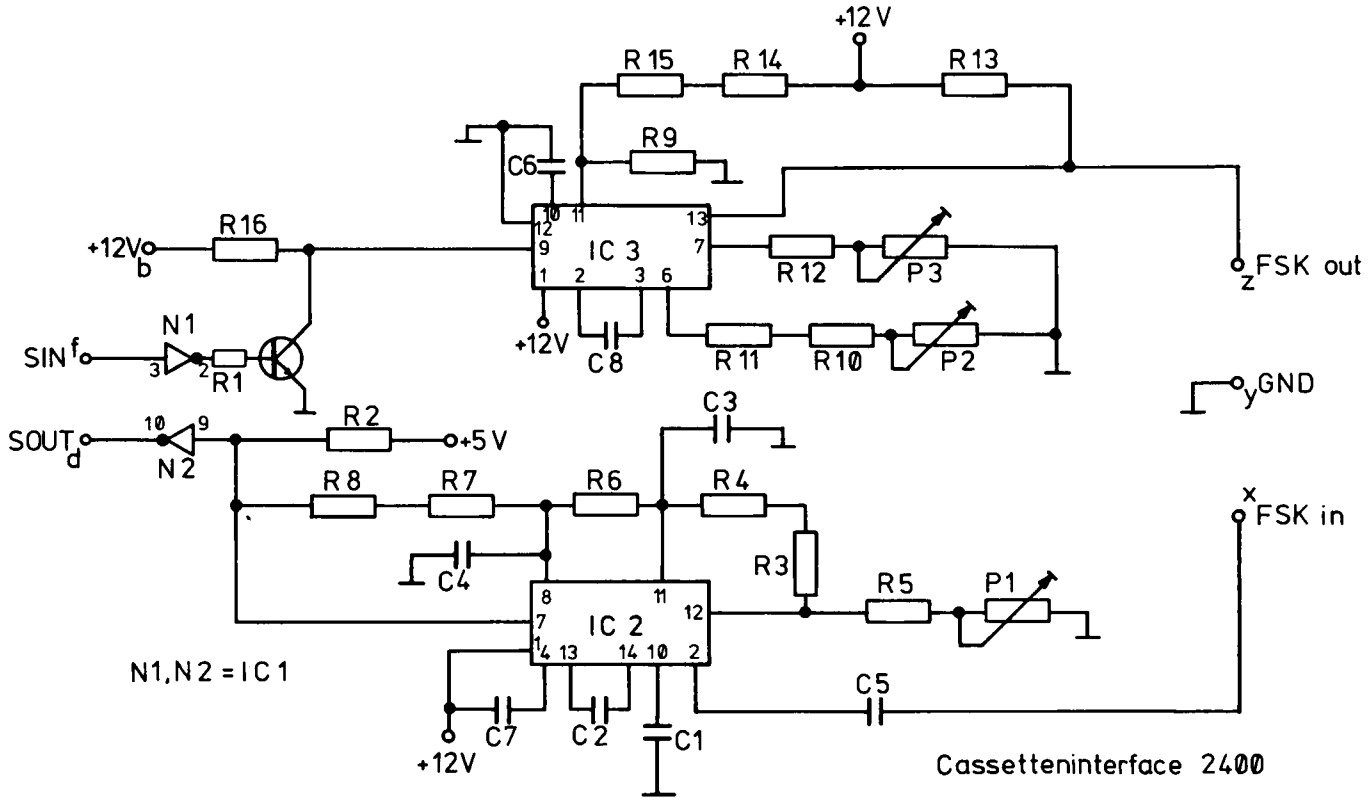
R 6	100K
R 7	470K
R 8	47K
R 9	3,9K
R 10	6,8K
R 11	22K
R 12	56K
R 15	390
R 16	10K
P 1, 2	4,7K
P 3	10K

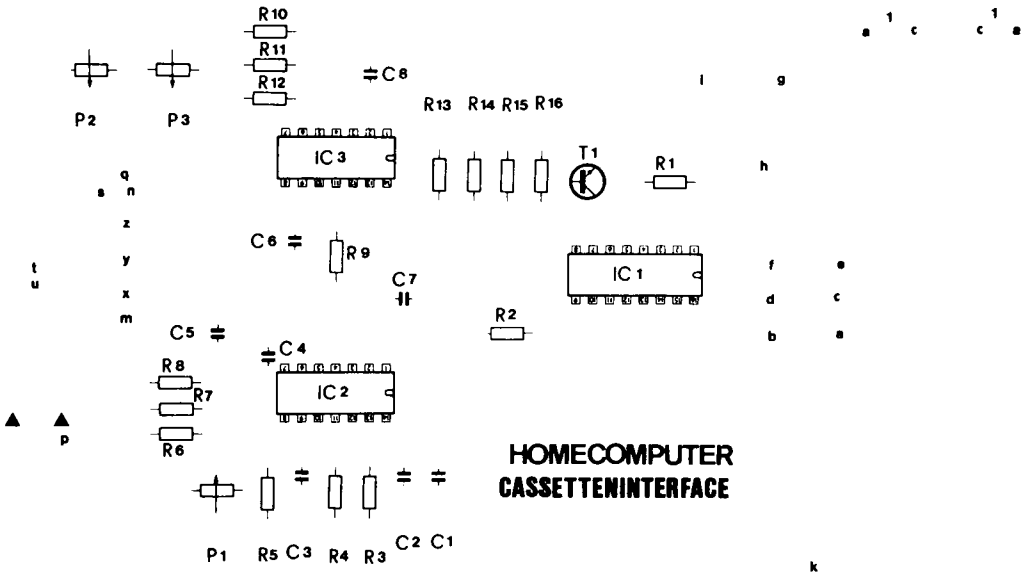
Kondensatoren

C 1	1my
C 2	15n
C 3	3,9n
C 4	1n
C 5, 6, 7	100n
C 8	10 n

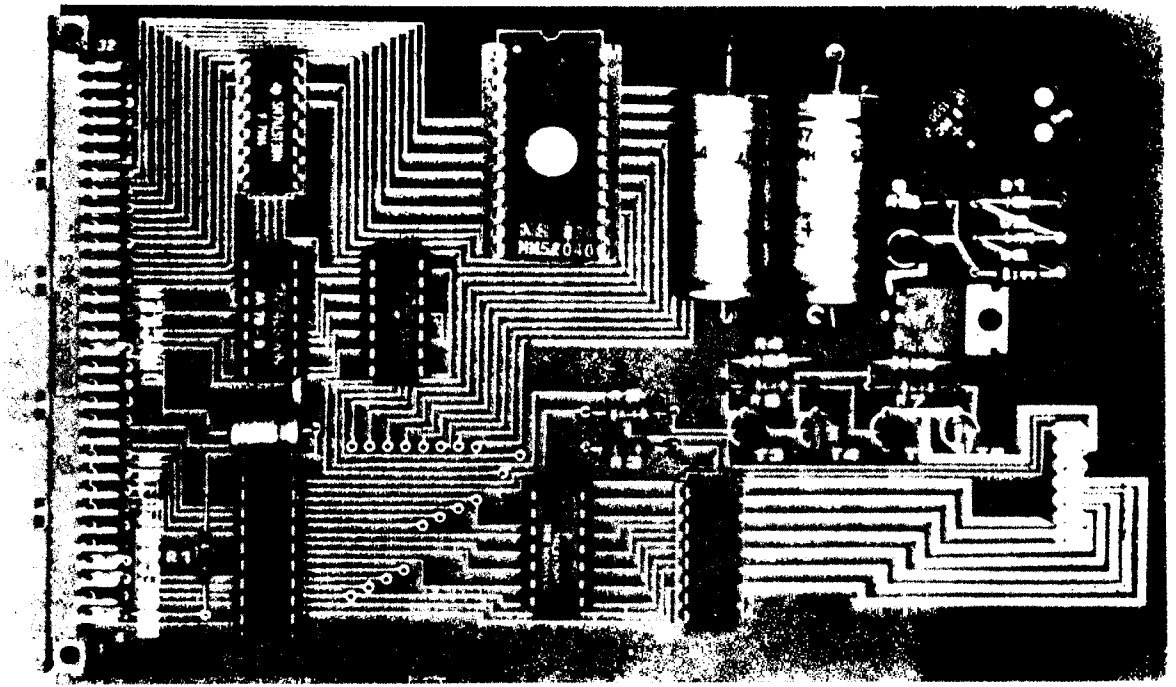
Fassungen

1	DIL 16
2	DIL 14


+12V
R15
R14
R13
R9
C6
R16
+12V
b
IC 3
R12
P3
z FSK out
SIN f
N1
3
2
R1
+12V
C8
R11
R10
P2
y GND
SOUT d
10
9
N2
R2
+5V
C3
R8
R7
R6
R4
x FSK in
C4
R3
R5
P1
8
11
7
12
IC 2
1
4
13
14
10
2
N1, N2 = IC 1
C5
+12V
C7
C2
C1
Cassetteninterface 2400
Hoco 7.'79 BNOE


R10
R11
R12
C8
P2
P3
R13 R14 R15 R16
IC 3
T1
R1
C6
R9
C7
IC 1
R2
C5
C4
R8
R7
IC 2
R6
HOMECOMPUTER
CASSETTENINTERFACE
P1
R5 C3
R4
R3
C2 C1
32
32

Metallpapierdrucker - Interface



Der Drucker ermölglicht es dem Anwender, Computerdaten auszulesen und zu fixieren. Eine preiswerte Version ist der Kleindrucker. Auch dieser wird über ein Interface mit dem Mikroprozessor verbunden.

Beschreibung

Das Kleindrucksystem verwendet das moderne, elektrosensitive Aufzeichnungsverfahren. Dieses Verfahren bietet große Betriebssicherheit bei minimalem Wartungsaufwand. Es beruht auf dem Prinzip des selbstheilenden Metallpapier-Kondensators. Das Metallpapier besteht aus einer dünnen Metallschicht mit einer Stärke von ca. 0,1 μm, die auf ein Papierband mit Kontrastschicht aufgedampft wird. Im Drucker läuft das Band mit seiner Metallseite über eine großflächige zylindrische Elektrode. Der Druckknopf mit seinen sieben Gegenelektroden gleitet mit einem geringen Druck über die Mettalfläche. Wird an die Elektroden des Druckkopfes eine Spannung von mindestens 24 Volt gelegt, so fließt durch die Metallschicht ein Strom und das Metall in der nächsten Umgebung der Elektroden verdampft infolge der hohen Stromdichte und die Kontrastschicht wird sichtbar. Jeder einzelne Ausbrand dauert ca. 1 μ Sekunde bei Stromspitzen von einigen hundert Milliampere.

Einige Vorzüge dieses Aufzeichnungsverfahrens sind:

- Besondere Geräuscharmut
- Hohe Druckgeschwindigkeit
- Geringe Abhängigkeit von Temperatur, Feuchte und Erschütterungen
- Kein Nachfüllen von Tinte oder Auswechseln von Farbbändern

Das sehr kompakt aufgebaute Druckwerk beinhaltet bis auf die Papierrollenhalterung alle für die Funktion erforderlichen mechanischen Teile. Es kann völlig getrennt von der notwendigen Elektronik angeordnet werden.

Beschreibung des Druckwerkes

Im Druckwerk (Bild 1) befindet sich ein Gleichstrommotor, der über ein Getriebe eine Kurvenscheibe antreibt, die einen Drehwinkel von

296° ausführt. Diese Bewegung wird über ein Seil auf den Druckkopf übertragen. Am Ende jeder Zeile erfolgt automatisch ein Schreibkopfrücklauf sowie ein Papiervorschub um eine Zeile. Während des Druckkopfrücklaufes heben die Schreibstifte vom Metallpapier ab. Elektrisch ist der Druckkopf mit seinen sieben Schreibstiften über eine flexible, gedruckte Schaltung mit dem Druckwerk verbunden.

Gleichfalls vom Motor wird eine Zahnscheibe angetrieben, die in einer Spule eine sinusförmige Spannung induziert, die der Synchronisation zwischen Druckkopf und Zeichengenerator dient. Durch diese Maßnahme haben Änderungen der Motordrehzahl auf das Druckbild keinen Einfluß.

Ein eingebauter Reedkontakt meldet an die Elektronik die Freigabe des Ausdruckes bzw. die Abschaltung des Motors nach jeder Zeile.

Deshalb muß die abzudruckende Information zeilenweise aus einem Speicher zur Verfügung stehen.

Die darzustellenden Zeichen werden aus einer 7 x 5 Punktmatrix zusammengesetzt.

Betriebshinweise für das Druckwerk

- Nur 24 Volt-Metallpapier verwenden. Bei anderen Papiersorten kommt es zu einem unklaren Ausdruck bzw. die Lebensdauer des Druckkopfes kann nicht garantiert werden.

- Über den rechteckförmigen Ausschnitt im Bodenblech des Druckwerkes soll der anfallende Abbrand des Metallpapiers abgeführt werden. Es ist darauf zu achten, daß dieser Ausschnitt nicht abgedeckt wird.

- Sollte das Metallpapier schräg einlaufen oder sich im Druckwerk verklemmen, so ist mit Hilfe des Papierlöseknopfes die exakte Lage des Streifens herzustellen. Den Papierlöseknopf nur bei Stillstand des Druckkopfes betätigen.

- Die Papierabrißschiene läßt sich wie folgt entfernen:

Bewegen in Richtung 1 und abziehen in Richtung 2.

Wieder einsetzen läßt sich die Abrißschiene, indem die Blechzunge C mit Hilfe der Schiene in Pfeilrichtung gedrückt wird und die Teile A und B wieder in die Ausschnitte der Blechhaube hineinpassen.

Technische Daten des Druckwerkes
A und B wieder in die Ausschnitte der Blechhaube hineinfassen.

Technische Daten des Druckwerkes

Druckprinzip	Elektrosensitiv, 7x5 Punktmatrix, Wahlweise	
Schreibdichte	8, 16 oder 32 Zeichen je Zeile	
Zeichendarstellung	Alphanumerisch und Symbole	
Druckgeschwindigkeit	530 m Sek + 170 m Sek./Zeile. Das entspricht ca. 64 Zeichen/Sekunde bei 32 Zeichen/Zeile	
Zeichenhöhe	2,4 mm + 0,2 mm	
Punktdurchmesser	0,3 mm	
Zeilenabstand	0,4 mm + 0,6 mm	
Zeichenhöhe	2,4 mm + 0,2 mm	
Punktdurchmesser	0,3 mm	
Zeilenabstand	0,4 mm + 0,6 mm	
Motorspannung	24 Volt DC + 5 %	
Motorstrom	Mittelwert:	85 mA
	Anlauf:	350 mA
	Bremsung:	150 mA

Der Drucker-Interface befindet sich auf einer doppelseitig beschichteten, durchkontaktierten Karte im Europaformat und ist mit einer 64 pol. Steckerleiste nach DIN 41612 versehen.

Am vorderen Ende der Platine befindet sich ein 14 pol. Wire-Wrap-Stützpunkt, über den unter Zuhilfenahme eines 14 pol. Litzenleiterkabels das Druckwerk angeschlossen wird (siehe gesonderte Beschreibung). Das Druckwerk benötigt eine Spannung von ca. 27V, diese müssen von außen angelegt werden (∿ 25 V). Die Stabilisierung befindet sich auf der Karte. Als "Nadeltreiber" wird das neue Transistorarray XR2203 eingesetzt. In ihm befinden sich 7 NPN Darlington-

Transistoren, welche alleine die Ansteuerung der Druckernadeln übernehmen.

Ebenso befindet sich die Betriebssoftwar für das Drucksystem in einem 5204 EPROM auf dieser Karte (Beschreibung hierzu siehe gesonderte Ausführungen).

Dargestellt werden können Buchstaben, Zahlen und Sonderzeichen im ASCII-Code (6 Bit). Durch die beiden anderen Bits wird die Schriftbreite durch die Software gesteuert (siehe Software-Beschreibung).

Alle TTL's sind in LS-Ausführung.

Erklärung

Das Drucker-Interface belegt die Adresse F000 - F5FF wie folgt:

F000 - F1FF	Drucker-Betriebssoftware
F200 - F3FF	Nadellatch
F400 - F5FF	Takt- und Reedabfrage

Der Druckvorgang

Bevor dieser ausgelöst wird, muß der Inhalt der zu druckenden Zeile komplett zur Verfügung stehen. Der Motor wird gestartet (durch Einschreiben einer 80 in den Nadellatch). Die Motorgegentaktschaltung schaltet den Motor an. Während der Motor läuft, wird duch ein im Drucker mitlaufendes Zahnrad eine sinusförmige Wechselspannung in eine Spule induziert (diese wird für die Steuerung des Druckbildes herangezogen). Über einen Schmitt-Trigger (74 LS 14) wird die sinusförmige Spannung in ein Rechtecksignal gewandelt und an einen Leitungstreiber (74 LS 125) gelegt (hier kann er über einen getrennt steuerbaren Chipselekt vom SC/MP gelesen werden). Ebenso kann hier über einen internen Reedkontakt festgestellt werden, ob der Druckkopf am Anfang einer Druckzeile steht. Nachdem die Zeile gedruckt wurde, wird automatisch ein Zeilenvorschub ausgeführt und die Software steuert den Druckkopf so, daß er immer an der gleichen Stelle (mit abgehobenen Nadeln) zum Stehen kommt. Damit der Motor sofort stehen

bleibt, wird dieser durch die Gegentaktschaltung kurzgeschlossen, sonst würde er langsam auslaufen.

Das Latchen der Dateninformation übernimmt hier ein 8 Bit D-Latch (74 LS 237). Durch einen Power-on-Reset wird er beim Einschalten gelöscht. Dies kann auch durch die Reset-Taste auf der HEX I/O geschehen.
Alle 16 Adreß-Bits werden unter Zuhilfenahme eines 1 aus 8 Decoders (74 LS 138) und eines NAND-Gatters (74 LS 00) verarbeitet. Der 1 aus 3 Decoder erzeugt auch die nötigen Chip-Select. Das Freigeben des Latches erfolgt, wenn die Adresse des Latches angesprochen wird und das NWDS-Signal auf dem Bus liegt.

Achtung!!!

Da in dem Transistor-Array NPN-Transistoren enthalten sind, liegen am Druckergehäuse + 25 V.

Kabel-Anschluß (siehe Schaltbild)

Ab Anschluß "Spule" fortlaufen das mitgelierferte Kabel ab Punkt A an der 15 pol. Buchse anlöten.

Hinweise zum Gebrauch der Druckerroutine

Wie schon erwähnt, können nur ganze Zeilen auf einmal gedruckt werden.

Der Text für eine Zeile wird zunächst als ASCII-Zeichen in einen dafür vorgesehenen RAM-Bereich geladen. Dieser Bereich ist innerhalb des 64K Speichers frei wählbar, sofern die Routine an dem Label PRINT angesprungen wird. Beim Anspringen des Labels PRINT wird automatisch die Adresse 0F00 als TEXTBASE definiert, d.h. die ASCII-Zeichen werden von 0F00 in das RAM geschrieben bzw. von hier geholt und ausgedruckt.

Bei frei gewählter TEXTBASE muß deren Adresse in TEXTAD (Higher Byte = 0 FF4, lower Byte = 0 FF3) hinterlegt werden. Die Druckerroutine "sieht dort nach", von wo sie die Zeichen zu holen hat.

Es ist zu beachten, daß 5 RAM-Byte unterhalb von TEXTBASE von der Drucker-Routine als Zwischenspeicher für diverse Zähler benutzt werden. Konkret: An der betreffenden Adresse muß sich tatsächlich RAM befinden und ein eventueller vorheriger Inhalt dieser 5 Byte wird zerstört.

Bei einem Vorrat von 64 Zeichen werden nur die niederen 6 Bit des ASCII-Codes benötigt. Z.B.:

A = ASCII-Code 41
= 6-Bit-ASCII 01

Bit 6 und 7 sind im vorliegenden System zum Definieren der Zeichen-Breite herangezogen worden.

Bit 6	7		
0	0	schmale Zeichen	(32/Zeile)
0	1	mittlere Zeichen	(16/Zeile)
1	0	breite Zeichen	(8/Zeile)

Z.B.:0000 0001 = schmales A (X'01)
0100 0001 = mittleres A (X'41)
1000 0001 = breites A (X'81)

Wenn eine Speicher-Zeile im Text-RAM mit X' 41 geladen ist, wird ihr Inhalt als mittelbreites "A" erkannt und ausgedruckt.

Das Ende des Textes, der nicht immer eine ganze Zeile ausfüllen muß, wird mit X'FF in der nächsthöheren Adresse (nach dem letzten Zeichen im Text-RAM) markiert.
Nachdem im Anwender-Programm (oder im Anfangsstadium mit "Modify") die TEXTBASE definiert und die Zeichen für eine Zeile geladen sind, erfolgt der Ansprung der Druckerroutine. Diese kann nur über eine ELBUG-Software-LIFO-Stack geschehen. Der Vorzug dieses Ansprungs liegt darin, daß nach dem Rücksprung in das Anwender-Programm alle internen Register des SC/MP wieder ihren vorherigen Inhalt haben.

Man geht folgendermaßen vor:

– Die Startadresse der Druckerroutine - 1 (PRINT - 1) oder PRINT-1) in ROUTAD (Higher Byte = 0FFC, lower Byte = 0FFD) laden und anschließend
– JS 3 (PUSH)

Siehe hierzu auch das "Hex-Matrix-Programm".

Das Drucker-Programm holt nun die ASCII-Zeichen für eine Zeile, druckt sie aus und springt zurück ins Anwenderprogramm.

Für einen weiteren Ansprung der Druckerroutine braucht die TEXT-BASE nicht erneut definiert zu werden; der Inhalt von TEXTAD hat sich inzwischen nicht geändert, kann aber geändert (neu definiert) werden, wenn die Zeichen für die nächste Zeile aus einem anderen RAM-Bereich geholt werden sollen. Ebenso kann die Druckerroutine zum wiederholten Ansprung in einem einfachen XPPC 3 erreicht werden, sofern der Inhalt des Pointer-Registers 3 inzwischen im Anwender-Programm nicht geändert wurde.

Vor einem erneuten Ansprung muß aber das Text-RAM (mit dem neuen Text) geladen sein.

Software

Mitgeliefert wird eine Cassette, auf der sich eine HEX-Matrix befindet. Mit ihm ist es möglich, Programme in hexadezimaler Schreibweise auszudrucken.

Nach Starten des Programms bei 0E50 erscheint auf dem Display PR ..., nun kann die Start- und Endadresse des auszudruckenden Blocks angegeben werden. Wird vorher die T-Taste gedrückt, kann noch ein Text von max. 8 Buchstaben als Überschrift angegeben werden.

Aufbauhinweise

Beachten Sie die allgemeinen Aufbauhinweise.

Testhinweise

Unter der Adresse F 200 mit 80 den Motor starten, es darf keine Nadel "an" sein. Mit 81, 82, 84 usw. können die einzelnen Nadeln angeschaltet werden, so kann die Funktion jeder einzelnen Nadel getestet werden.

Starten Sie das Drucker-Programm bei F 00E oder F 002. Nun werden zwei Zeilen ausgedruckt. Die Zeichen, die ausgedruckt werden, stehen unter der Adresse 0F00 und folgende.

Definition der Label-Adresse:

PRINT - 1 = = FFFF

Stückliste

1 Platine
1 Steckerleiste 64 Pol
1 14 Pol Wire-Wrap Stützpunkt

Integrierte Schaltkreise

IC 1	74 LS 273
IC 2	74 LS 00
IC 3	74 LS 138
IC 4	74 LS 14
IC 5	74 LS 125
IC 6 Software in 5204	
IC 7	XR 2203

Transistoren

T 1, T 3 - T 5	TUN
T 6	TUP
T 2	2N 3055 pl.

Dioden

D 1 - D 3 Z-Dioden insg. 27V-29V
1 Gleichrichter

Widerstände

R 1 - R 3, R 5, R 7	3 K 3 - 4 K 7
R 4, R 6	10K
R 8	330-470

Kondensatoren

C 1, C 2	470 my/40V
C 3	10 my

Fassungen

1 x DIL 24
1 x DIL 20
2 x DIL 16
3 x DIL 14

1 Trafo 25V
1 Druckwerk
1 Verbindungskabel
1 Cassette

```
programm: hex-matrix fuer kleindrucker
1370  c4  73  c9  06  c4  50  c9  05  c4  80  c9  04  c9  03  c9  02
1380  c9  01  c9  00  c9  ff  c4  00  ca  0f  37  c4  55  33  3f  c2
1390  08  e4  b0  9c  43  c4  78  c9  00  c4  10  c9  f1  ca  0f  c4
13a0  3e  ca  1d  c4  04  ca  0e  c4  d3  ca  12  3f  c2  02  d4  3f
13b0  dc  80  01  aa  12  01  ca  80  c2  01  d4  3f  dc  80  01  aa
13c0  12  01  ca  80  ba  0e  9c  e3  aa  12  01  c4  ff  ca  80  90
13d0  02  ca  0f  c4  0a  ca  1d  3f  c4  5f  c9  00  c4  5e  c9  ff
13e0  c4  3e  ca  1d  3f  c2  02  ca  11  c2  01  ca  10  3f  c4  ff
13f0  ca  1d  c4  ff  ca  1c  c4  b4  ca  13  c4  0f  ca  14  c2  0f
1400  98  01  3f  c2  10  31  c2  11  35  c4  d3  ca  12  35  ca  0d
1410  35  c4  02  ca  0e  c4  14  37  c4  8e  33  3f  aa  12  01  c2
1420  0c  dc  40  ca  80  aa  12  01  c2  0b  dc  40  ca  80  ba  0e
1430  98  06  31  ca  0d  31  90  dd  aa  12  01  c4  08  ca  0e  c4
1440  20  ca  80  c1  00  ca  0d  3f  c2  0c  01  aa  12  01  ca  80
1450  c2  0b  01  aa  12  01  ca  80  aa  12  01  31  e2  01  9c  0d
1460  e2  01  31  35  e2  02  98  1b  e2  02  35  90  03  e2  01  31
1470  c5  01  ba  0e  9c  c9  c4  ff  ca  80  c4  00  37  c4  55  33
1480  3f  90  86  c4  ff  ca  80  c4  00  37  c4  55  33  3f  00  c4
1490  14  37  ca  11  c4  b8  33  ca  10  c2  0d  d4  f0  1e  1e  1e
14a0  1e  01  c3  80  ca  0c  c2  0c  d4  0f  01  c3  80  ca  0b  c2
14b0  11  37  c2  10  33  3f  90  d7  30  31  32  33  34  35  36  37
14c0  38  39  01  02  03  04  05  06
.end.
```

```
f000  04 c1 f5 36 c1 f4 32 90 cd 08 06 c8 08 c8 c8 04
f010  c4 0f 36 c4 cc 32 c4 f3 35 c4 ff 31 c4 ff ca ff
f020  c4 80 c9 cc c1 01 1e 94 fb c1 01 d4 c1 9c fa aa
f030  ff 01 40 e4 20 98 c8 c2 80 c1 40 e4 ff 9c 0f c1
f040  01 a4 01 98 fa c4 cc c9 00 37 c4 14 33 3f c4 05
f050  ca fb 40 d4 3f 01 d4 c0 ca fe c4 f0 37 c4 c0 33
f060  c7 80 c7 80 c7 80 c7 80 c7 80 c4 00 c1 c2 fe 9c
f070  04 c4 c1 90 0a e4 40 9c 04 c4 c2 90 02 c4 04 ca
f080  fc c1 01 94 fc c1 01 d4 80 9c fa c1 01 d4 01 9c
f090  b4 40 9c 0a c3 00 c9 cc c9 cc c4 80 c9 cc ba fc
f0a0  9c df c7 01 ba fb 9c c5 40 9c 84 c4 c2 ca fb 01
f0b0  90 bb 15 48 c2 0c 6e 8f 51 a8 73 3f 08 13 7e db
f0c0  be c1 dd d5 de fe 89 89 89 fe ff c9 c9 c9 b6 be
f0d0  c1 c1 c1 a2 ff c1 c1 a2 9c ff c9 c9 c9 c1 ff 89
f0e0  89 89 81 be c1 c1 c9 f9 ff 86 68 88 ff 80 c1 ff
f0f0  c1 80 a0 c0 c1 bf 81 ff 88 94 a2 c1 ff c0 c0 c0
f100  c0 ff 82 84 82 ff ff 84 88 90 ff be c1 c1 c1 be
f110  ff 89 89 89 86 be c1 d1 a1 de ff 89 99 a9 c6 c6
f120  c9 c9 c9 b1 81 81 ff 81 81 bf c0 c0 c0 bf 9f a0
f130  c0 a0 9f bf c0 b8 c0 bf e3 94 88 94 e3 87 88 f0
f140  88 87 e1 d1 c9 c5 c3 80 ff c1 c1 80 98 84 88 90
f150  8c 80 c1 c1 ff 80 88 b6 c1 c1 80 c1 c1 b6 88 80
f160  80 80 80 80 80 80 80 cf 80 80 80 87 80 87 80 94
f170  ff 94 ff 94 a4 aa ff aa 92 83 93 88 e4 e2 b0 ce
f180  d9 a6 d0 80 84 82 81 80 80 9c a2 c1 80 80 c1 a2
f190  9c 80 94 86 be 88 94 86 88 be 88 88 80 c0 b0 80
f1a0  80 86 88 88 86 88 80 e0 e0 80 80 a0 90 88 84 82
f1b0  be d1 c9 c5 be 80 c2 ff c0 80 c2 e1 d1 c9 c6 a1
f1c0  c1 c5 cb b1 98 94 92 ff 90 a7 c5 c5 c5 b9 bc ca
f1d0  c9 c9 b0 81 f1 89 85 83 b6 c9 c9 c9 b6 86 c9 c9
f1e0  a9 9e 80 e3 e3 80 80 c0 b3 80 80 88 94 a2 c1
f1f0  80 94 94 94 94 94 80 c1 a2 94 88 82 81 d1 89 86
.end.
```

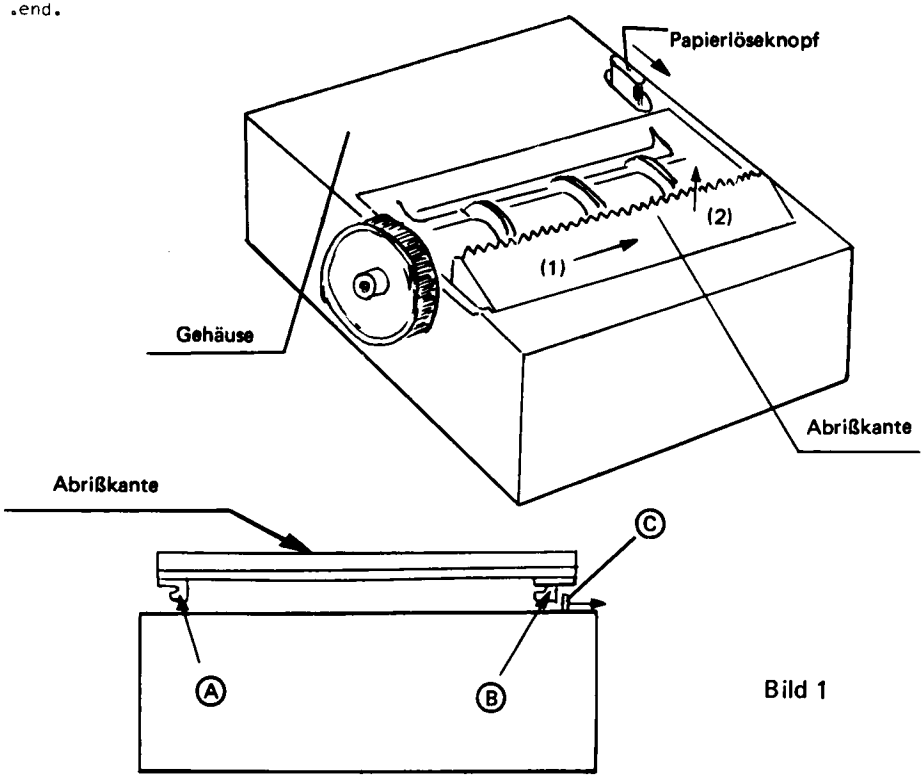
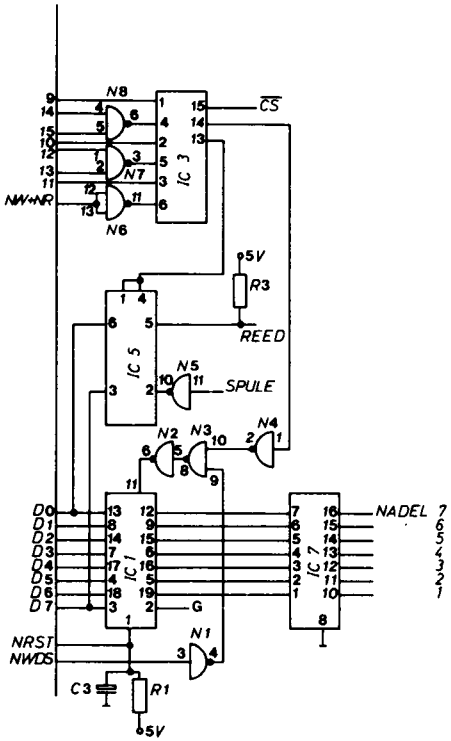
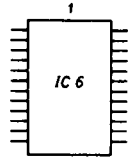


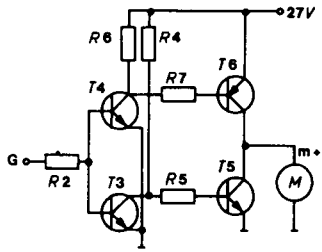
Bild 1

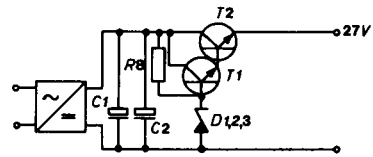
SCHALTBILD





5204
software



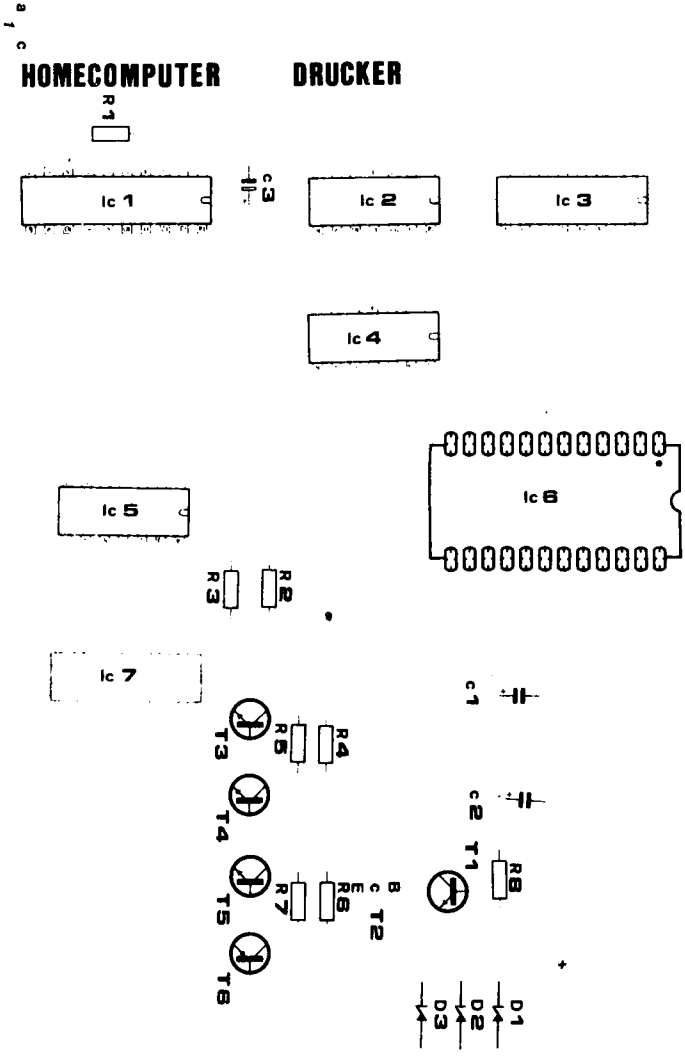


reed o spule
masse o om+
o o27V
nc o onadel 1
2 o o 3
4 o o 5
6 o o 7

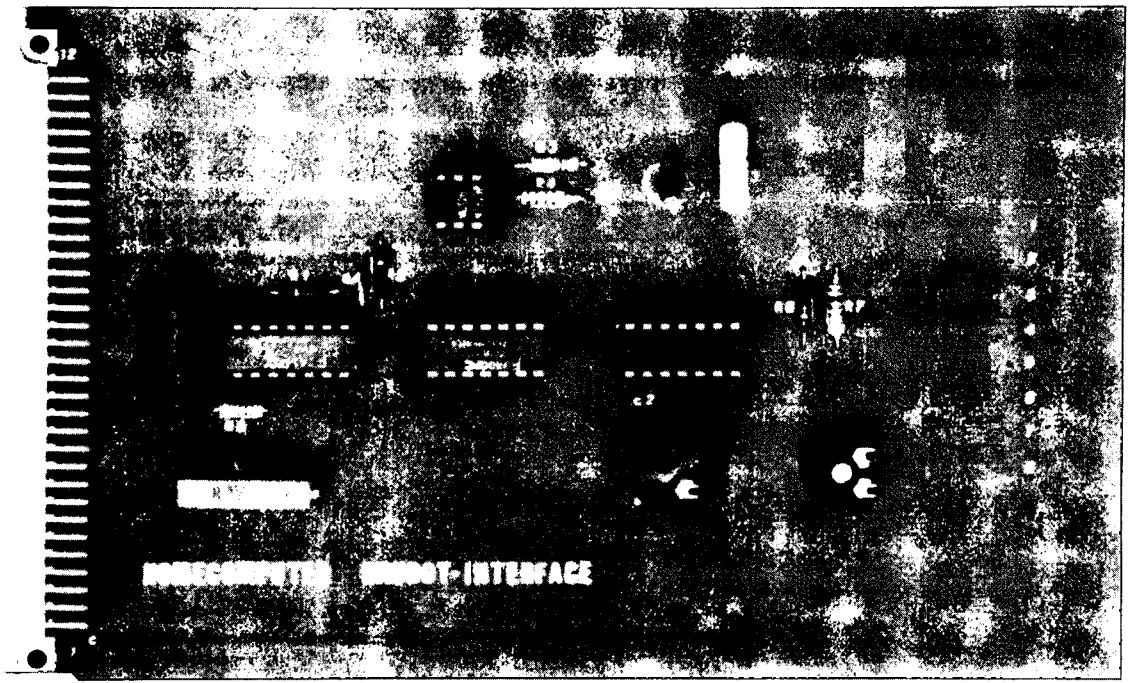
PLATINEN RAND STECKER v. oben

N1,2,4,5 = IC 4
N3,6-8 = IC 2

BESTÜCKUNGSPLAN



Baudot - Interface



Eine andere preiswerte Druckausgabe ist über einen Fernschreiber möglich. Die erforderliche Schnittstelle befindet sich auf dem Baudot-Interface. Hieran kann ein handelsüblicher Postfernschreiber mit 40mA Stromschleife sowie ein Lochstreifenleser für 5 oder 8 Bit Lochstreifen angeschlossen werden.

Beschreibung

Das Interface befindet sich auf einer einseitigen Europakarte. Mit einer 64 pol. Steckerleiste kann es direkt auf den SC/MP-Bus gesteckt werden. Die Spannung + 5 V, - 12 V, + 12 V werden direkt vom Bus entnommen. Am Kopfende der Karte befinden sich Bohrungen zur Aufnahme von "Lötnäglen" oder Schraubklemmen, an die peripheren Geräte (Postfernschreiber, Microleru) angeschlossen werden.

Mit drei Potentimetern können der Ruhestrom des Fernschreibers und die Baudrate des Lochstreifenlesers eingestellt werden.

Stückliste

1 Platine
1 Steckerleiste 64 pol.

Integrierte Schaltkreise

IC 1	TIL 112 (o.ä.)
IC 2	7414
IC 3	7400
IC 4	74121

Widerstände

R 1 u. 2	150
R 3	390
R 4	1 K 5
R 5	470
R 6	560
R 7	1 K

Kondensatoren

C 1	220 n
C 2	100 n
C 3	*
C 4	820 p

Transistoren

T 1	BC 307 (0.ä.)

Trimmer

P 1	1 K (Spindel)
P 2 u. 3	500 E

* Für C 3 kann ein Entstörkondensator 1 uF MKM eingebaut werden.

Erklärung

Die Fernschreibzeichen müssen über das Flag 0 seriell ausgegeben werden.
Diese Signale steuern über die Inverter N 1, N 2, N 3 und den Vorwiderstand R 2 der Optokoppler. Bei einer "1" an Flag 0 ist der Fototransistor gesperrt. In diesem Zustand ist der über den Widerstand R 3 nachgeschaltete PNP Transistor ebenfalls gesperrt, und in der Fernschreiberschleife fließt kein Strom. Im anderen Fall fließt ein Strom durch den an den Anschlüssen 2 und 8 angeschlossenen Fernschreiber. Dieser Strom kann mit dem Trimmer P 3 auf 40 mA abgeglichen werden. Soll die Tastatur des Fernschreibers benutzt werden, so ist diese mit den Anschlüssen 5 und 6 zu verbinden. Die Signale des Fernschreibers gelangen dann über einen Inverter an Sense B der CPU.

Für einen gesteuerten Lochstreifenleser sind die Anschlüsse 1, 3, 4, 5 und 7 vorgesehen. An Punkt 7 liegen + 5 V und an Punkt 1 0V an. Anschluß 5 ist der Dateneingang, Anschluß 4 der Baudrate-Generator und Anschluß 3 der Steuergang für den Motor.

Durch Setzen von Flag 2 der CPU wird das Monoflop (IC 4) getriggert und schaltet dabei das Flip-Flop N 8, N 9.

Die Impulslänge des Monoflops läßt sich über P 2 einstellen. Das Flip-Flop gibt nun den Motor des Lesers frei. Das Startbit des gelesenen Zeichens liegt nun an Sense B und am Flip-Flop an, während das Flip-Flop zurückgesetzt wird, werden die übrigen Datenbits übertragen. Die Baudrate läßt sich am Oszillator N 4, N 5 durch das Poti P 1 einstellen. Durch Rücksetzen und Setzen von Flag 2 kann das nächste Zeichen abgerufen werden.

Sollen Fernschreiber-Tastatur und Lochstreifenleser gleichzeitig angeschlossen werden, sehen Sie einen Umschalter an Anschluß 5 vor.

Aufbauhinweise

Zum Aufbau siehe allgemeine Aufbauhinweise.

Testhinweise

Prüfen Sie alle Versorgungsspannungen.

Schalten Sie die Spannungen ab und schließen Sie den Fernschreiber an den Punkten 2, 5, 6 und 8 an.

In die Leitung zu Punkt 2 schalten Sie ein DC-Amperementer, Meßbereich 50 mA (+ an Punkt 2).

Schalten Sie die Versorgungsspannung und den Fernschreiber ein. Nach dem Abgleich (P 3) auf 40 mA darf die Maschine nicht durchlaufen. Schreiben Sie nun folgendes Programm in den RAM-Bereich und starten Sie es bei X'0C00.

```
0C00  06
0C01  1C
0C02  1C
0C03  1C
0C04  1C
0C05  1C
0C06  07
0C07  90 F 7
```

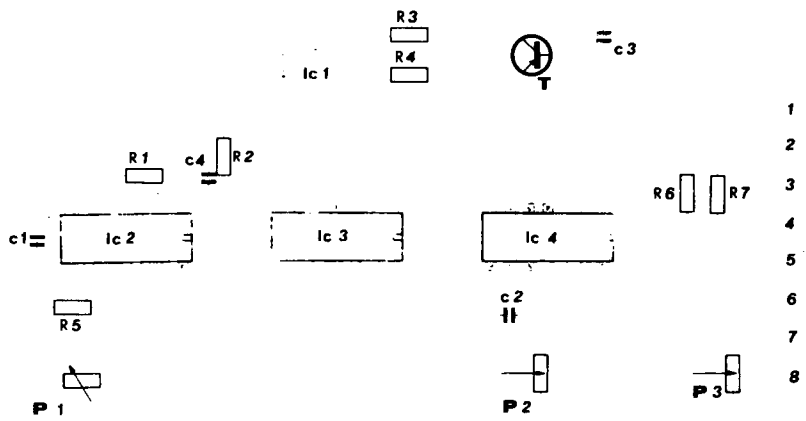
Sie müssen nun mit der Fernschreiber-Tastatur auf dem Fernschreiber schreiben können. Falls fehlerhafte Zeichen auftreten, setzen Sie Kondensator C3 (1uF) ein.

Schließen Sie den Lochstreifenleser nach bedienungsanleitung an den Punkten 1, 3, 4, 5, und 7 an.

Setzen Sie nun Flag 2 der CPU auf "1". Der Motor des Lesers muß nun laufen. Legen Sie einen Lochstreifen ein. Beim Erreichen des ersten Zeichens muß der Motor anhalten. Mit Poti P1 können Sie nun die gewünschte Baudrate einstellen (an Punkt 4 liegt das 16-fache dieser Baudrate).

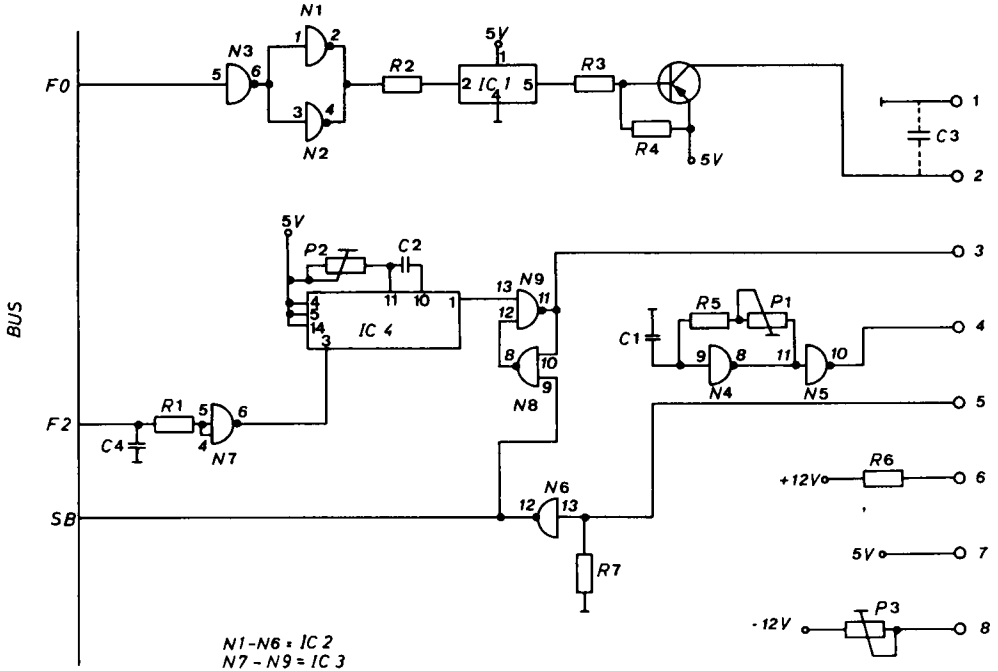
Poti P 2 ist so einzustellen, daß der Lochstreifen jeweils nur um ein Zeichen weitertransportiert wird.

BESTÜCKUNGSPLAN

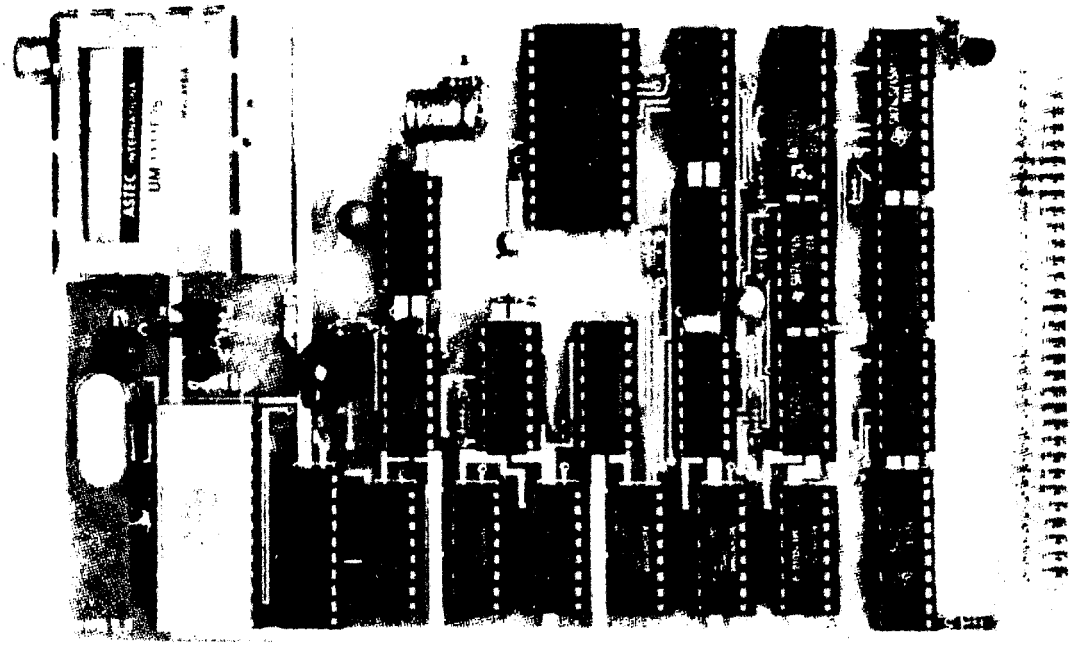


HOMECOMPUTER BAUDOT-INTERFACE

SCHALTPLAN


BUS
F0
F2
SB
N1
N2
N3
R2
5V
IC 1
R3
R4
5V
C3
1
2
P2
C2
IC 4
N9
3
C1
R5
P1
N4
N5
4
N8
R1
C4
N7
5
N6
R6
+12V
6
R7
5V
7
-12V
P3
8
N1-N6 = IC 2
N7 - N9 = IC 3

TV - Interface



Nachdem wir nun mehrere Möglichkeiten der schriftlichen Ausgabe von Daten aufgezählt haben, folgt nun die Darstellung von größeren Datenmengen auf einem TV-Gerät oder einem Video-Monitor. Auch hier ist wieder ein Bindeglied erforderlich. Dieses stellt das TV-Interface dar. Ohne Modulator kann es an einem Monitor angeschlossen werden, mit Modulator an den Antenneneingang eines handelsüblichen Fernsehgerätes.

HC - TV

Beschreibung

Das HC - TV Interface ist auf einer doppelseitigen, beschichteten und durchkontaktierten Karte im Europaformat mit einer 64 pol. Steckerleiste nach DIN 41612 versehen, passend zum SC/MP Bus.

Die Karte wird wie ein normaler RAM-Speicher angesprochen. Der RAM-Bereich auf der Karte hat eine Größe von 1K x 6 Bit. Verwendet werden RAM's vom Typ 2102. Jede Speicheradresse hat eine zugeordnete Stelle auf dem Bildschirm, somit kann jede Stelle frei adressiert und geändert werden (direkter Speicherzugriff beim Schreiben wie beim Lesen).

Insgesamt werden 16 Zeilen á 64 Zeichen dargestellt. Der TV-RAM-Speicher liegt im Adreßbereich FC 00 - FFFF. Das Interface verfügt über einen Charaktergenerator von 64 Zeichen im ROM. Der Zeichensatz umfasst ASCII-Codierte Großbuchstaben, Zahlen und Sonderzeichen.

Zur Erzeugung des Videosignals wird ein IC der Firma Thommson verwendet. Gegenüber den herkömmlichen Videointerfaces bei denen die Daten mit einer geringeren Baudrate eingeschrieben werden können, und nur über den gesteuerten Cursor zu ändern sind, kann der Bildschirm bei diesem Interface mit einer Geschwindigkeit vollgeschrieben werden, mit dem der RAM-Bereich geladen wird. Im günstigsten Falle geschieht dies unter 100 m/sec.

Ebenfalls kann bei diesem Interface der Bildinhalt sofort in einen beliebigen Speicher, sei es RAM-Bereich oder eine Cassette umgeladen werden.

Das Interface kann mittels eines Modulators an den Antenneneingang eines beliebigen handelsüblichen Fernsehgerätes angeschlossen werden.

Gegebenenfalls können Sie auch videomäßig in Ihr Fernsehgerät eingreifen. Wie bei allen unseren Bausätzen sind alle Signale gepuffert.

Stückliste

TV Interface

1 Platine Europakarte durchkontaktiert
1 Steckerleiste durchkontaktiert

Integrierte Schaltkreise

IC 1, 4 und 22	74 LS 245
IC 8, 12, 15, 16, 18, und 21	2102
IC 3 und 7	74 LS 125
IC 10	74 LS 00
IC 6	74 LS 14
IC 5	74 LS 374
IC 20	74 LS 132
IC 19	74 LS 121
IC 9	74 LS 174
IC 14	74 LS 165
IC 17	74 LS 163
IC 2	74 LS 30
IC 11	74 LS 74
IC 23	SFF 96 264
IC 13	RO 3-2513

Transistoren

T 1, 2 und 3	TUN

Dioden

D 1 und 2	DUS

Widerstände

R 1 und 10	1 K

R 2, 3, 5, 8, 16, 17, und 15	3K3 - 4K7
R 4	390
R 6	330
R 7	15 M
R 9 und 12	470
R 11	680
R 13	82
R 14	75

Tantalkondensatoren

C 4, 5 und 6	10 - 22 my

Elektrolytkondensatoren

C 7	10 my
C 8	100 my
C 9	22 my

Kondensatoren

C 1	680 p
C 2 und 3	82 p

Poti

P 1	100

Quarz

1 Quarz	1008 Khz

Option

UHF oder VHF Modulator

Erklärung

Interner Ablauf

Das gesamte interne Timing der TV-Karte wird von dem IC SFF 96364 der Firma Thompson übernommen. Durch Anschalten eines Zählers

(IC 17) und eines Schieberegisters (IC 14), die durch den Controller (IC 23) getaktet werden, wird der ASCII-Charakter Generator angesteuert. Dieser gibt in Abhängigkeit der anliegenden Daten die erste Zeile für das gewählte ASCII-Zeichen heraus. Über die Adreßausgänge des Controllers werden die Dateninformationen aus der RAM-Adresse gelesen und einem Latch (IC 9) zwischengespeichert. Über den Video-Verstärker (T 2 + T 3) gelangt das Video-Signal des Controllers an den Modulator.

Die Cursor-Steuerung des Controllers wird nicht ausgenutzt, da dies das Interface stark verlangsamen würde. Um den blinkenden "Cursor" (der immer vorhanden ist) immer an der gleichen Stelle (oben links am Bildschirm) sichtbar zu machen, wird durch einen "power on reset" (IC 19) der "Cursor" an diese Stelle gelenkt.

An Punkt B kann durch kurzes Masseanlegen ggf. dieser Reset auch erreicht werden. Bei diesem Interface wird die Cursor-Steuerung durch Software (wo nötig) realisiert.

Externer Ablauf

Wird der Adreßbereich FC00 - FFFF angesprochen, ist ein bestimmtes Feld auf dem Bildschirm angewählt. Mit dem NWDS werden die neuen Daten in die RAM's übernommen. Während dieser Zeit sind die Adreß-Puffer des Controllers hochohmig, da jetzt die Adressen vom System-Bus an den RAM's anliegen. Mit dem NRDS werden die eingeschriebenen Daten in IC 5 zwischengespeichert, wo sie dann gelesen werden können. Sollte die CPU das Interface ansprechen, während der Controller die RAM's ausliest, wird die CPU über dem N-Hold angehalten und erst wieder freigegeben, wenn die RAM's vom Controller nicht mehr ausgelesen werden (z.B. Zeilenrücklauf). Deshalb muß eine evtl. Brücke auf dem BUS oder der CPU-Karte von N-Hold gegen 5V entfernt werden.

Auf der Platine sind zwei Punkte mit "V" bezeichnet, einer davon ist mit einem dritten Punkt verbunden. Hier kann das Bild invertiert werden, wenn die Verbindung zum ersten "V" unterbrochen und mit dem zweiten "V" verbunden wird (ggf. mit Schalter umschaltbar).

Sollte die Ausgangsspannung des Modulators zu groß sein (Bild syn-

chronisiert nicht), kann ein Widerstand von 82 Ohm vom Ausgang des Modulators gegen Masse gelegt werden.

Der auf der Platine befindliche Poti dient zur Bildbreiten-Einstellung.

Aufbauhinweise

Die ICs 13, 14 und 23 sind entgegen den anderen ICs einzusetzen. Es empfiehlt sich folgende Bauteile stehend einzulöten:
R6, R11, R10, C8, C9 und C7. Der Trimmpoti wird ebenfalls stehend eingelötet.

Testhinweise

Wenn ein Bild vorhanden ist:
Testen Sie den Bildspeicher (FC00 - FFFF) bei einer beliebigen Adresse mit Modify für jedes einzelene Bit (6 Bit). Z.B.: M0 FC00 00, jetzt muß auf dem Hexdisplay unter der Date C0 stehen. 01 ergibt C1 usw. da nur 6 Bit verarbeitet werden. Gegebenenfalls suchen Sie bei den RAMs nach Kurzschlüssen. Nun geben Sie folgendes Programm ein:

```
C4 00
31
C4 FC
35
C4 20
CD 01
90 FC
```

Das Programm löscht den Bildschirm (Laden des Bildspeichers mit 20). Nun nochmals mit Modify die Adresse überprüfen, ob evtl. mehrere Bildstellen gleichzeitig angesprochen werden. Da das komplette Austesten des Bildspeichers zu langwierig wäre, empfiehlt es sich, dies mit einem Programm zu tun. Es können z.B. Fehler beim Rollen des Bildes im Speicher entstehen. Der Text kann nun mit einem in BASIC geschriebenen Programm erfolgen (a) oder mit einem Programm, welches auf der Cassette mit den kleinen Betriebsprogrammen zu finden ist (b).

Das RAM Diagnose-Programm kann nicht verwendet werden, da nur 6 Bit Speicher vorhanden sind.

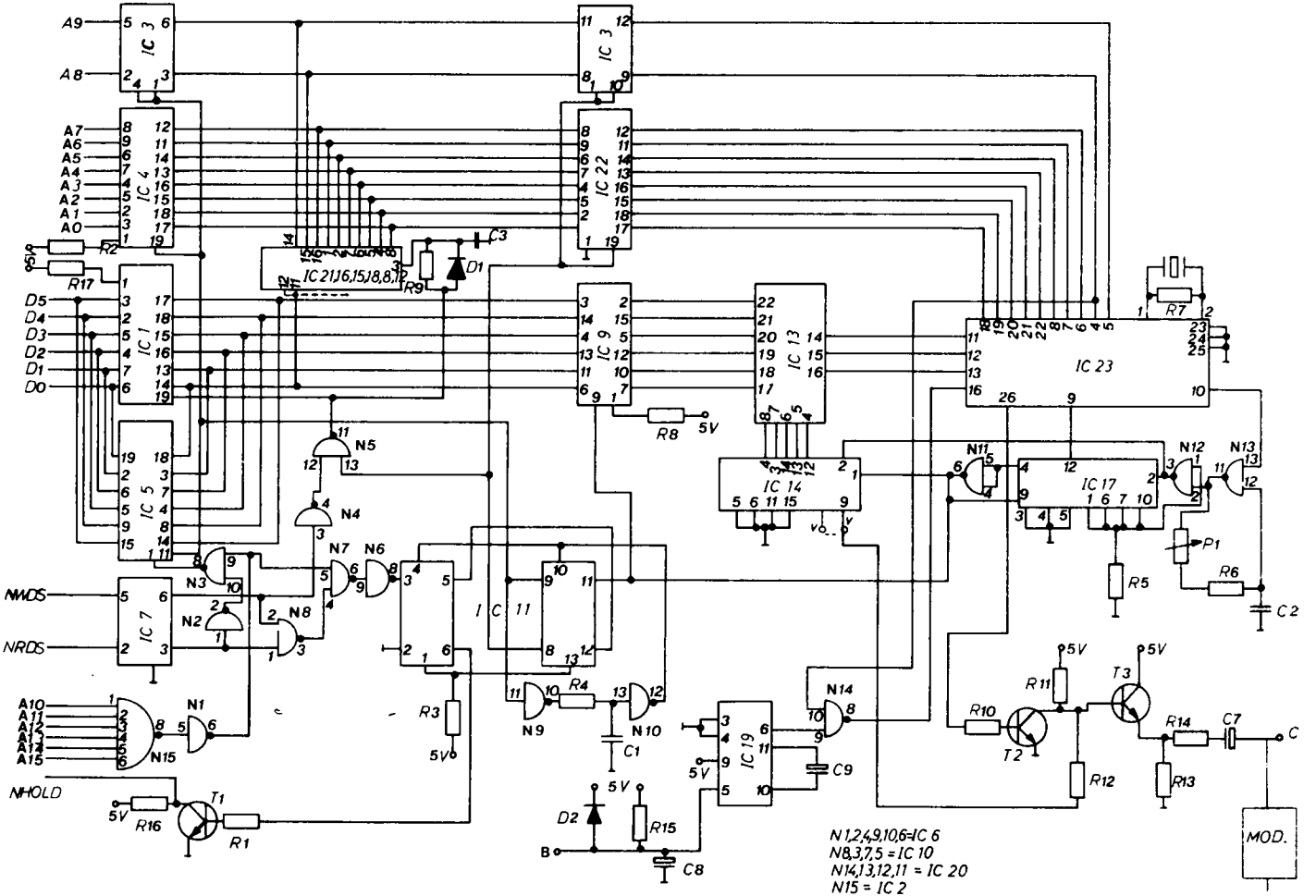
a.
10 PR "AAA";:GOTO 10
Der Bildschirm wird mit der Dreiergruppe AAA fortlaufend vollgeschrieben, wenn der Bildschirminhalt nach oben rollt, lässt sich sofort erkennen, ob der Bildspeicher in Ordnung ist.

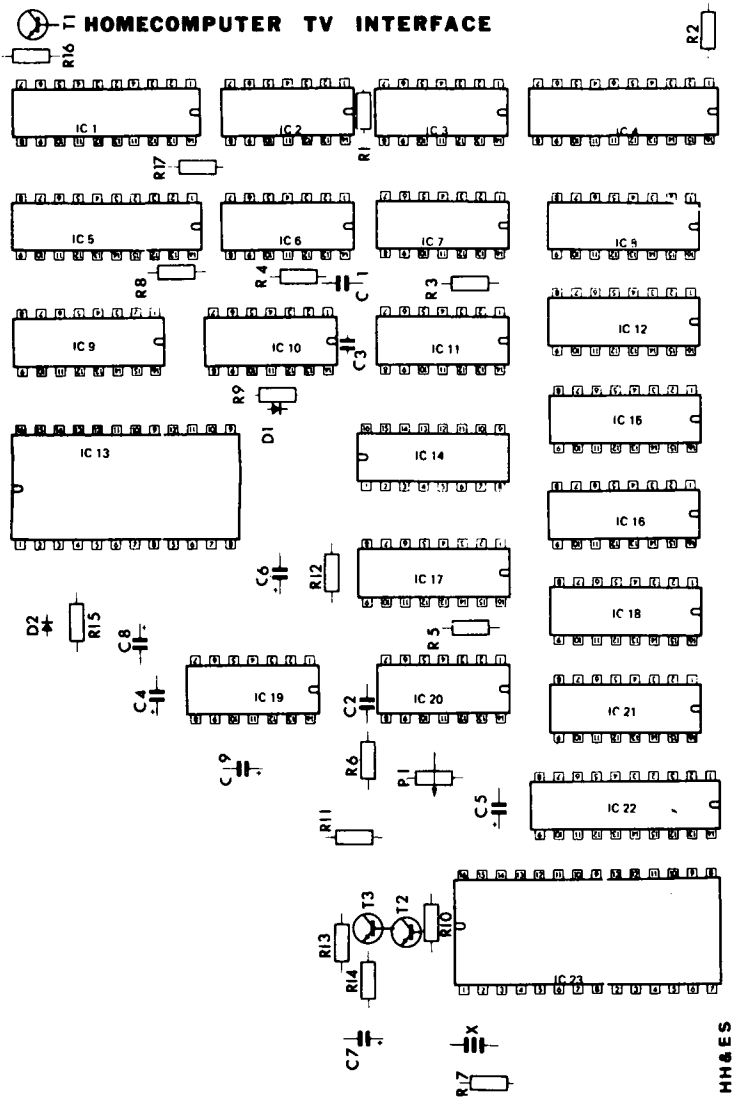
b.
Mit dem Programm 12 auf der Cassette "Kleine Betriebsprogramme" werden alle möglichen ASCII-Zeichen des Charaktergenerators auf den Bildschirm geschrieben. Auch hiermit ist eine schnelle Kontrolle des Bildspeichers möglich.

Wenn kein Bild vorhanden ist:

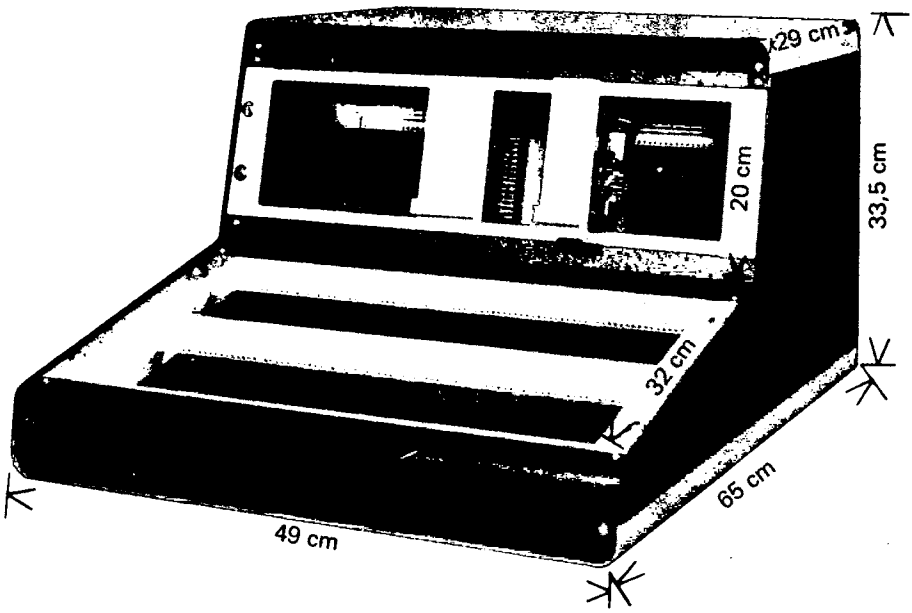
Kontrolle ob der Quarz schwingt, Taktsingal an PIN 10 des Controllers weiter verfolgen nach IC 17 und IC 14. Anschluss 26 des Controllers ist das Videosignal, welches über die Transistoren T1 und T3 zu verfolgen ist.

Wird die Karte nicht angesprochen oder bleibt der SC/MP nach dem Ansprechen sofort stehen, nach Kurzschlüssen in der Adress-Decodierung suchen (IC 2, 11, 10 und 6). Ebenso prüfen, ob die NHOLD Schaltung arbeitet. Beachten Sie die Testhinweise für Speicherkarten.


A9
A8
A7
A6
A5
A4
A3
A2
A1
A0
IC 3
IC 4
IC 22
IC 1
IC 5
IC 9
IC 13
IC 23
IC 14
IC 17
IC 7
IC 11
IC 19
IC 21,16,15,18,8,19
R2
R17
R9
D1
C3
D5
D4
D3
D2
D1
D0
R8
5V
N5
N4
N7
N6
N3
N2
N8
N11
N12
N13
P1
R5
R6
C2
R7
NWDS
NRDS
A10
A11
A12
A13
A14
A15
N1
N15
NHOLD
T1
R16
R1
R3
R4
N9
N10
C1
D2
R15
C8
B
N14
C9
R11
T3
R10
T2
R14
C7
C
R12
R13
MOD.
N1,2,4,9,10,6=IC 6
N8,3,7,5 = IC 10
N14,13,12,11 = IC 20
N15 = IC 2


HOMECOMPUTER TV INTERFACE
T1
R16
R2
IC 1
IC 2
IC 3
IC 4
R1
R17
IC 5
IC 6
IC 7
IC 8
R8
R4
C1
R3
IC 9
IC 10
C3
IC 11
IC 12
R9
D1
IC 13
IC 14
IC 15
IC 16
C6
R12
IC 17
IC 18
D2
R15
C8
R5
C4
IC 19
C2
IC 20
IC 21
C9
R6
P1
C5
IC 22
R11
T3
T2
R13
R10
R14
IC 23
C7
X
R17
HH&ES

Computergehäuse



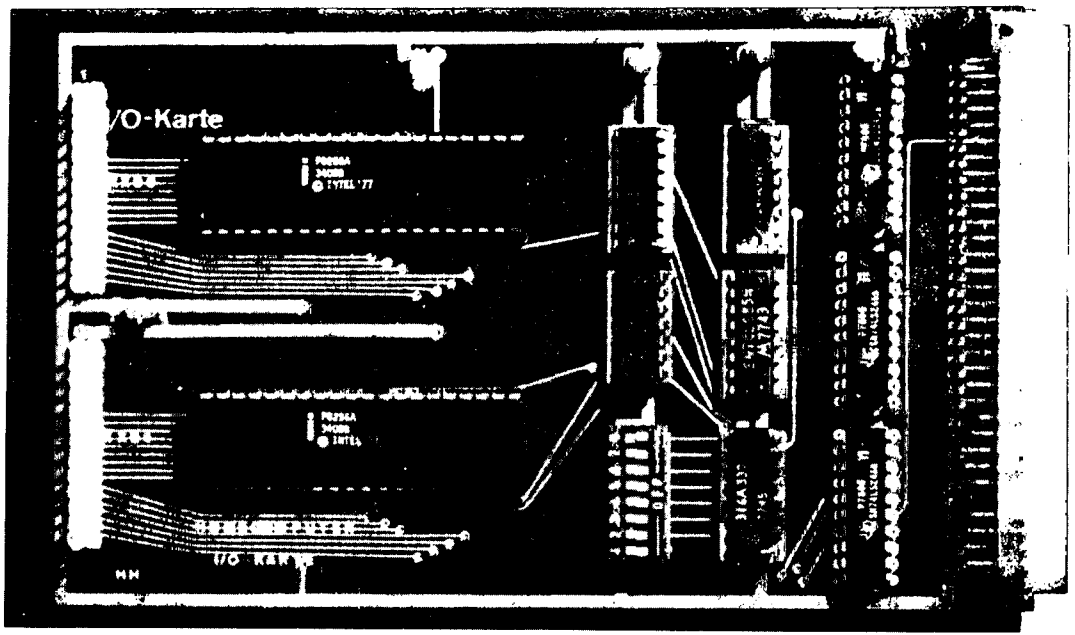
Das Gehäuse für Microprocessor-Systeme bietet neben einem geräumigen Tastaturpult die Möglichkeit, im oberen Frontteil Bedienungselemente und Displays unterzubringen. Wahlweise kann auch ein Baugruppenträger von vorne eingesetzt werden. Das Gehäuse hat 6 Höheneinheiten und bietet somit Platz für zwei Baugruppenträger. Die hintere Abdeckplatte ist im Preis inbegriffen. Tastaturplatte und vordere obere Platte müssen gesondert bestellt werden. Das Gehäuse gibt es in den Farben schwarz/orange. Frontplatten sind Aluminium. Alle Maße entsprechen internationaler Normung.

Für die Befestigung der Baugruppenträger sind Käfigmuttern erforderlich, die am Gehäusekorpus eingerastet werden. Jeder Baugruppenträger erfordert 4 Käfigmuttern (12 Käfigmuttern, 4 Schrauben und Scheiben sind jedem Gehäuse beigegeben).

Zur Anbringung der Busplatine werden Z-Schienen geliefert. Bei der Bestellung muß angegeben werden, ob ein anderer Bus verwandt wird; pro Baugruppenträger sind 2 Z-Schienen erforderlich. 2,5 mm Schrauben und Muttern werden zur Montage benötigt.

Bei Einschieben der einzelnen Platinen verwendet man Kartenführungen, die im Baugruppenträger eingerastet werden. Je Platine sind zwei Kartenführungen erforderlich.

48 I/O Lines



Beschreibung

Sollen an ein Mikroprozessorsystem externe Elemente angeschlossen werden, muß das System über eine serielle oder parallele Schnittstelle verfügen. Eine parallele Schnittstelle stellt diese Karte dar. Es können direkt ICs, Transistoren etc. angeschlossen werden, die dann z.B. Relais oder sogar Triacs ansteuern können. Auch ist es z.B. möglich, mit dieser Karte eine Speicherbereichserweiterung (größer als 64K) zu schaffen.

Eigene Schaltungen lassen sich sofort adaptieren, da die I/O Lines des ICs als normale TTL Ein- Ausgänge behandelt werden können. Es ist nicht mehr erforderlich, sich um das Bustiming zu kümmern.

Die 48 I/O Lines Karte ist im Europaformat und mit einer 64 pol. Steckerleiste versehen passend zum SC/MP System.

Als I/O Baustein wird das IC 8255 verwendet. Seine Betriebsart (Verwendungsart) wird durch Software gesteuert.

Jeweils 24 Leitungen sind auf einen 26 pol. Pfosten herausgeführt. Die Adresse der Karte wird an einem 8 pol. DIL-Schalter eingestellt. Alle auf der Karte benötigten Signale sind gepuffert und alle TTLs sind in LS-Ausführung.

Erklärung

Als Bustreiber wird auch hier das IC 74 LS 245 eingesetzt. Die Adreß-Decodierung erfolgt durch den 1 aus 8 Decoder 74 LS 138, die Adreß-Auswahl übernehmen zwei 4 Bit Vergleicher in Kaskade geschaltet (74 LS 85). Die Umschaltung der Datenbuffer erfolgt mit dem NRDS und einem CS, der einen der beiden 8255 auswählt. Dieser Baustein kann so angesprochen werden, als ob er ein normales RAM wäre. Unter einer bestimmten Adresse lassen sich Daten lesen oder auch einschreiben mit dem Unterschied, daß eine Hardware außerhalb des Mikroprozessorsystems diese Daten weiter verarbeitet bzw. Daten der externen Hardware in das System eingegeben werden können (ASCII Tastatur, Drucker etc.).

Die niederwertigen Adressen der beiden Bausteine sind durch die Hardware der Karte folgendermaßen festgelegt:

1. Niederwertige Adressen des mit 00 bezeichneten 8255:

Kontrollwort	Adr. 03
Port A	" 00
Port B	" 01
Port C	" 02

2. Niederwertige Adressen des mit 04 bezeichneten 8255:

Kontrollwort	Adr. 07
Port A	" 04
Port B	" 05
Port C	" 06

Stückliste

1 Platine
1 Steckerleiste 64 pol.
2 Wire-Wrap Stützpunkte 26 pol.

Integrierte Schaltkreise

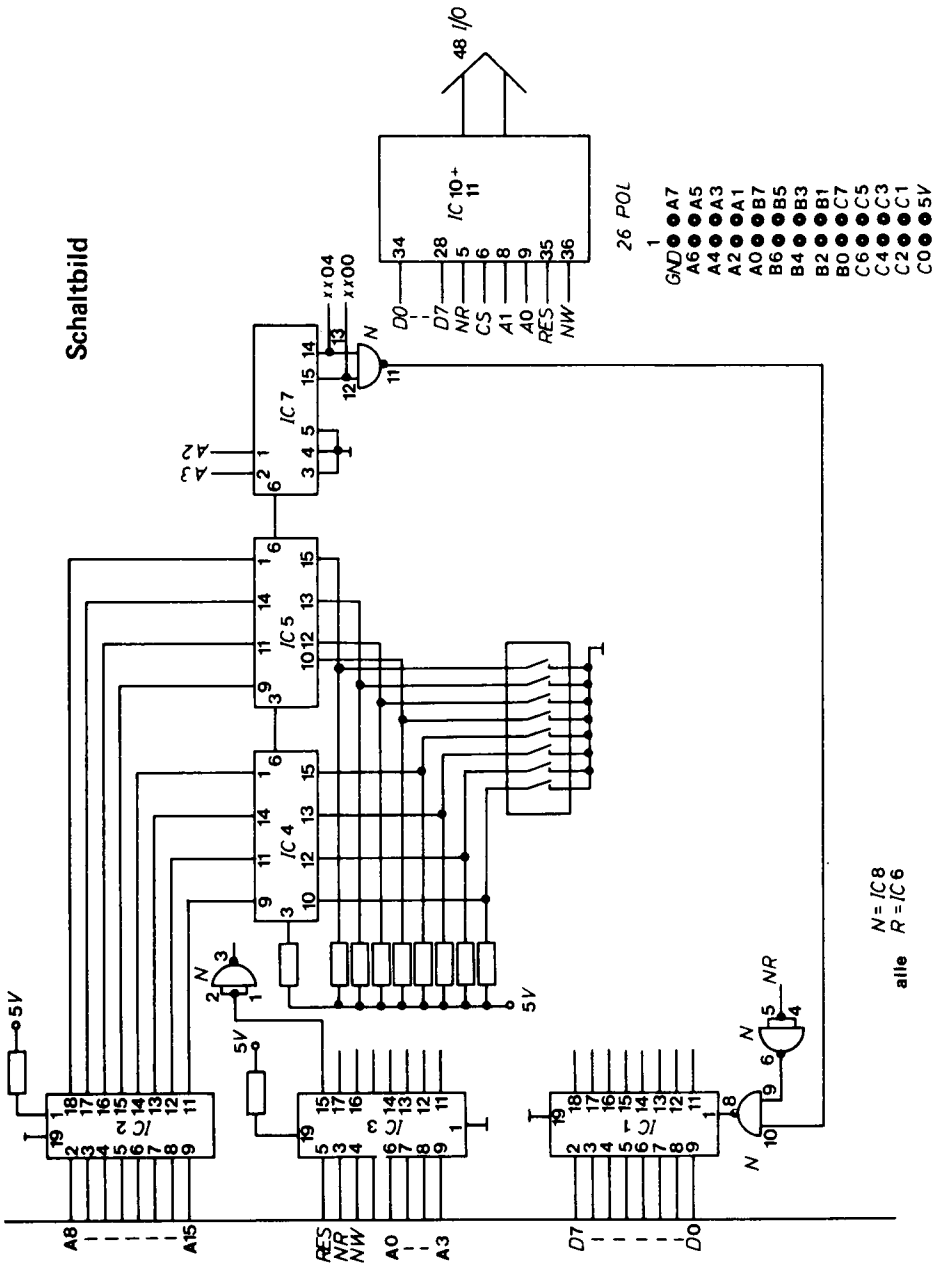
IC 1 - 3	74 LS 245
IC 4 - 5	74 LS 85
IC 6	Widerstandsarray
IC 7	74 LS 138
IC 8	74 LS 00
IC 10, 11	8255

Kondensatoren

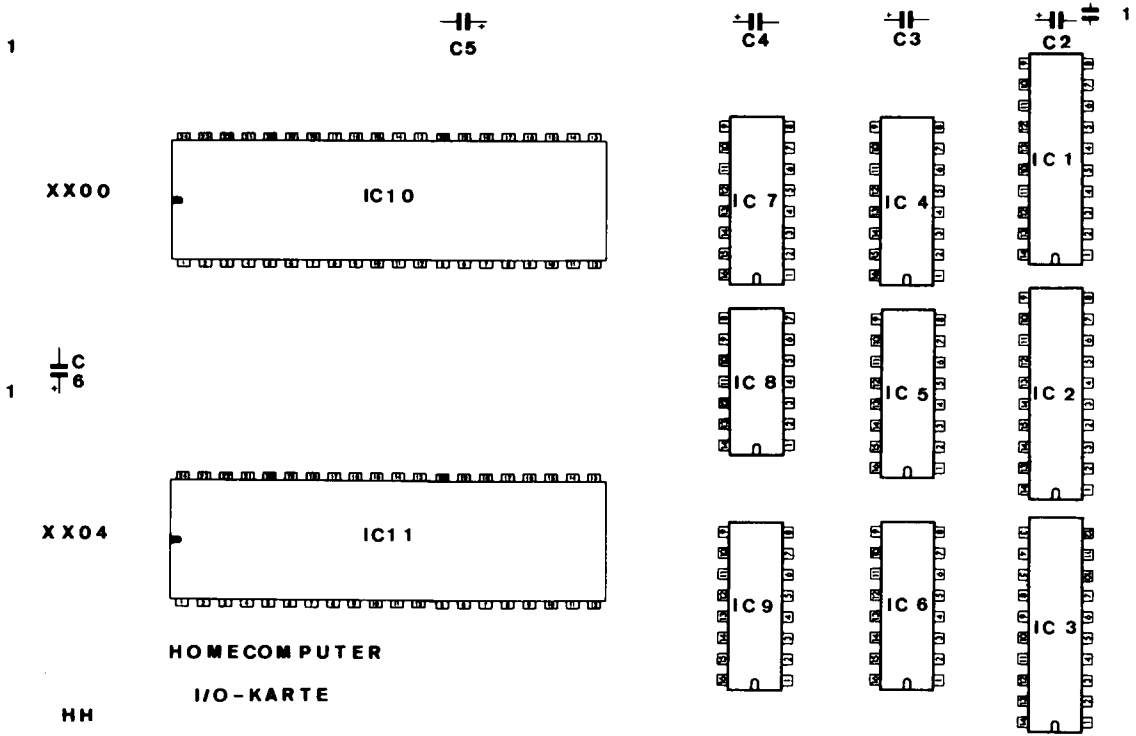
C 1	100n
C 2 - 6	22my Tantal

Fassungen

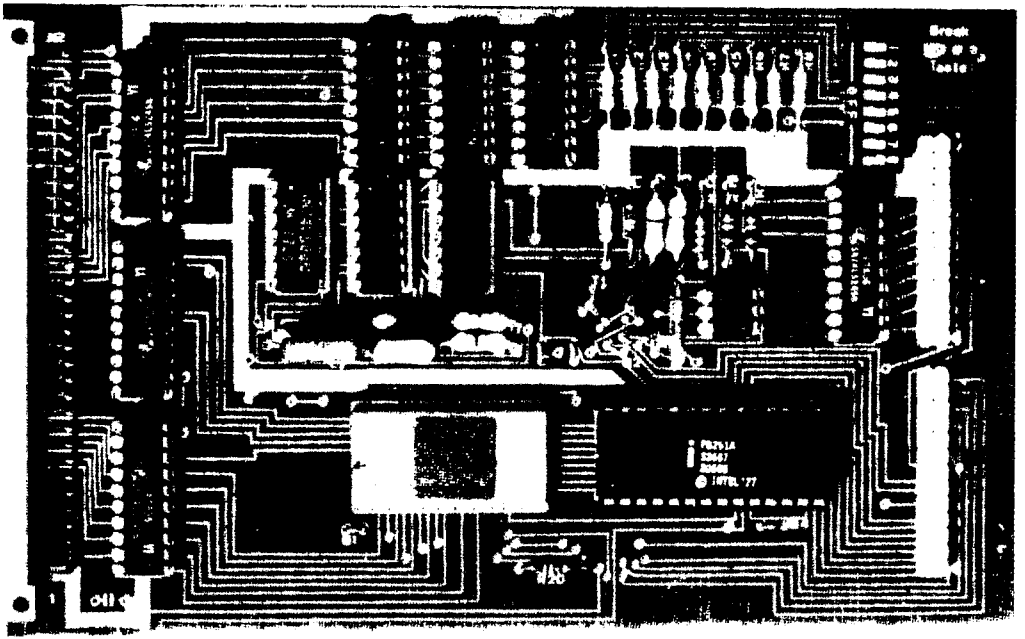
2 DIL 40
3 DIL 20
4 DIL 16
1 DIL 14
1 DIL 8 Schalter


Schaltbild
48 I/O
IC 10
IC 7
IC 5
IC 4
IC 2
IC 3
IC 1
xx04
xx00
26 POL
GND A7 A6 A5 A4 A3 A2 A1 A0 B7 B6 B5 B4 B3 B2 B1 B0 C7 C6 C5 C4 C3 C2 C1 C0 5V
D0 D7 NR CS A1 A0 RES NW
A8 A15
RES NR NW A0 A3
D7 D0
5V
N
NR
N = IC 8
R = IC 6
alle

Bestückungsplan



USART - Timer - ASCII - TTY - Interface



Beschreibung

Sollen an ein Mikroprozessorsystem externe Elemente angeschlossen werden, benötigt das System ein paralleles oder serielles Interface. U.a. ist auf dieser Karte ein serielles Interface untergebracht.

Serielle Interfaces haben den Vorteil, mit wenigen Verbindungsleitungen eine Verbindung zwischen zwei Geräten herzustellen. Diese Schnittstelle wird z.B. bei Terminals, Modems etc. verwendet. Als Nebenprodukt enthält diese Karte eine parallele Schnittstelle zum Anschluß einer ASCII-Tastatur.

Das ASCII, USART, Timer, TTY-Interface befindet sich auf einer Europakarte und ist mit einer 64 pol. Steckerleiste versehen, passend zum SC/MP System. Als USART wird das IC 8251 und als Timer das IC 8253 eingesetzt.

Wie aus den vorgenannten Erklärungen zu ersehen ist, sind auf der Karte 4 Funktionsgruppen vereint.

1. ASCII Interface
2. Timer Interface
3. USART Interface
4. TTY Interface

Als Grundaufbau ist das ASCII Interface zu sehen, alle weiteren Funktionsgruppen können nachgerüstet werden. Im Endausbau stellt die Karte eine sehr komfortable (da durch einfache Software zu programmieren) und an jeden seriellen Datenstrom anzupassende serielle Schnittstelle dar (mit oder ohne TTY, RS 232).

Außerdem stehen noch 2 Timer zur freien Verwendung zur Verfügung. Mittels eines DIL 8 Schalters kann die Adresse der Karte eingestellt werden.
Natürlich sind wie bei allen unseren Karten alle auf der Karte benötigten Signale gepuffert und alle TTLs in LS-Ausführung.

Erklärung ASCII Interface

Als Bustreiber wird wieder das IC 74 LS 245 eingesetzt. Die Adreß-Decodierung für alle auch später hinzugefügten OPTIONEN sind in dieser Ausbaustufe schon vorhanden. Die Erzeugung des CS übernimmt wieder der 1 aus 8 Decoder 74 LS 138, welcher von zwei in Kaskade geschalteten 4 Bit Vergleichern (74 LS 85) enabelt (freigegeben) wird. Durch die zwei 4 Bit Vergleicher werden die ADR 8 - 15 vom Systembus und das am DIL 8 Schalter eingestellte Wort verglichen. Bei Gleichheit (Karte ist angewählt) gibt der Vergleicher ein High-Signal aus. Der Datenpuffer wird durch diesen enabelt und durch das NRDS Signal umgeschaltet.

Als Anschluß für die ASCII Tastatur wird ebenfalls ein 74 LS 245 eingesetzt. Dieser ist so lange TRI-STATE, bis er vom CS des LS 138 und dem NRDS Signal freigegeben wird. In diesem Falle kann die 8 Bit Information unter dieser ADR. (xx00) gelesen werden.

Die Belegung des 26 pol. Steckers entnehmen Sie dem Schaltbild.

Timer

Der TIMER (Zähler) ist durch Software programmierbarer Rückwärtszähler. Eine Eingangsfrequenz wird durch einen bestimmten Wert (der durch die Software festgelegt wird) geteilt.

Durch verschiedene Betriebsarten kann der Timer unter bestimmten Bedingungen bzw. in einer bestimmten Art die Eingangsfrequenz teilen. Als Timer wird das IC 8253 eingesetzt. In diesem IC sind drei solcher TIMER als 16 Bit Zähler untergebracht.

Jeder der 3 Timer kann getrennt und unterschiedlich in 6 verschiedenen Betriebsarten benutzt werden. Ein Kontrollwort bestimmt die Betriebsart, Zählart und die Auswahl des Timers.

Ein ebenfalls auf der Karte befindlicher Quarzoszillator erzeugt eine Frequenz von 6, 144 MHz. Diese wird durch hintereinander geschaltete D Flip-Flops durch 4 geteilt, so daß dann eine Frequenz von 1,536 MHz zur Verfügung steht. Diese kann als Eingangsfrequenz für den Timer und als Systemclock für den USART benutzt werden.

Einer der drei Timer wird, bei Verwendung des USARTs, als Baud Rate-Generator verwendet, so daß dann noch zwei weitere Timer zur Verfügung stehen.

USART

USART (universeller synchron asynchron Sender - Empfänger) dient zur seriellen Datenübertragung.

Die Baud-Rate, mit welcher der USART die Daten sendet oder empfängt, wird durch einen anliegenden Takt bestimmt, welcher in diesem Fall, wie schon erwähnt, von einem der 16 Bit Zähler erzeugt wird. Durch Steuerwörter kann der USART auf verschiedene Betriebsarten festgelegt werden (s.a.w.u.).

Zu jedem Datenwort, das in den USART geschrieben wird, fügt dieser automatisch Start, Stop und Paritybits hinzu.

Nach jedem empfangenen Zeichen ist es möglich festzustellen, ob bei der Datenübertragung ein Fehler erkannt wurde. Der USART besitzt hierfür ein besonderes Register, das gelesen werden kann. Hier ist es möglich, drei verschiedene Fehlerarten nach jedem Datenwort festzustellen.

Auch läßt sich die Länge des zu übertragenden Datenwortes zwischen 5 - 8 Bit wählen. Ebenfalls läßt sich in diesem Steuerwort feststellen, ob der USART ein Zeichen empfangen hat oder das eingeschriebene Zeichen schon gesendet wurde. Der USART übernimmt das kpl. Timing der seriellen Übertragung. Es muß ihm nur (wie in ein RAM unter einer bestimmten ADR') ein Datenwort eingeschrieben werden, welches er dann mit der gewählten Baud-Rate sendet. Oder es wird aus dem USART (wie aus einem RAM unter einer bestimmten ADR. gelesen) das empfangene Zeichen gelesen.

Der USART läßt eine Toleranz der Taktfrequenz (für die Baud-Rate) von einigen Prozent zu, so daß Abweichungen der Quarzfrequenz nicht die Übertragungssicherheit beeinflussen, wie es bei Übertragungen, die direkt über einen seriellen Ein - Ausgang der CPU erfolgen, leicht vorkommen kann (weil das Timing durch Software festgelegt ist).

Der USART ist ebenfalls in der Lage, zur gleichen Zeit zu senden und zu empfangen (voll duplex; bei Terminals oft verwendet).

Er läßt sich durch einfache Software an jeden Datenstrom anpassen.

TTY, RS 232

Um Daten in serieller Form über eine "weitere Strecke" zu übertragen und den Störpegel herabzusetzen, werden verschiedene "Treiberschaltungen" eingesetzt (TTA, RS 232, V 24 etc.).

Zwei dieser genannten Schnittstellen befinden sich auf dieser Karte.

Lötbrückendefinition:

Für normalen Betrieb alle dick gezeichneten Brücken einlöten.

B 1 Verbindung: TTY, RS 232 zum USART-Eingang
B 2 Interne/externe Erzeugung der 20mA Schleife
B 3 Externer Clock Timer 1 und 2
B 4 Verbindung: TTY, RS 232 zum USART-Ausgang

Aufbauhinweise

IC 2, 3, 11 sind entgegen den anderen einzusetzen.

Testhinweise

Für die Adreß-Decodierung beachten Sie die allgemeinen Testhinweise.

Test-Timer

Kontrollieren Sie, ob an den Anschlüssen 9, 15, 18 der Takt von 1,5 MHz am Timer anliegt und die Anschlüsse 11, 14 und 16 des gleichen ICs auf high liegen.

Stückliste

ASCII-Interface

1 Platine

1 Steckerleiste 64 pol.
1 Wire wrape Stützpunkt 26 pol.
1 8 Bit DIL Schalter

Integrierte Schaltkreise

IC 1 - 4	74 LS 245
IC 5, 6	74 LS 85
IC 7	74 LS 138
IC 10	74 LS 00
IC 9	74 LS 04

Widerstände

R 1 - 9	3,3K - 4,7K

Kondensatoren

C	22my 16V Tantal

Fassungen

4 DIL 20
3 DIL 16
3 DIL 14

Option Timer

Integrierte Schaltkreise

IC 8	74 LS 74
IC 12	8253

Widerstände

R 17, 18	1 K
R 10, 20	3,3 K - 4,7 K

Kondensatoren

C 1	56 p
C 2	10 n

Fassungen

1 DIL 24

Option USART

Integrierte Schaltkreise

IC 13	8251

Fassungen

1 DIL 28

1 Wire-Wrap Stützpunkt 26 pol.

Option TTY

Integrierte Schalkreise

IC 11	TIL 112

Transistoren

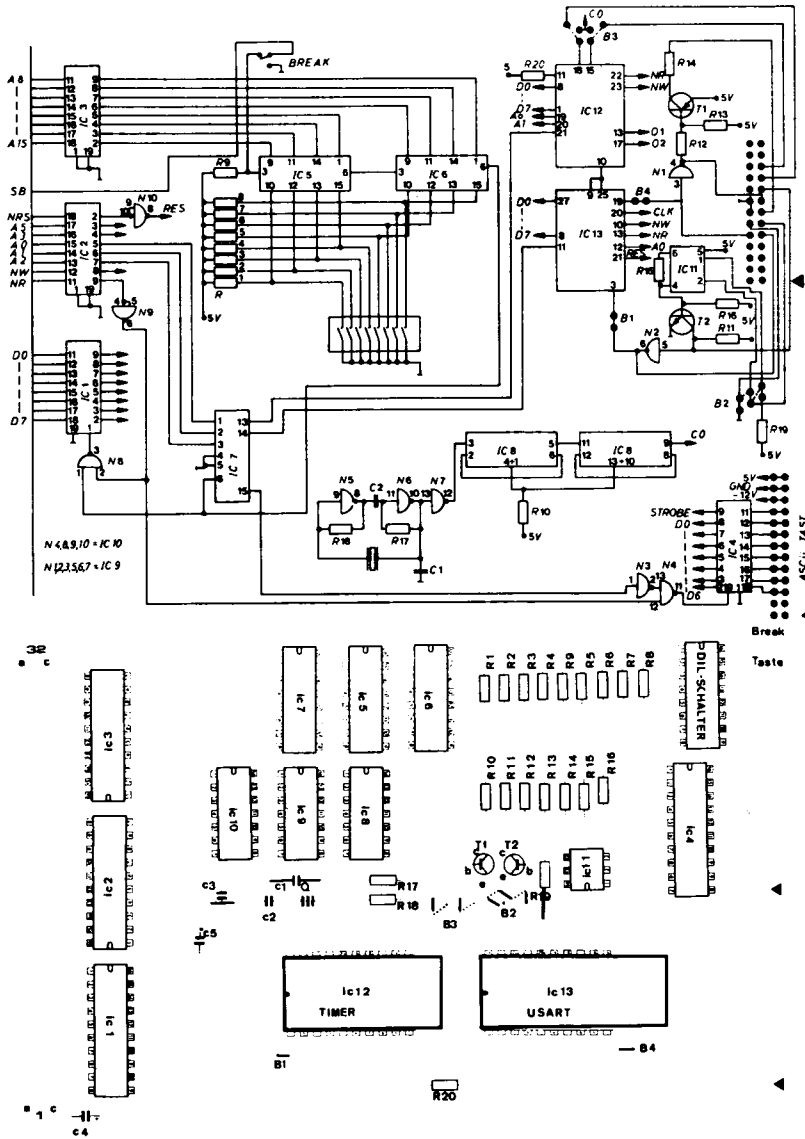
T 1	TUP
T 2	TUN

Widerstände

R 14, 19	150
R 12, 13	1 K
R 11	3,3 K - 4,7 K
R 15	100 K
R 16	10 K

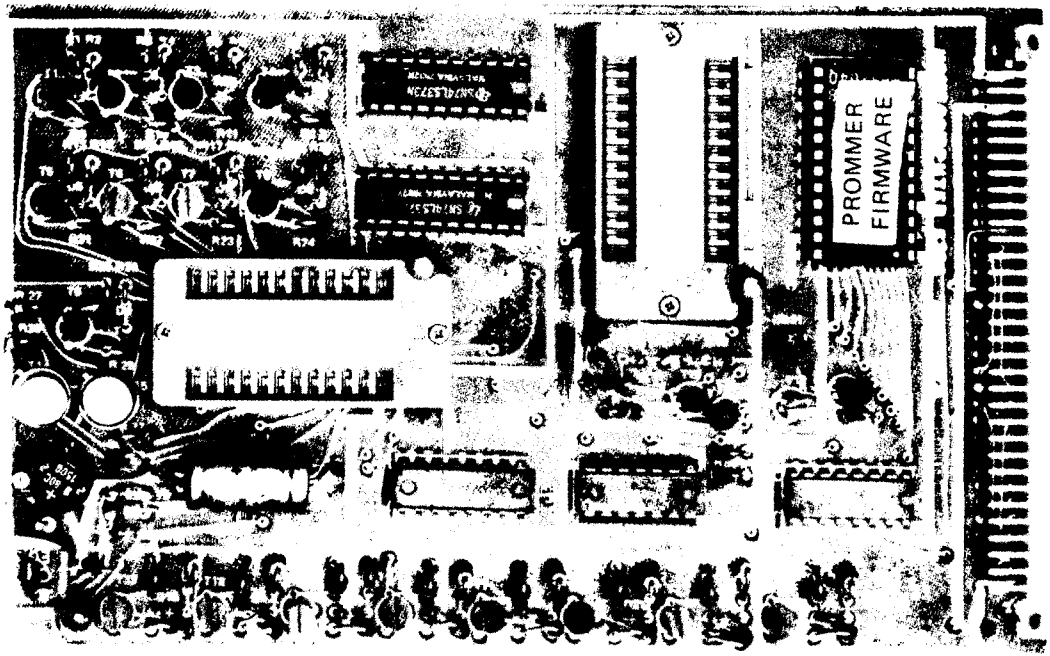
Adreßbelegung

ASCII-Interface	XX00	
Timer	XX0B	Kontrollwort
	XX0A	Date Timer 2
	XX09	Date Timer 1
	XX08	Date Timer 0
USART	XX05	Steuerwort
	XX04	Daten


BREAK
IC 12
IC 13
IC 11
IC 4
IC 5
IC 6
IC 7
IC 8
IC 9
IC 10
N1
N2
N3
N4
N5
N6
N7
N8
N9
N10
T1
T2
STROBE
GND
5V
Break
Taste
ASCII TAST.
DIL-SCHALTER
IC 12
TIMER
IC 13
USART

Homecomputer PROMMER für 5204

Bestückungsplan 6K - EPROM 5204



Damit der Anwender auch eigene Programme in EPROMs abspeichern kann, benötigt er ein Programmiergerät.

HOME-COMPUTER PROMMER für 5204

Der Prommer befindet sich auf einer durchkontaktierten Europakarte. Er ist versehen mit einer 64 Pol Steckerleiste und kann direkt auf den SC/MP Bus aufgesteckt werden. Die + 5 V und - 12 V Spannungen werden vom Bus entnommen, während die Programmierspannung von - 50 V von außen herangeführt wird.

IC 3 ist die Lesefassung und IC 5 ist die Programmierfassung. In der Fassung IC 4 befindet sich das Software-PROM. Alle Fassungen sind auf der Platine so angeordnet, daß auch eine größere Wechselfassung eingesetzt werden kann; es empfiehlt sich, bei häufiger Benutzung des Prommers solche Wechselfassungen mit Andruckhebel zu verwenden.

Gleichrichtung, Glättung und Stabilisierung der von außen herangeführten Programmierspannung erfolgen auf der Platine.

Auf der Karte sind 15 Adreßbits dekodiert, somit werden für den Prommer nur 3 Adreßbereiche, und zwar

7800 - 79FF für die Software
7A00 - 7BFF für die Lesefassung und
7C00 - 7DFF für die Programmierfassung

benötigt.

Stückliste

1 Platine
1 Steckerleiste 64 Pol

Integrierte Schaltkreise

IC 1 und 2	74 LS 373
(IC 3 Lesefassung)	
IC 4 Software	PROM 5204
(IC 5 Programmierfassung)	
IC 6	74 LS 75

IC 7	74 LS 00
IC 8	74 LS 138

Transistoren

T 1 - T 9, T 11, T 13, T 17 - 24	TUN
T 10, T 12	TUP
T 16	BD 135 o.ä.
T 14, T 15	2 N 1613 o.ä.

Zener Dioden

D 1 - 9, D 16 - 23	ZD 13
D 13, D 14	ZD 51
D 15	ZD 12

Dioden

D 10 - 12	DUS

Widerstände

R 1 - 8, 13 - 20, 25 - 28, 30 - 36, 41 - 56	4 K 7
R 9 - 12, 21 - 24, 29, 37, 38, 57 - 64	10 K
R 39 - R 40	8 K 2

Elektrolythkondensator

C 1 22 - 100 my / 100 V	(my)

Tantalkondensatoren

C 2	22 my 16 V

1 Gleichrichter
1 Trafo 220/50V

Fassungen

3 DIL 24
2 DIL 20
2 DIL 16
1 DIL 14

Die beiden Textool Wechselfassungen sind als Option erhältlich.

5204 Prommer

Erklärung

Um das EPROM 5204 zu programmieren, müssen an die Adressen und Daten 0V und - 12V (high und low) angelegt werden. Die eigentliche Programmierung erfolgt durch Anlegen von - 50V auf zwei Eingänge.

Der Programmiervorgang im einzelnen

In einem Latch werden die zu programmierenden Daten und die entsprechende Adresse festgehalten. Über Pegelumsetzer wird der TTL Pegel in 0V bzw. - 12V gewandelt. Flag 0 wird auf log. 1 gesetzt, einige mys später Flag 1 ebenfalls. Die Anschlüsse 4 und 23 der Programmierfassung liegen nun auf - 50V. Nach ca. 3ms werden die Flags wieder auf 0 gesetzt. Nach einigen ms wird dieser Vorgang wiederholt (Flags setzen), insgesamt 32 Mal pro Byte. Der gleiche Vorgang erfolgt mit den nächsten Daten und Adressen.

Die Adreßdecodierung des Prommers ist so gewählt, daß nur 3 Adreßbereiche benötigt werden (7800, 7A00, 7C00).

Alle 16 Bit des Adreßbus werden unter Zuhilfenahme eines 1 aus 8 Decoders (74 LS 138) und eines NAND-Gatters decodiert. Das Einschreiben der Daten und Adressen in die Latches erfolgt mit einem einfachen "Store" Befehl unter der Adresse 7C00. Bei der Adresse 7A00 kann das Master PROM gelesen werden.

Das Latchen der Daten und Adressen übernehmen zwei 74 LS 373 und ein 74 LS 75.

Aufbauhinweise

Alle Widerstände und Dioden senkrecht einlöten.
Bei T 16 weist die Metallseite nach innen.
Gleichrichter im Quadrat (oder, wenn eckiger Gleichrichter geliefert wurde, Anschlüsse in einer Reihe) einlöten.

Testhinweise

Das Software EPROM muß unter der Adresse 7800 ausgelesen werden können; ist dies nicht der Fall, beachten Sie die Testhinweise für Speicherkarten. Ebenso verfahren Sie mit den auf der Karte befindlichen Latches, diese müssen unter der Adresse 7A00 angesprochen werden können (nur mit Schreiben).

Programm bei 7800 starten und eine Blockprogrammierung auslösen. Anschließend kontrollieren, ob alle Adreß- und Datenbits der Programmierfassung zwischen 0V und - 12V wechseln (es müssen noch keine 50V anliegen). Ggf. entsprechenden Transistor-Schalter kontrollieren (vom Latch zur Programmierfassung).
Achtung! Bei den Adressen auf die Wertigkeit achten (Adreßbit 8 wechselt z.B. nur einmal während des gesamten Programmiervorganges). Am einfachsten läßt man das "ELBUG" durch die Blockprogrammierung programmieren (da hier auf jeden Fall Änderungen in allen Datenbits vorkommen).

50V Wechselspannung anlegen. Kontrollieren, ob an D 15 +12V und an C1 - 50V anliegen. Programm starten und feststellen, ob die beiden elek. Schalter für die - 50V (T 10 - T 15), die über Flag 0 und 1 gesteuert werden, schalten.

Programmier-Programm für 5204 EPROM

Das Programm meldet sich nach dem Start bei 7800 mit "PROM..". Die beiden Punkte weisen darauf hin, daß eine weitere Eingabe erwartet wird. Das Programmierprom enthält mehrere Routinen, die nun wahlweise aufgerufen werden können. Diese sind:

1. Löschungskontrolle - hiermit können Sie feststellen, ob ein PROM tatsächlich leer ist. Dies ist erforderlich, da sonst Fehler durch bereits programmierte Bits entstehen können.

Nach dem Einstecken in die Lesefassung drücken Sie Taste "E" auf der Hex I/O. Auf dem Display erscheint "EC" und "xxx Error", wobei die "xxx" die erste Adresse angeben, unter der Bits mit log. "1" zu finden sind. Nach dem Drücken einer beliebigen Taste erscheint wieder

"PROM" auf dem Display und eine neue Routine kann aufgerufen werden. Möchte man ein leeres PROM programmieren, so steckt man es in die Schreibfassung und schaltet die 48 V Versorgungsspannung ein, die zur Programmierung gebraucht werden. Diese Spannung (und die - 12 V) werden durch Flag 0 (bzw. Flag 1 ein - und ausgeschaltet. Es muß also darauf geachtet werden', daß im Augenblick des Programmierens keine andere Peripherie an diesen Flags angeschlossen ist.

Sie haben drei Möglichkeiten, ein PROM zu programmieren:
a. nach einem Master PROM
b. aus einem beliebigen RAM-Bereich
c. von 0C00 - 0DFF.
Hierzu enthält das Programmier-PROM 2 Routinen

2. Duplikat-Programmierung (Möglichkeit 1).
Man steckt das Master PROM in die Lesefassung und drückt Taste D. Es erscheint "PROM..DP" auf dem Display und das PROM wird mit dem Inhalt des Masters programmiert (Anlegen der 48 V nicht vergessen). Ist die Programmierung beendet, erscheint "Ende". Nach Drücken einer beliebigen Taste wieder PROM..

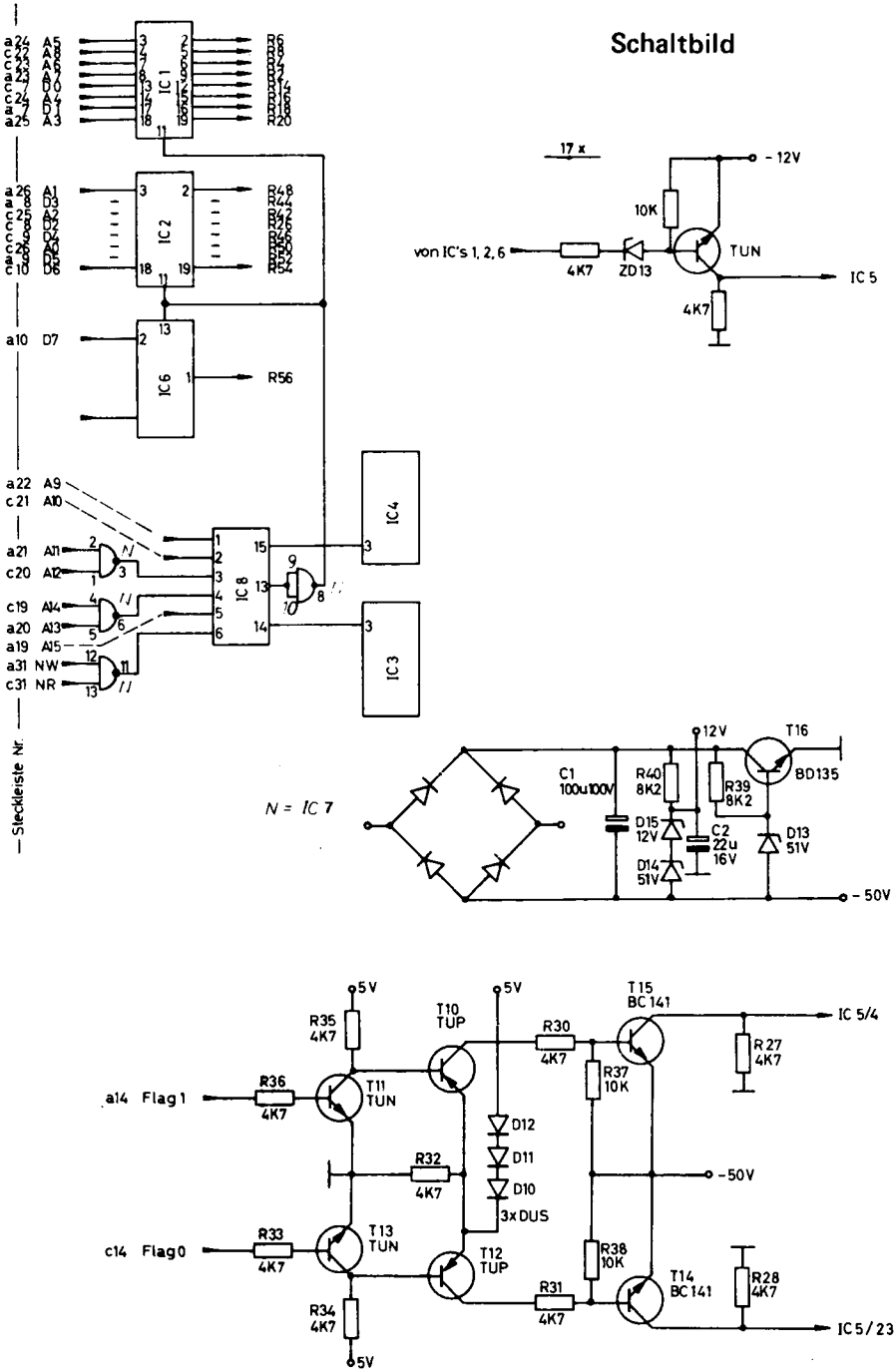
3. Bock-Programmierung (Möglichkeiten 2 und 3).
Man verfährt wie inter 2, drückt aber die Taste B. Es erscheint "Pr...B" auf dem Display. Möchte man nun den Inhalt des RAM-Bereichs 0C00 - 0DFF in das PROM einlesen, so drückt man nochmals die Taste B und es erscheint "Pr...BP". Das "P" zeigt an, daß die Programmierung läuft. Will man dagegen ein beliebiges anderes 1/2 K in das PROM kopieren, so drückt man statt der Taste "B" eine beliebige andere Taste und das Display zeigt "ad...b". Nun kann man die Anfangs- und Endadresse des zu kopierenden Blockes eingeben. Nach dem letzten Tastendruck erscheint auf dem Display wieder "BP". Auch hier läuft jetzt die Programmierrung. "Ende" zeigt den Abschluß des Programmiervorganges an.

Um das EPROM auf fehlerlose Programmierung zu überprüfen, dient die

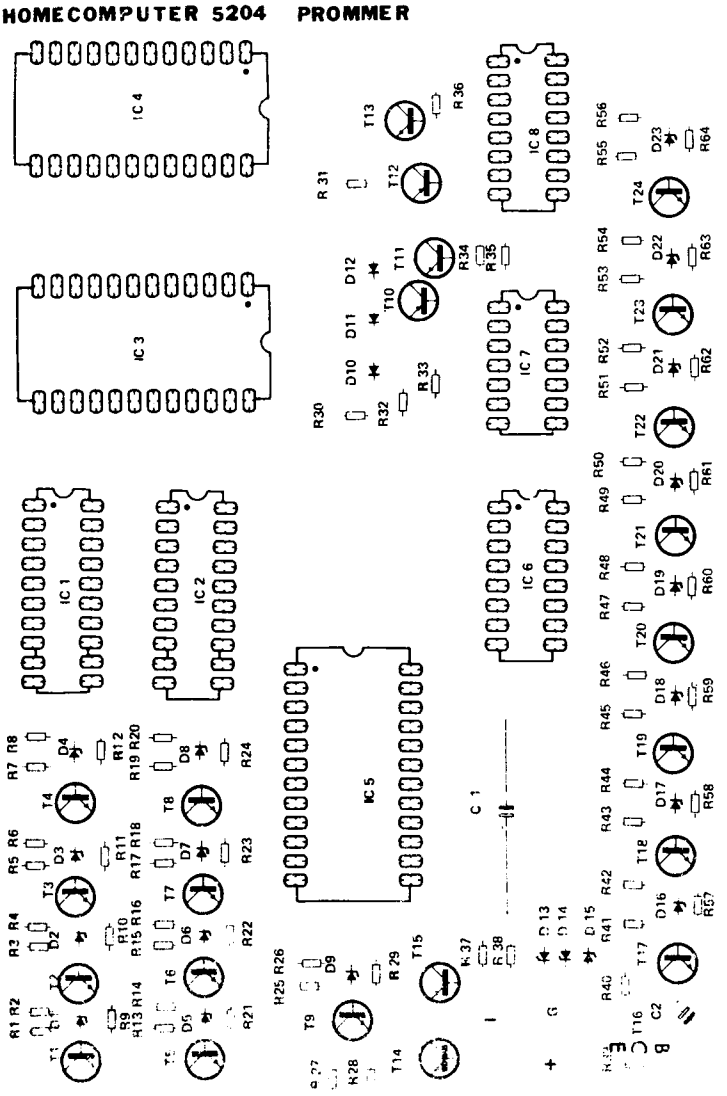
4. Compare-Routine
Das programmierte Prom wird in die Lesefassung gesteckt. Die Routine meldet sich nach Drücken der Taste "C" mit PROM co.
Will man den Bereich 0C00 - 0DFF vergleichen,
wird nochmals die "C" Taste gedrückt.

Beim Vergleich eines beliebigen anderen RAM-Bereichs mit dem PROM drückt man eine beliebige andere Taste (Ad.. co), folgend die Startadresse des gewünschten Blockes. Ist bei der Programmierung ein Fehler unterlaufen, so erscheint "xxx error", wobei xx die erste fehlerhafte Adresse innerhalb des PROMs angibt. Ergibt der Vergleich keine Abweichung, so erscheint "no error" und das PROM ist in Ordnung.

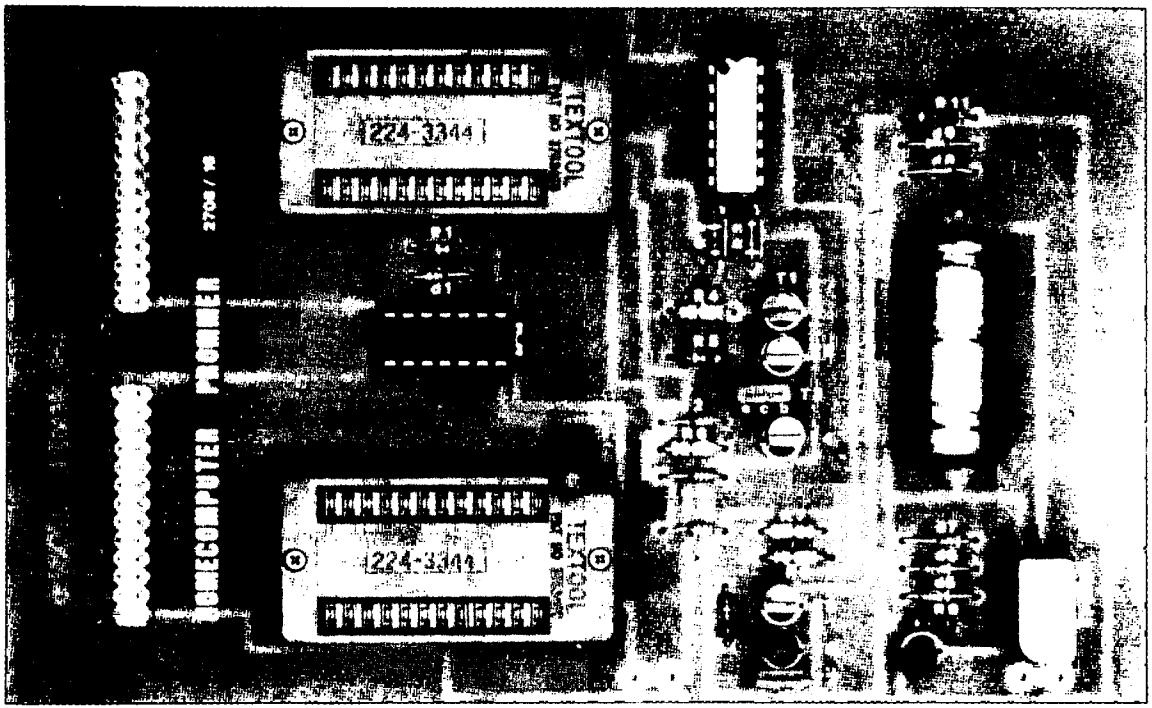
Schaltbild


17 x
- 12V
10K
von IC's 1, 2, 6
4K7
ZD13
TUN
IC 5
4K7
IC 1
IC 2
IC 6
IC 8
IC4
IC3
R56
N = IC 7
Steckleiste Nr.
C1
100u100V
12V
T16
BD135
R40
8K2
R39
8K2
D15
12V
D14
51V
C2
22u
16V
D13
51V
- 50V
5V
R35
4K7
T10
TUP
R30
4K7
T15
BC 141
IC 5/4
R27
4K7
R37
10K
a14 Flag 1
R36
4K7
T11
TUN
D12
D11
D10
3x DUS
R32
4K7
- 50V
c14 Flag 0
R33
4K7
T13
TUN
T12
TUP
R38
10K
T14
BC 141
R28
4K7
R31
4K7
R34
4K7
IC 5/23

Bestückungsplan



PROMMER 2708/16



Beschreibung 2708 PROMMER

Der Prommer befindet sich auf einer einseitig beschichteten Europakarte und ist mit zwei 26 pol. Wire-Wrap-Stützpunkten versehen. Zum Betrieb des Prommers ist eine 48 I/O line Karte erforderlich.
Für andere Systeme wie z.B. Z 80 oder 8080 kann die Verbindung sofort mit der PIO erfolgen.
Programmiert werden können Eproms 2708 oder 2716 mit 3 Spannungen Der Prommer wird durch zwei 26 polige Litzenleiter mit der 48 I/O line Karte verbunden. Dadurch kann die Hardware einfach gehalten werden. Es hat außerdem den Vorteil, daß nur 1/4 K auf dem Systembus belegt werden (ohne I/O line würden bis zu 4 K benötigt).
Durch Software wird der gesamte Programmier und Lesevorgang gesteuert.

Belegung und Benutzung für 26 Pol Verbindungen:

Port A	D0 - D7
Port B	A0 - A7
Port C0 - C2	A8 - A10
C4	Programmier Bit 1
C5	Programmier Bit 2
C6	Programmier Bit 3
C7	Programmier Bit 4

Weitere Erläuterungen siehe Hardwareerklärung.
Wird ein ähnlicher oder gleicher Portbaustein wie beim HC I/O Port an diese Anschlussbelegung des Steckers angepasst, kann dieser Prommer auch für jedes andere System benutzt werden.
Als Option ist für den SC/MP eine 1,5 K Software auf Cassette erhältlich. Näheres siehe unter Softwareerklärung.

Stückliste 2708/16 Prommer

1 Platine
2 Wire-Wrap Stützpunkte 26 pol.

Integrierte Schaltkreise

IC 1	74 LS 04
IC 2	74 LS 08

Tansistoren

T1,2,4 - 6,8	TUN
T3,7	BD 135

Zener-Dioden

Dy Zener-Dioden in Reihe (3 Stück) 26,4 V
Dx Zener-Dioden in Reihe (2 Stück) 12,9 V

Dioden

D 1 - D 4	DUS
1 Gleichrichter	

Widerstände

R 2 - 4, R 6 - 8	3K 3 - 4K 7
R 5, 10	100
R 9	1 K
R 9, 11	150 K

Kondensatoren

C	470 my/40V

Fassungen

2 DIL 24
2 DIL 14

Option
2 Wechselsockel
1 Trafo
Prommer Software

Erklärung

Sowohl 2708 als auch 2716 (3 Spannungen) werden in Schleifen programmiert. Das bedeutet, daß nacheinander alle Adressen angelegt (mit entsprechenden Daten). Dies geschieht so lange - ca. 50 - 100 mal - bis die Information in den EPROMs steht.

Programmierfassung lesen

Soll ein 2708 auf der Programmierfassung gelesen werden, müssen die Anschlüsse C4, C5, C6 und C7 auf 0 liegen. Adressen werden angelegt. Dann kann durch einfaches Lesen wie aus einem RAM der Inhalt abgerufen werden.
Beim 2716 muß nur C5 auf log. 1 liegen (Adress-Bit 10 durchschalten). Genauso in der Masterfassung. Adressbit 10 wird hier mit C7 durchgeschaltet. C4, C5, C6 sind hier nicht belegt. Dies ist nur zum Programmieren erforderlich.

Programmiervorgang für den 2708

Adressen und Daten werden über die 48 I/O lines angelegt. C6 wird auf 1 gesetzt (die Datenausgänge des EPROMs werden zu Dateneingängen) dann C7 für 1 ms (pro Adresse) auf 1 setzen. Diesen Vorgang für alle 1024 Adressen wiederholen. Um den 2716 zu programmieren, muß zusätzlich C4 und C5 auf 1 gesetzt werden. Sonst wird in gleicher Art programmiert.

Software

Die als Option erhältliche Software ermöglicht das Programmieren beider EPROMs. Sowohl vom Master- als auch aus dem 64 K Bereich kann dupliziert werden. Es ist auch möglich von einem 2716 auf der Masterfassung in einen 2708 in der Programmierfassung, oder umgekehrt, zu programmieren. Nach jedem Programmzyclus (1024 Adressen) erfolgt ein Vergleich, ob die Daten schon richtig im EPROM stehen. Ist dies der Fall, wird noch 25 mal nachprogrammiert.

Es steht weiter eine Compare und eine Erase-Routine zur Verfügung. Die Bedienung der Software entnehmen Sie bitte folgenden Ausführungen.

Bedienungsanleitung EPROM 2 2708/2716
(alle Ein- und Ausgaben über Hex I/O)

Start = X' 3000
Startmeldung: p r o m 27 = =

1. **Hauptabfrageschleife:**
In der Hauptabfrageschleife können die verschiedenen Unterprogramme aufgerufen werden. Außerdem muß der Typ des EPROMs in der Programmierfassung angegeben werden. Die erste Eingabe muß so lauten:

0 8 oder 1 6.

Bei einer ungültigen Eingabe wird die Abfrage wiederholt.

Jetzt hält der Prozessor mit der Meldung

s e t 2 7 0 8 oder s e t 2 7 1 6 an.

Nun wird das EPROM in die Programmierfassung gesetzt, und Sie können das gewünschte Unterprogramm wählen.

1.2. Unterprogramm Erase

Dieses Unterprogramm gibt Auskunft über den Löschzustand des EPROMs und wird erreicht durch Drücken der Taste "E".

1.3. Unterprogramm Compare

Dieses Unterprogramm dient zum Vergleichen des EPROM-Inhaltes und wird durch die Taste "C" aufgerufen.

1.4 Unterprogramm-Block-Programmierung

Nach dem Aufruf durch die Taste "B" kann ein ausgewählter Speicherbereich in das EPROM gebracht werden.

2. **Erase:**

Das Programm beginnt sofort mit dem Test. Das Programm endet mit der Meldung clear or not clear (gelöscht - nicht gelöscht).

3. **Compare Meldung c o m p . .**

Zuerst müssen Sie entscheiden, ob Sie den Inhalt des EPROMs in der Programmierfassung mit dem Inhalt des EPROMs in der Lesefassung

oder dem Inhalt eines Speicherbereiches vergleichen wollen.
Befehlstaste "R" = Speicherbereich
Befehlstaste "M" = Lesefassung

3.1.Speicherbereich: Meldung c r . . . . a d

Jetzt geben Sie die Startadresse des Speicherbereiches ein.

3.2.Lesefassung: Meldung c m 2 7 = =

Als erstes geben Sie den Typ des EPROMs in der Lesefassung an (08 oder 16). Das Programm beginnt sofort mit dem Vergleich, wenn sich in beiden Fassungen der gleiche Typ befindet. Ist dies nicht der Fall, müssen Sie angeben, welche Hälfte des EPROMs 2716 gemeint ist (1 = 1. Hälfte, 2=2. Hälfte)

3.3.Ergebnis:

Als Ergebnis erscheint entweder n o e r r o r oder e r r o r

4. Block-Programmierung: Meldung p r . . . . AO
Programmierung aus Speicherbereich.

Handbuch wie unter 3.1.

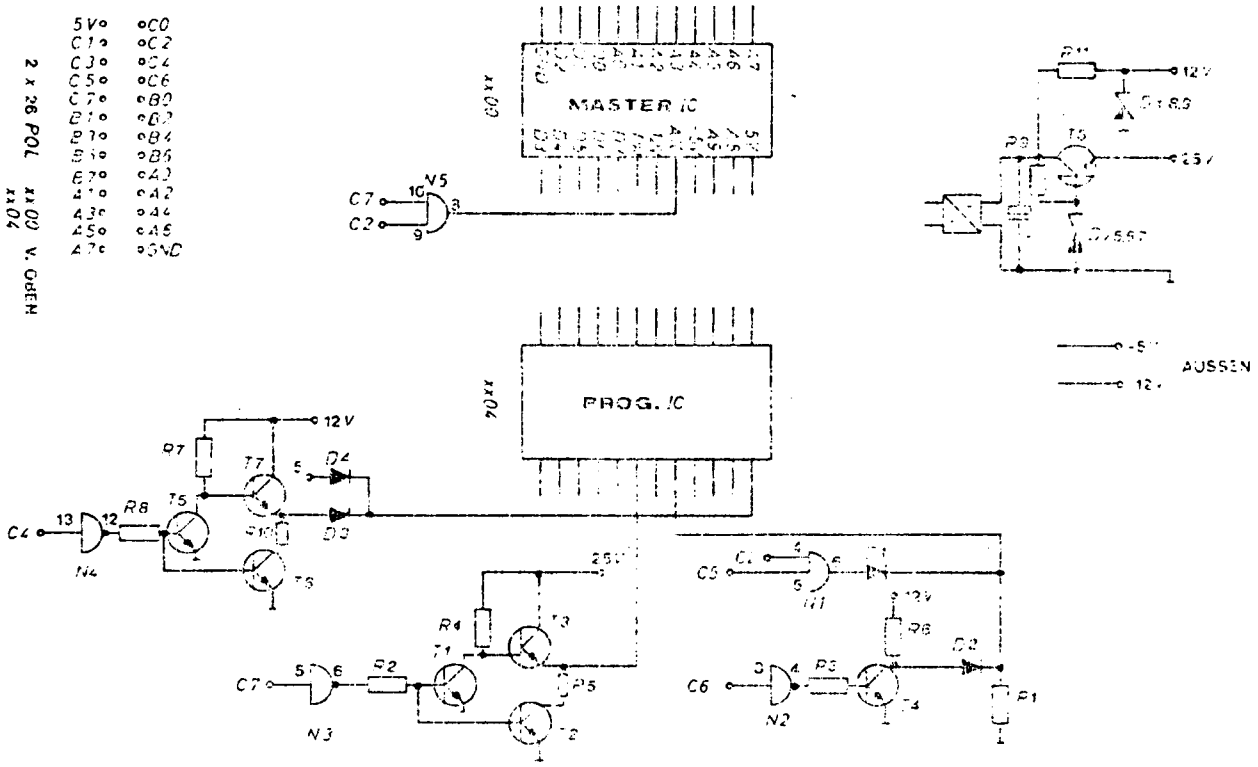
Beim Programmieren findet ein automatischer Vergleich statt, bei einem defekten EPROM bricht das Programm ab. Als Schlußmeldung erscheint entweder r e a d y oder e r r o r.

5. Duplizieren: Meldung d u p 2 7 = =

Handhabung wie unter 3.2.. Wie bei 4. findet ein automatischer Vergleich statt, und es werden dieselben Schlußmeldungen ausgegeben.

Aufbauhinweise
Beachten Sie die allgemeinen Aufbauhinweise

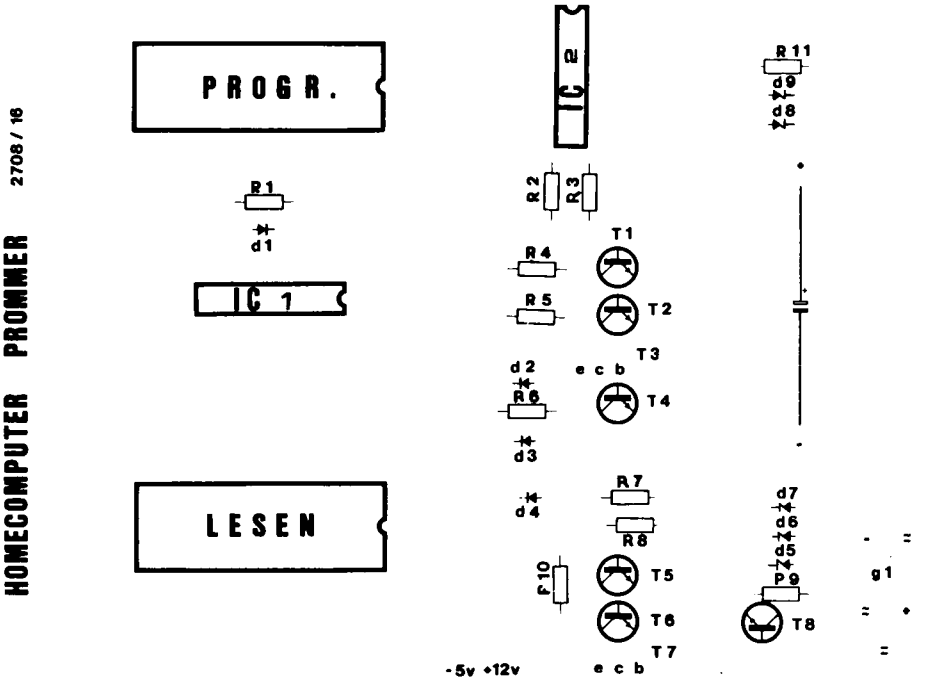
Testhinweise
Kontrollieren Sie die Belegungen der 26 pol Verbindungen und das Funktionieren der Transistor-Schaltung. Diesen Test machen Sie ohne das IC einzustecken. Die 25 Volt müssen angelegt sein.



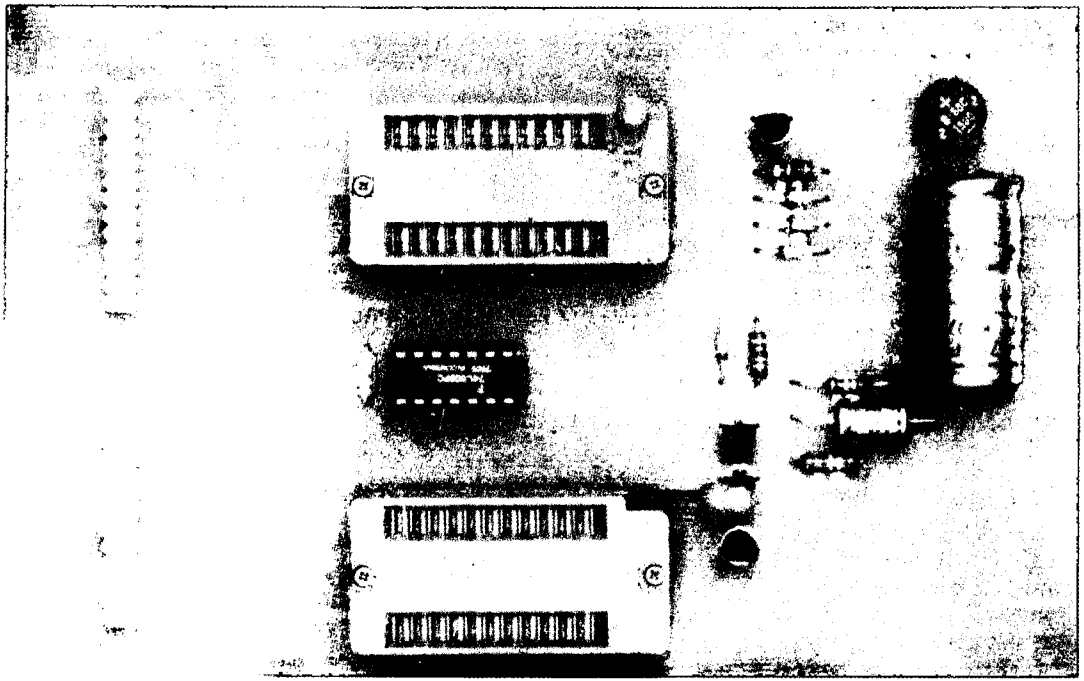
N1,5 = IC 1

N 2,3,4 = IC 2

Bestückungsplan



PROMMER 2758/16



Beschreibung

Der Prommer

Der Prommer befindet sich auf einer einseitig beschichteten Europakarte und ist mit zwei 26 poligen Wire-Wrap Stützpunkten versehen. Programmiert werden können die EPROM's 2758 und 2716 mit einer Spannung. Zum Betrieb des Prommers ist eine 48 I/O line Karte erforderlich. Der Prommer wird unter Zuhilfenahme zweier 26 poliger Litzenleiter mit dieser Karte verbunden. Dadurch kann die Hardware recht einfach gehalten werden und es hat den Vorteil, daß nur 1/4 K im Adreßbereich des CPU-Busses belegt wird. Wäre der Prommer direkt an den Systembus gelegt worden, wären (für 2716) 4K Byte erforderlich gewesen.

Durch Software wird der ganze Programmiervorgang und auch, wenn gewünscht, der Lesevorgang auf der Master- und Programmierfassung gesteuert.

Belegung und Benutzung der 26 Pol-Verbindung

PORT A	D0 - D7
PORT B	A0 - A7
PORT C0 - C2	A8 - A10
PORT C4	Pr.-Bit 1
PORT C5	Pr.-Bit 2
PORT C6	Adr.-Bit 10 Durchschalten
PORT C7	Pr.-Bit 3

Weitere Erläuterungen siehe unter Hardware Erklärung.

Wird ein gleicher (8255) oder ähnlicher I/O-Baustein an die Belegung des 26 poligen Steckers angepaßt, kann dieser Prommer auch für jedes andere System benutzt werden.

Als OPTION ist eine 2K SC/MP Software für diesen Prommer erhältlich. Näheres siehe unter Software Erklärung.

Stückliste 2758/16 (1 Spannung)

1 Platine
2 26 pol. Wire-Wrap Stützpunkte

Integrierte Schaltkreise

IC 1 74 LS 08
IC 2, 3 Programmierplätze

Transistoren

T 1 - T 4 TUN

Dioden

D 1 - D 3 drei Zener-Dioden in Reihe 26,4 V
D 4 DUS

Widerstände

R 1 1K
R 2 10 K
R 3 150 K
R 4, 5 3K3 - 4K7

Kondensatoren

C 1 470 my/40V
C 2 10 my

1 Gleichrichter

Optionen

1 Trafo 25 V
2 Wechselsockel
Programmier-Software

2 26 pol. konfek. Litzenleiter

Erklärung der Hardware

Sowohl 2758 als auch 2716 (mit einer Spannung) lassen sich recht einfach programmieren. Nämlich Daten und Adressen über entsprechende I/O line Belegung anlegen, 5V an PIN 20 schalten, PORT C4 auf 1,25V an PIN 21 schalten, PORT C7 auf 1 und einen TTL Takt von 50ms Länge für jedes Byte an Anschluß 18 des EPROM's. (Takten von Port C5 25 V Wechselspannung anlegen, die Stabilisierung ist auf der Karte.

Beim 2716 muß noch das Adreßbit 10 mit angelegt werden. Über PORT C2 läßt sich diese Adresse durchschalten oder sperren.

Beide EPROMs können sowohl in der Masterfassung als auch in der Programmierfassung betrieben werden. Ein gemischter Betrieb, z.B. Masterfassung 2758 und Programmierfassung 2716, ist möglich.

Lesen auf der Programmierfassung

Soll ein 2758 auf der Programmierfassung gelesen werden, müssen die Anschlüsse C2, C4, C5 und C7 auf log. 0 liegen. Adressen werden über die entsprechenden Anschlüsse (siehe weiter oben) angelegt. Nun kann der Inhalt des EPROMs unter dieser Adresse gelesen werden. Wird ein 2716 gelesen, muß C2 zusätzlich auf log. 1 geschaltet werden. Hiermit wird, wie schon vorher erwähnt, Adreßbit 10 durchgeschaltet. Ebenso kann auf der Masterfassung gelesen werden, nur sind hier die Anschlüsse C4, C5, C7 nicht belegt (diese sind nur zur Programmierung erforderlich).

Programmiervorgang

Adresse über genannten Port anlegen, höherwertige Adresse mit C4, C7 log 1 anlegen (EPROM schaltet die Datenausgänge in Dateneingänge um). Nun Daten anlegen und durch einen 50 ms TAkt dieses Byte in die EPROMs schießen. Diesen Vorgang mit den anderen Adressen wiederholen.

Der Programmiervorgang für den 2758 dauert ca. 55s.

Für den 2716 muß nur zusätzlich C2 auf log 1 gesetzt werden (Adr. 10). Die Masterfassung ist für den Fall vorgesehen, daß ein Master-EPROM vorhanden ist, welches kopiert werden soll.

Software

Die als OPTION erhältliche Software ermöglicht das Programmieren beider EPROMS. Sowohl vom Master als auch aus den 64K des Systems kann ein beliebig langer Block in das gewählte EPROM kopiert werden.

Es ist möglich, von einem 2716 in der Masterfassung in einen 2758 in der Programmierfassung einen beliebig langen Block zu kopieren, ebenso umgekehrt. Es ist möglich, teilweise zu programmieren (z.B. nur 1 Byte) an beliebiger Stelle im EPROM.

Nach jedem Programmiervorgang erfolgt die Kontrolle, ob das Byte richtig im EPROM steht.

Weiter stehen zur Verfügung eine Compare- und eine Erase-Routine.

Die Programmierung der Software entnehmen Sie bitte den folgenden Ausführungen.

Bedienungsanleitung EPROM 3 2758/2716
(alle Ein- und Ausgaben über Hex I/O)

Start = X' 3000
Startmeldung: p r o m 27 = =

1. Hauptabfageschleife:

In der Hauptabfrageschleife können die verschiedenen Unterprogramme aufgerufen werden. Außerdem muß der Typ des EPROMs in der Programmierfassung angegeben werden. Die erste Eingabe muß also lauten:

5 8 oder 1 6.

Bei einer ungültigen Eingabe wird die Abfrage wiederholt.

Jetzt hält der Prozessor mit der Meldung

s e t 2 7 5 8 oder s e t 2 7 1 6 an.

Nun wird das EPROM in die Programmierfassung gesetzt, und Sie können das gewünschte Unterprogramm wählen.

1.2.Unterprogramm Erase

Dieses Unterprogramm gibt Auskunft über den Löschzustand des EPROMs und wird erreicht durch Drücken der Taste "E".

1.3.Unterprogramm Compare

Dieses Unterprogramm dient zum Vergleichen des EPROM-Inhaltes und wird durch die Taste "C" aufgerufen.

1.4.Unterprogramm-Block-Programmierung

Nach dem Aufruf durch die Taste "B" kann ein ausgewählter Speicherbereich in das EPROM gebracht werden.

2. Erase: Meldung e r a s e . .

Wollen Sie das ganze EPROM auf seinen Löschzustand untersuchen, drücken Sie zuerst die Befehlstaste "UP", sonst eine beliebige andere Taste. Im ersten Fall beginnt das Programm sofort mit dem Test, im zweiten wird die Eingabe der Start, und danach die Stop-adresse erwartet. Das Programm endet mit der Meldung clear oder not clear (gelöscht - nicht gelöscht).

3. Compare Meldung c o m p . .

Zuerst müssen Sie entscheiden, ob Sie den Inhalt des EPROMs in der Programmierfassung mit dem Inhalt des EPROMs in der Lesefassung oder dem Inhalt eines Speicherbereiches vergleichen wollen.
Befehlstaste "R" = Speicherbereich
Befehlstaste "M" = Lesefassung

3.1.Speicherbereich: Meldung c r . . . . a d

Jetzt geben Sie die Startadresse des Speicherbereiches ein. Wollen Sie mit dem ganzen Inhalt des EPROMs vergleichen,drücken Sie die Befehlstaste "UP", und es erscheint r u n r u n auf der Anzeige. Im anderen Fall (Drücken einer beliebigen anderen Taste) erscheint

c r . . . . a d und Sie müssen Start und Stopadresse des gewünschten Bereiches im EPROM eingeben, und es erscheint ebenfalls r u n r u n

3.2. Lesefassung: Meldung c m 27 = =

Als erstes geben Sie den Typ des EPROMs in der Lesefassung an (58 oder 16). Drücken Sie nun die Taste "UP", beginnt das Programm mit dem Vergleich des ganzen Inhaltes, wenn sich in beiden Fassungen der gleiche Typ befindet. Ist dies nicht der Fall, müssen Sie angeben, welche Hälfte des EPROMs 2716 gemeint ist (1 = 1. Hälfte, 2 = 2. Hälfte). Falls Sie jedoch anstatt der Taste "UP" eine andere gedrückt haben, meldet sich das Programm mit c m . . . p a und erwartet Start- und Stopadresse des EPROMs in der Programmierfassung, und nach der Meldung c m . . . . l a die gewünschte Startadresse des IC's in der Lesefassung.

3.3. Ergebnis:

Als Ergebnis erscheint entweder n o e r r o r oder e r r o r x x x wobei xxx die Adresse des Fehlers in der Programmierfassung angibt.

4. Block-Programmierung: Meldung p r . . . . r a
Programmierung aus Speicherbereich.

Handhabung wie unter 3.1., jedoch mit der "down" Taste anstatt der "UP" Taste.

Beim Programmieren findet ein automatischer Vergleich statt, bei einem defekten EPROM bricht das Programm ab. Als Schlußmeldung erscheint entweder r e a d y oder e r r o r x x x.

5. Duplizieren: Meldung d u p 2 7 = =

Handhabung wie unter 3.2 jedoch mit der "down" Taste anstatt der "UP" Taste. Wie bei 4. findet ein automatischer Vergleich statt, und es werden dieselben Schlußmeldungen ausgegeben.

Aufbauhinweise

Beachten Sie die allgemeinen Aufbauhinweise

Testhinweise

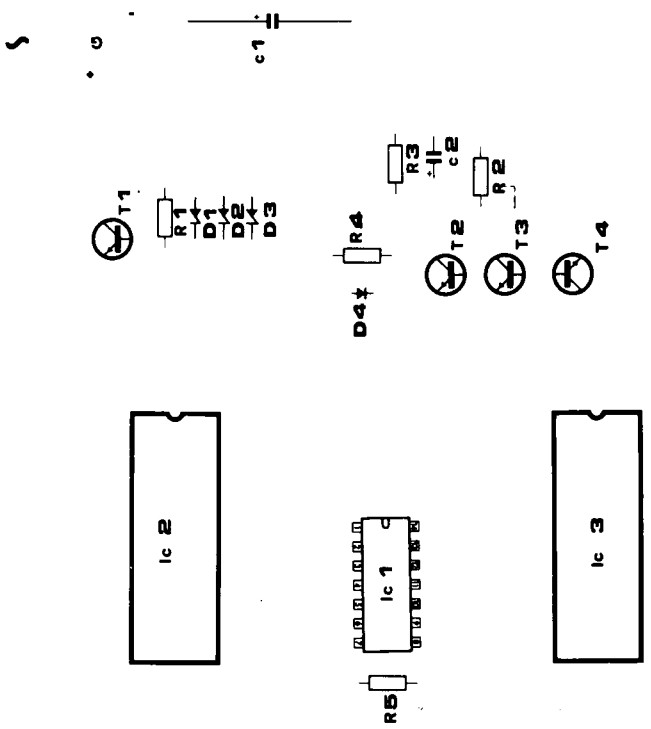
Kontrollieren Sie die Belegungen der 26pol Verbindungen und das Funktionieren der Transistor-Schaltung.
Diesen Test machen Sie ohne das IC einzustecken. Die 25 Volt müssen angelegt sein.

Um den SC/MP beim Lösen mathematischer Aufgaben noch leistungsfähiger zu machen, kann man sich einen sog. Number Crunchers bedienen. Der Hersteller des SC/MP bietet ein passendes IC mit der Bezeichnung MM 57109 an.

Schaltbild

25V
L1
1
MASTER IC 3
A7
A6
A5
A4
A3
A2
A1
A0
D0
D1
D2
GND
5V
A8
A9
Vpp
CS
A10
PD
D7
D6
D5
D4
D3
xx00
N
8
10
9
C2
C6
PROG. IC 2
C4
C5
xx04
N
3
1
2
C2
C6
25V
T4
R2
T3
R4
5V
T2
R5
C7
C2
R3
D4
5V
C0
C1
C2
C3
C4
C5
C6
C7
B0
B1
B2
B3
B4
B5
B6
B7
A0
A1
A2
A3
A4
A5
A6
A7
GND
2 x 26 POL
xx00
xx04
V. OBEN
N= IC 1

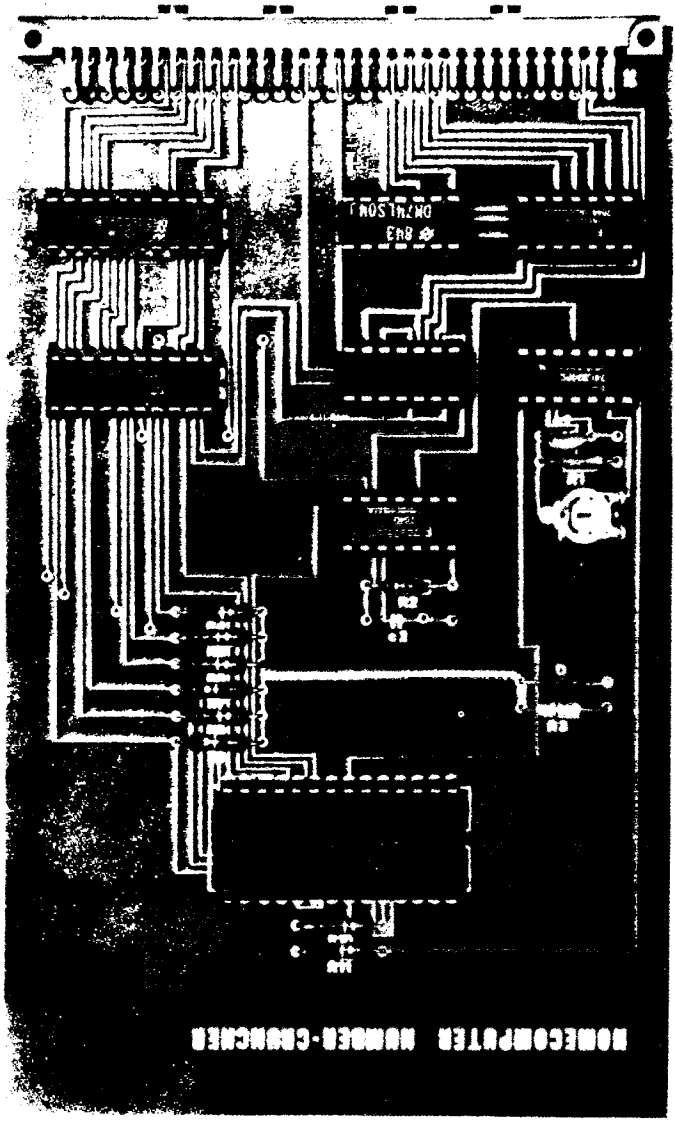
Bestückungsplan



HOMECOMPUTER PROMMER 2758 / 16

XX04 XX00

Number Chruncher



Um den SC/MP beim Lösen mathematischer Aufgaben noch leistungsfähiger zu machen, kann man sich einen sog. Number Crunchers bedienen. Der Hersteller des SC/MP bietet ein passendes IC mit der Bezeichnung MM 57109 an.

Number Chruncher

Beschreibung

Das Number-Chruncher-Interface befindet sich auf einer doppelseitig beschichteten, durchkontaktierten Karte im Europa-Format und ist mit einer 64 pol. Steckerleiste nach DIN 41612 versehen. Der Number-Chruncher (arithmetischer Prozessor) ist ein speziell für arithmetische Operationen ausgelegter Chip. Er übernimmt, nachdem ihm Daten und die Operation in einem Codewort mitgeteilt worden sind, die Verarbeitung der Daten nach der gewählten mathematischen Funktion. Das Ergebnis kann dann von der CPU gelesen werden. Folgende arithmetische Funktionen beherrrscht der Number-Cruncher: Er arbeitet grundsätzlich in umgekehrter Polnscher Notation.

Allgemeine Operationen:

Zahlen 0 - 9	
Dezimalpunkt	im Zahlenbereich
Exponent	$-9{,}9999999 \times 10^{-99}$
Vorzeichen	$+9{,}9999999 \times 10^{+99}$
Konstante phi	

Fließkomma oder Eponentialdarstellung; bei Überschreitung der 8 Stellen Umschaltung auf Exponentialdarstellung.

Funktionen:	Enter, NOP, Halt, Roll (interne Register XYZT routieren lassen); Austausch - Speicherbefehle für die internen Register; interner Stack vorhanden.

Mathematische Operationen:

+, -, x, :, y^x, Speicher +, Speicher -, Speicher x, Speicher :, 1/x, x, x^2,

10^x, e^x, lnx, 10gx, sin x, cos x, tan x, arcsin x, arccos x, arctan x, rad/grad-, grad/rad-Umschaltung sowie einige weitere Befehle für interne Operationen. Weitere Ausführungen und wie Sie den Number Cruncher mit eigener Software benutzen können, siehe gesonderte Beschreibung. Zum Lieferumfang gehört eine Cassette mit einem Programm, mit welchem alle oben genannten Funktionen über die Hex-Tastatur des SC/MP's abgerufen werden können. Der SC/MP kann dann wie ein Taschenrechner benutzt werden (dies soll natürlich nicht sein einziger Zweck sein).

Stückliste Number Cruncher

1 Platine
1 Steckerleiste

Integrierte Schaltkreise 64 Pol

IC 1	74 LS 30
IC 2, 4	74 LS 04
IC 3	74 LS 373
IC 5	74 LS 02
IC 6	74 LS 241
IC 7	74 121
IC 8	MM 57109 Number Cruncher

Zener-Dioden

D	3,3 V - 3,9 V

Widerstände

R 1	1,8 k
R 2	8,2 k
R 3	330
R 4 - R 11	10 K
1 Trimmpoti	5 k

Kondensatoren

C 1	470 p
C 2	15 n

Fassungen

1 DIL 28
2 DIL 20
5 DIL 14

Erklärung

Das Interface liegt im Adreßbereich 8F00 - 8FFF, decodiert durch einen 8 Bit-Nand (74 LS 30). Mit dem NWDS werden in ein Latch die Eingangsdaten zwischengespeichert. Mit dem NRDS werden diese über einen Leitungstreiber (74 LS 241) auf den System-Bus geschaltet. Der auf der Karte befindliche Oszillator muß mit dem Poti auf 400 kHz abgeglichen werden. Dieser erzeugt das Taktsignal für den Number-Chruncher. Ist der Number-Chruncher bereit, eine 6 Bit-Information zu empfangen, geht der RDY-Ausgang auf high. Der Error-Ausgang geht auf high, wenn eine nicht definierte Operation ausgeführt werden soll.

Das DAS-Signal geht auf low, sobald der Number-Chruncher Ergebnisse ausgibt. Da dieses Signal sehr kurz ist, wird es über einen Monoflop (74 121) verlängert. Alle drei Signale werden über den Leitungstreiber 74 LS 241 auf den Datenbus gelegt, wo diese vom SC/MP gelesen werden können. Mit Flag 1, der an den Hold-Eingang des Number-Chrunchers geht, kann dieser angehalten werden. Die - 4 V, die der Number-Chruncher benötigt, werden aus den auf dem Bus befindlichen - 12 V durch eine 3,3 V Zener-Diode und einen Widerstand gewonnen.

Übersicht über die HC - NCU Befehle

Mnemonic	HEX	Beschreibung
0	00	Die Zahl wird ins X-Register gebracht. Für
1	01	die übrigen Register gilt:
2	02	
3	03	
4	04	X ⇄ Y
5	05	Y ⇄ Z
6	06	Z ⇄ T
7	07	
8	08	
9	09	

DP	0A	Dezimalpunkt. Die danach eingegebenen Werte sind Zahlen nach dem Komma.
EE	0B	Enter Exponent. Die danch eingegebenen Werte (max. 2) bilden einen Exponenten zur Basis 10.
CS	0C	Vorzeichenwechsel. Abhängig vom Eingabemodus wird das Vorzeichen der Mantisse oder das Exponenten gewechselt.
PI	0D	Die Konstante π wird ins X-Register gebracht. Die übrigen Register verhalten sich wie bei Eingabe einer Zahl.
EN	21	Enter. Schließt die Eingabe ab und verschiebt die Register: X ⇦ Y Y ⇦ Z Z ⇦ T
ROLL	23	verschiebt die Register Y ⇦ X Z ⇦ Y T ⇦ Z X ⇦ T
POP	2E	verschiebt die Register Y ⇦ X Z ⇦ Y T ⇦ Z 0 ⇦ T
XEY	30	tauscht X und X Register aus X ⇨⇦ Y

Benutzungsbeispiel:

2 x (3 + 17) - 34,2 = ?

Tastenfolgen

a) 3, EN, 1,7↓, EN, 2, ▽, 2, EN, 3, 4, DP, 2, ▽, 1,
Anzeige 5. 8000 00

Zweite Möglichkeit

b) 2, EN, e, EN, 1, 7, ▽, 0, ▽, 2, EN, 3, 4, DP, 2, ▽, 1

Anzeige 5.8000 00

Es gibt sicherlich noch weitere Möglichkeiten, derartige Gleichungen zu berechnen.

Wie Sie den Number-Cruncher auch für eigene Programme verwenden können, entnehmen Sie bitte den folgenden Flußdiagrammen:

Im Ausgabe-Modus haben die 8 Bit folgende Bedeutung:

Bit 7	ERROR	(geht auf "0", bevor eine BCD Zahl angegeben wird)
Bit 6		ohne Bedeutung
Bit 5	ERROR	(geht auf "1" bei unzulässigen Operationen, z.B. Division durch 0)
Bit 4	RDY	(zeigt Bereitschaft des NC an)
Bit 3 - 0		BCD Zahl

MCLR	2F	Löscht alle Register und Flags
ECLR	2B	Löscht ERROR FLAG
OUT	16	Befehl zur Ausgabe des C-Registers
TOGM	22	Betriebsartwechsel Fließkomma ⟷ Scientific Notation
SMDC	18,0X	Anzahl der Stellen der Mantisse wird festgelegt

Bedienungsanleitung des NCU-Programm 1

Nach Eingabe und Start des Programms bei # 0C00 besteht die Möglichkeit, auf der HEX I/O wie auf einem Taschenrechner zu rechnen. Benutzt werden die Taste 0 - F und "UP" und "Down". Die "UP" und "Down" Tasten bilden Präfix-Tasten, durch die jede der Tasten 0 - F drei Funktionen erhält. Der Rechner arbeitet in der umgekehrt Polnischen Notation.

Übersicht über die Tastenbelegung:

POP +/- arc SIN	X M DP arc COS	MR E E arc tan	DTR I I MANT
ROLL 8 SIN	X Y 9 COS	MS E N TAN	RTD CE/CLR SN/FK
x^2 4 M+	e^x 5 M-	10^x 6 M*	$1/x$ 7 M/
x 0 +	ln 1 -	log 2 *	yx 3 /

XEM	1B	Tauscht X-Register und Speicher aus: X ⇄ M
MS	1C	Bringt X-Register in den Speicher X → M
MR	1D	Bringt Speicher ins X-Register X ← M
LSH	1E	Verschiebt die Mantisse um eine Stelle nach links, der Dezimalpunkt bleibt unverändert.
RSH	1F	wie LSH, jedoch verschieben nach rechts.
+	39	X und Y - Register werden verknüpft. Das Ergebnis steht im X-Register. Für die übrigen Register gilt:
-	3A	
*	3B	Z Y T Z
	3C	0 T
$**(Y_x$		
$**(Y^x)$	38	

INV+	20,39	Speicher und X-Register werden ver-
INV -	20,3A	knüpft. Die übrigen Register bleiben unbe-
INV*	20,3B	rührt.
INV/	20,3C	
1/X	37	Mathematische Operationen mit dem
SQRT	34	X-Register
SQ	33	
10X	32	
EX	31	
LN	35	
LOG	36	
SIN	24	
COS	25	
TAN	26	
INV SIN	20,24	
INV COS	20,25	
INV TAN	20,26	
OTR	2D	
RTD	2C	

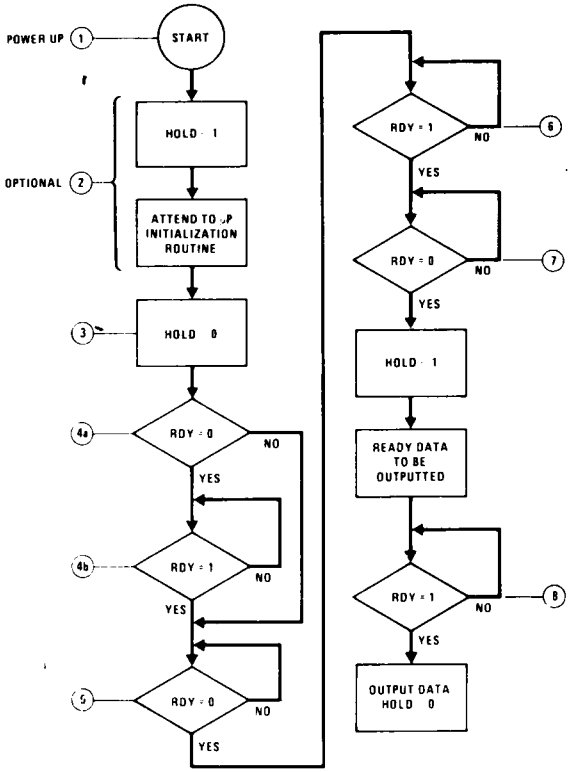
Aufbauhinweise

Mit einem Frequenzmesser an Punkt 7 des Number-Crunchers eine Frequenz von 400 kHz mit dem Poti einstellen.

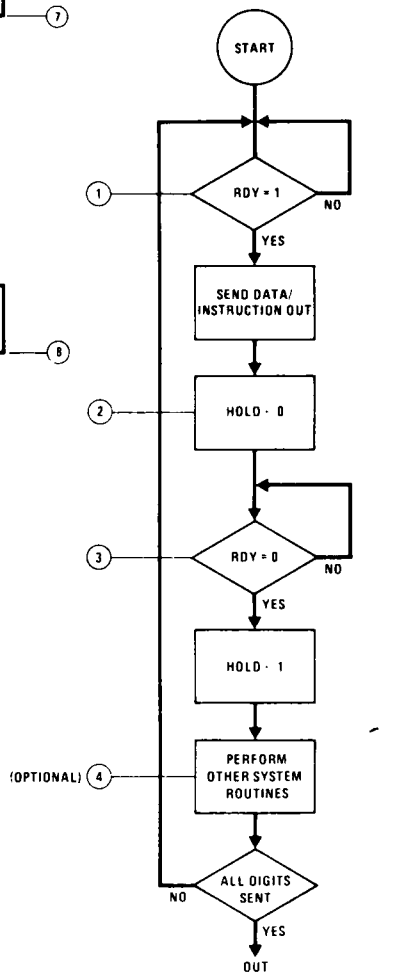
Testhinweise

Überprüfen Sie die 400 kHz am Anschluß 7 des Number-Crunchers. Unter Zuhilfenahme der Chip-Select-Programme (Testhinweise für Speicherkarten) überprüfen Sie das Freigeben des ICs 3 beim Schreiben (Pin 11) und des ICs 6 beim Lesen (Pin 1, 19).

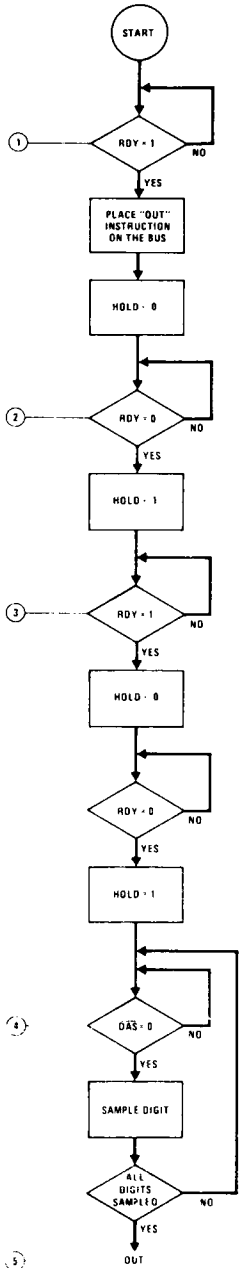
Flußdiagramme für eigene Number Cruncher Software



NCU Initialization Flow Diagram

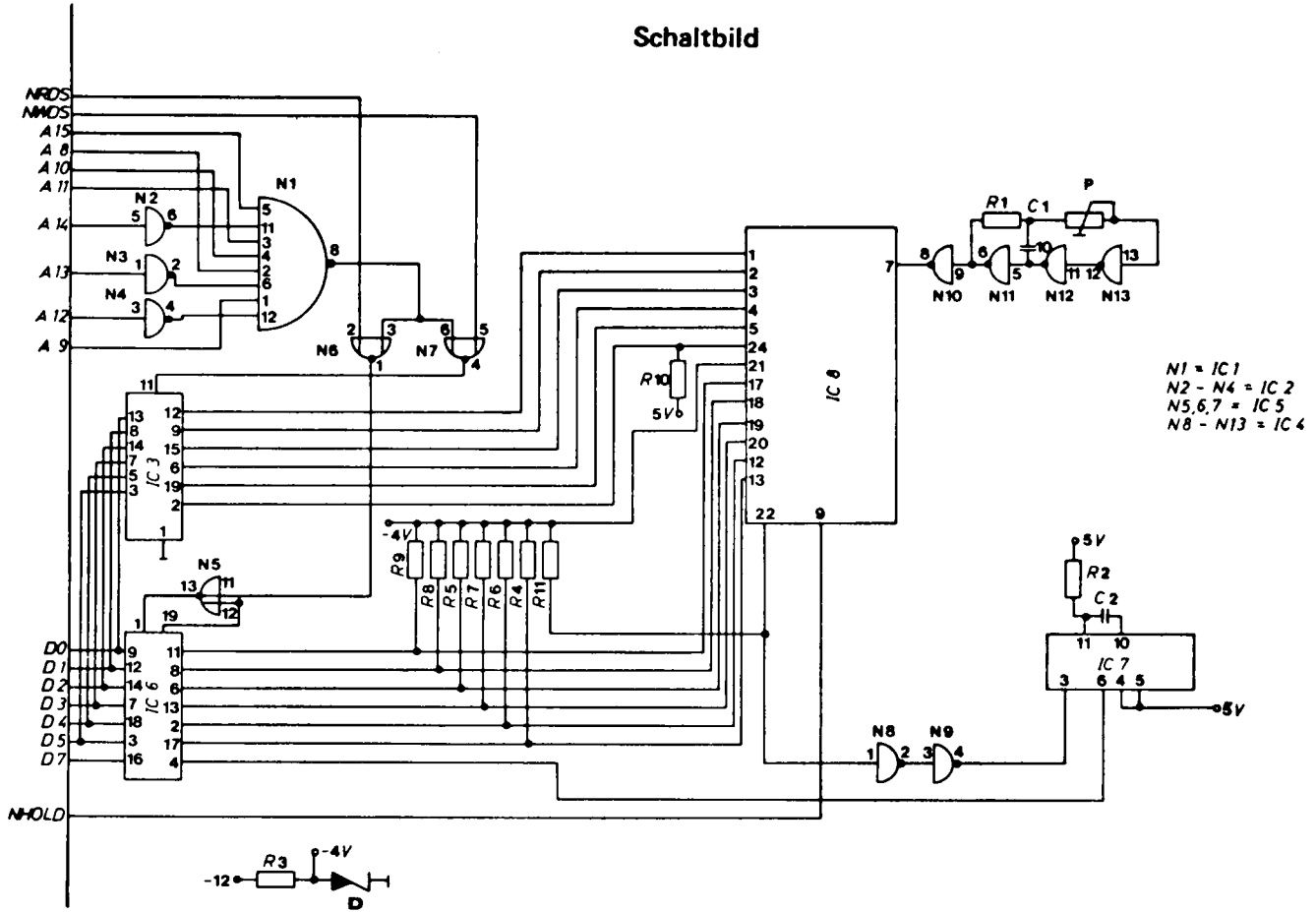


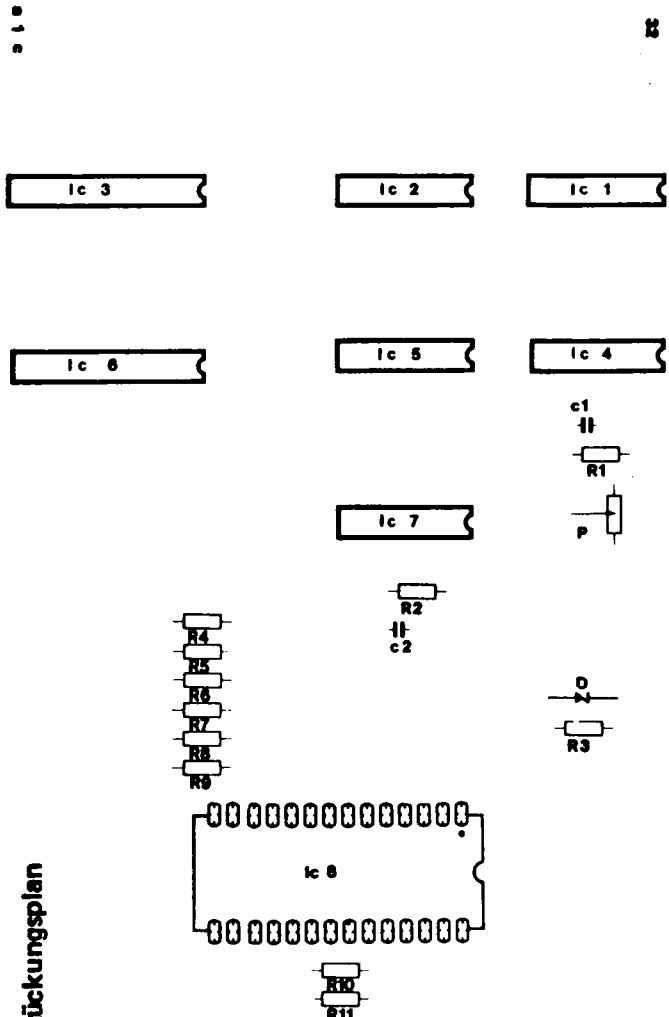
NCU Digit and Instruction Input Flow Diagram



NCU Digit Output Flow Diagram

Schaltbild


NRDS
NMIOS
A 15
A 8
A 10
A 11
A 14
A 13
A 12
A 9
N1
N2
N3
N4
N5
N6
N7
N8
N9
N10
N11
N12
N13
IC 3
IC 6
IC 8
IC 7
R1
C1
P
R2
C2
R3
R4
R5
R6
R7
R8
R9
R10
R11
-4V
5V
-12
D
D0
D1
D2
D3
D4
D5
D7
NHOLD
N1 = IC 1
N2 - N4 = IC 2
N5,6,7 = IC 5
N8 - N13 = IC 4


Ic 3
Ic 2
Ic 1
Ic 6
Ic 5
Ic 4
c1
R1
P
Ic 7
R2
c2
R4
R5
R6
R7
R8
R9
D
R3
Ic 8
R10
R11
Bestückungsplan
HOMECOMPUTER NUMBER-CRUNCHER

Intelligentes Floppy-Interface

Das Interface befindet sich auf einer doppelseitig beschichteten und durchkontaktierten Karte.
Sie kann direkt unter einem Minifloppy Drive geschraubt werden und bildet so eine Einheit mit dieser.

Das Interface ist mit dem Controller 1791 aufgebaut und wird durch Einsetzen einer Z 80 CPU und einer parallelen Schnittstelle universell verwendbar.

Der Controller arbeitet in Double Density nach dem IBM-Format (pro Sektor 256 Byte).
Es eignet sich für Mini-, Standard- und auch Doppelkopflaufwerke. Bei Minilaufwerken können bis zu 3 Stück angeschlossen werden. Durch das Double Density Verfahren können auf einer Minidiskette 160K Byte abgespeichert werden. Beim Standardlaufwerk sogar 600K Byte.

In einem 2708/58, welches auf der Interface-Karte untergebracht ist, befindet sich das Urladeprogramm für das DOS, welches auf einer Diskette zur Verfügung steht. Dieses DOS wird in die 8K Byte RAM des Floppy-System geladen. Die "Steuerung" der Floppy von Anwender erfolgt über die parallele Schnittstelle durch Steuerwörter. 1Byte bedeutet dann z.B. Read, Write, Katalog, Log, Tedete ect. Das eigentliche Formatieren und Abspeichern, Suchen, Löschen auf der Diskette wird vom DOS (Disk Operating-System) ausgeführt. Somit ist das Floppy-Interface an jedes MP-System adaptierbar sofern es über eine parallele Schnittstelle verfügt. Der Anwender hat nur noch ca. 15 Steuerbefehle von seinem MP-System an das Floppy-System zu schicken und dann nur noch Daten auszugeben oder abzuholen. Z.B. wird durch den Befehl "Katalog", welcher durch 1 Byte gekennzeichnet ist, die Inhaltsübersicht der Diskette an das Anwendersystem übergeben, ohne das dieser die Stelle kennt, wo es auf der Diskette abgelegt wurde. Allerdings kann auch das - falls gewünscht - festgestellt werden. Hierdurch ist die Anwendersoftware zur Korrespondenz mit dem Floppy-System auf ein Minimum beschränkt.

* Im Normalfall werden Floppy-Interfaces speziell für ein System ausgelegt. Dies hat aber eben diesen Nachteil, daß es für andere Systeme nicht verwendet werden kann. Das hier vorgestellte Interface umgeht diesen Nachteil durch den Einsatz einer parallelen Schnittstelle, eigenen CPU (Z80), Systemspeicher und des DOS.

Erklärung

Durch das DOS werden folgende Befehle erkannt: Write, Read, Katalog, Lock, Unlock, Open, Close, Reanag, Writesector, Readsector, Init, Delete, Drivenr, Side.
Ausführliche Anleitungen werden zum DOS mitgeliefert.

Drive-Schnittstelle

Die Drive-Schnittstelle ist auf einem 64 pol. Wire-Wrap-Stützpunkt herausgeführt und wird mittels eines 34 pol. Litzenleiterkabels und eines entsprechenden Leiterplattensteckers an den Drive angeschlossen. Für BASF, Shugart oder kompatible Minilaufwerke kann der Stecker direkt aufgesteckt werden. Andere, auch Standardlaufwerke, müssen an die Anschlußbelegung angepaßt werden. Mehrere Drives können durch aufpressen weiterer Stecker einfach adaptiert werden.

Parallel-Schnittstelle

Die Parallel-Schnittstelle ist für den 8255 ausgelegt (Betriebsart 2). Mit der HC 48 I/O line Karte kann diese unter Zuhilfenahme eines 26 pol Litzenleiterkabels direkt verbunden werden. Andere Schnittstellen müssen an die Steckerbelegung angepaßt werden. Die Schnittstelle arbeitet bidirektional für 8 Datenbits mit 4 Handshakesignalen: ACK Lesen, ~~A~~ STB Schreiben, OBF Quittungssignal Ausgabepuffer voll, IBF Eingabepuffer voll. Diese Signale werden entsprechend ihren Erfordernissen an ihre Parallelschnittstelle angeschlossen. Durch diese Beschaltung werden die Richtungen umgeschaltet und beide Systeme können erkennen, ob die Schnittstelle für neue Informationen bereit ist.

Das Interface

Das Kernstück des Interfaces ist der Floppycontroller 1791, der mit einigen zusätzlichen TTLs und einer Z80 PIO die Ansteuerung der Drives übernimmt.
Die Z80 CPU ist für die Verarbeitung des Urladeprogramms des DOS und für die Bedienung der Paralellschnittstelle zuständig. Über eine konventionelle Adreßdecodierung werden 1K Byte EPROM (2708/2758), zur Unterbringung des Urladeprogramms und wichtiger Initialisierungsroutinen des Controllers, 8K Byte RAM, zur Unterbringung des von der Diskette zu ladenden DOS, und des Datentransfers, die PIO und die Parallelschnittstelle decodiert.
Der Systemtakt wird aus einem 8 MHz Mutteroszillator, der entsprechend mit einem 74 LS 193 geteilt wird, gewonnen.

Das Interface benötigt folgende Spannungen und Ströme:
+5V/ ,(-5V/50mA) nur bei Einsatz von einem EPROM 2708,
+12V/
Der Drive benötigt eine Spannungsversorgung von:
+5V/ ,+12V/

Aufbauhinweise

Wegen des komplexen Aufbaus der Platine sollte unbedingt darauf geachtet werden, daß sauber und ohne kalte Lötstellen gelötet wird. Weiterhin sollten alle ICs auf Fassungen gesetzt werden, (bei HC-Bausätzen und Fertiggeräten selbstverständlich) da diese teilweise nicht gerade billig sind. Beachten Sie beim Bestücken, daß das IC 21 (Z80 CPU) entgegen den anderen einzusetzen ist.

Anschließend ist der Drive und die Parallelschnittstelle entsprechend den gesonderten Applikationen und die Spannungsversorgung anzuschließen.
Das Interface sollte noch nicht unter dem Drive angebracht sein.

Testhinweise

Zum Betrieb des Floppysystems ist das Urlade-EPROM und DOS erforderlich. Diese werden im folgenden vorausgesetzt.
Nach Einschalten des Floppysystems muß der Kopf des Laufwerkes auf Spur 0 fahren und der Motor ausgeschaltet werden. Jetzt wird ein Befehl zum Laden des DOS oder Initialisieren einer Diskette vom Anwendersystem erwartet. Die weitere Handhabung des DOS siehe gesonderte DOS-Beschreibung.

Kontrollieren Sie das korrekte Handshaking der beiden Parallelschnittstellen. Diese müssen sich automatisch nur durch schreiben oder Lesen gesteuert auf die richtige Richtung einstellen. An zwei verschiedenen Bits kann erkannt werden, ob die Schnittstelle bereit ist die nächste Information zu verarbeiten. Genauere Beschreibung hierzu siehe Datenblätter dieser Bausteine.

16 Kanal-8Bit-AD/DA-Interface

Das HC AD/DA-Interface befindet sich auf einer doppelseitig beschichteten und durchkontaktierten Europakarte und ist mit einer 64 pol Steckerleiste sowie einem 26 pol Wire-Wrap Stützpunkt versehen, passend zum HC Bus. Über den 26 pol Wire-Wrap-Stützpunkt wird eine zweite Karte, die zum Betrieb des AD/DA-Wandlers erforderlich ist, die Clock, angeschlossen. Die Clock-Karte gehört zum Lieferumfang und übernimmt das Timing der Messungen.

Das Interface hat 8 analoge Ein- und 8 analoge Ausgänge, welche vom Prozessor einzeln angewählt werden können. An die analogen Eingänge können Spannungen bis max. +2550mV angeschlossen werden, bzw. an den analogen Ausgängen ist die max. Spannung + 2550mV.

Alle TTLs in LS Ausführung und alle vom Bus benötigten Signale werden gepuffert.

Erklärung

Die Ansteuerung wird vom Mikroprozessor über einen PIA-Baustein (MC 6821) übernommen.
Den Kern bildet der 8 Bit DA Umsetzer ZN 425E. Dieser Baustein besitzt einen 8 Bit Zähler, dessen Ausgänge Bit 0 - 7 herausgeführt sind und ein daran angeschlossenes R/2R-Netzwerk, an dessen Ausgang, die dem Zählerstand entsprechende Spannung, verfügbar ist.
Über den Logikselekt-Eingang können die Zählerausgänge abgeschaltet und die Bit-Anschlüsse als Eingänge benutzt werden.
Am Analog-Ausgang liegt ein Impedanzwandler (741), an dem der Offsetabgleich (10K Trimmer) und die Einstellung des Spannungsendwertes (5K Trimmer) vorgenommen werden können. Dem Impedanzwandler folgt eine Sample-AND-HOLD-Schaltung, spannungsfolgern mit FED-Eingängen (TL 064 o.Ä.).
Die Ausgänge des Demultiplexers werden über drei Adressbits angewählt (PB0 - PB2).

Die 8 Analog-Eingänge werden über einen Multiplexer (4051) an den Eingang eines Komparators (TCA 325 geführt.
Beim Messen einer Spannung geschieht folgendes: Der externe Clock-Generator zählt den 8-Bit-Zähler des Wandlers hoch. Die dem Zählerstand entsprechende Ausgangsspannung wird über einen Komparator mit der Eingangsspannung verglichen. Sobald der Komparator kippt, wird ein Bit im PIA-Baustein gesetzt, welches von der CPU abgefragt werden kann. Danach kann der digitale Wert gelesen werden.

Aufbauhinweise

Beachten Sie, daß R6 in die beiden äußeren Löcher eingesetzt wird.

Testhinweise

Beide Karten werden auf den Bus gesteckt (ohne daß Spannung anliegt) und mittels eines 26 pol. Kabels (siehe auch Bild) miteinander verbunden.

Jetzt wird unter Zurhilfenahme der "Programmierhilfen" ein kurzes Programm erstellt, die die einzelnen Aus- bzw. Eingänge des AD/DA-Wandler ansprechen. Testen Sie die Adressdecodierung anhand der Speichertesthinweise und überprüfen Sie die Clock-Frequenz des Glock-Generators. (1mHz).

Mittels der bislang beschriebenen Komponenten wird der Anwender in die Lage versetzt, einen eigenen Personalcomputer aufzubauen. Die Größe der Ausbaustufe kann selbst bestimmt werden.

Neben den bisher beschriebenen peripheren Geräten besteht die Möglichkeit zum Anschluß von Großdruckern und Terminals. Weiterhin versetzt Sie das System in die Lage, jede Art von Steuerungen auszuführen.

Software

Voraussetzungen zum Verständnis dieses Abschnittes sind Grundlagen in Digitaltechnik und Bool'scher Algebra. Sollten diese Kenntnisse nicht vorhanden sein, empfehlen wir, die entsprechende Literatur zu diesen Themen vorher durchzuarbeiten.

Die Grundlage für die Arbeitsfähigkeit eines Computers ist ein Programm. Für die Erstellung eines solchen Programmes gibt es verschiedene Möglichkeiten, wobei das Endergebnis, das reine Maschinenprogramm, immer dasselbe ist. Die Unterschiede bestehen also nur in der Art und Weise in der das Programm erstellt und eingegeben wird. Die erste Möglichkeit ist das Schreiben von Programmen in Maschinensprache. Dabei werden die Befehle des SC/MP direkt in hexadezimaler Form aufgestellt und in die Maschine eingegeben. Diese Methode hat die Vorteile, daß nur kleine Anforderungen an die Hardware gestellt werden (nur HEX I/O und Speicher erforderlich) und das man immer einen Einblick in die Arbeitsweise des Mikrocomputers hat. Die Nachteile bestehen in der schlechten Überschaubarkeit größerer Programme und in mühseligen Berechnungen einzelner Befehle. Um diese Nachteile auszugleichen wurden sog. Assembler-Programme entwickelt. Diese Programme benötigen als Befehl nicht mehr den hexadezimalen Code, sondern eine mnemonische (leicht zu merkende) Abkürzung für die Befehle. Evtl. auftretende Berechnungen werden vom Assembler miterledigt. Der prinzipielle Aufbau der Programme bleibt jedoch unverändert. Um auch an diesen Aufbau nicht gebunden zu sein, wurden höherorientierte Programmiersprachen entwickelt. Sie sind der menschlichen Sprache angenähert und werden in zwei Gruppen unterschieden.

Die erste Gruppe ist die der Compiler. Ein solcher Compiler übersetzt ein Programm einer Programmiersprache in ein Maschinenprogramm.

Die zweite Gruppe, die der Interpreter, übersetzt das Programm nicht, sondern arbeitet für jeden Programmbefehl eine Anzahl Maschinenbefehle ab. Die verschiedenen Programmiersprachen wurden für die Lösung unterschiedlicher Probleme entwickelt, wie mathematische Probleme, kaufmännische Probleme, Textverarbeitung usw. Die gebräuchlichsten Sprachen sind:

FORTRAN IV
BASIC
COBOL
ALGOL
PASCAL
PL/I

Für den SC/MP-Microcomputer stehen mehrere Versionen der Programmiersprache BASIC als Interpreter sowie Two Pass Assembler und Disassembler (Übersetzung von Maschinenprogrammen) zur Verfügung.

Programme in Maschinensprache

Mit Hilfe des Monitorprogrammes sind Sie nun in der Lage, eigene Programme einzugeben. Der Speicherbereich für ein solches Programm liegt wie beschrieben bei den Adressen 0C00-0FC9.

Wir wollen ein einfaches Problem lösen: Die Zahlen 3 + 4 sollen addiert und das Ergebnis auf dem 7 - Segment Display angezeigt werden.

Folgende Vorgänge müssen nun ablaufen:
Die Zahl 3 muß geladen werden
Die Zahl 4 muß geladen und zur Zahl 3 addiert werden.
Das Ergebnis muß zum Display transportiert werden.
Das Programm soll bei der Adresse 0C00 liegen.
Der Befehl zum Laden der Zahl 3 lautet: LDI 3 mit dem Op-Code (403). Die Zahl 3 steht nun im Akku des SC/MP. Mit dem Befehl ADI (F404) wird dazu die Zahl 4 addiert. Im Akku steht die Zahl 7. Nun muß diese Zahl zum Display gebracht werden. Die Adresse des Displays ist 0700. Wir laden diese Adresse in das PTR 1.

Um dabei unser Ergebnis (die Zahl im Akku nicht zu zerstören, retten wir diese ins E-Register. Der Befehl dazu lautet: XAE (o1). Nun laden wir in das PTR 1 Higher Byte mit der Befehlsfolge LDI 7 (C407); XPAH 1 (35) die Zahl 7 und in PTR 1 Lower Byte mit LDIO (C400); XPAL (31) die Zahl 0. Im Anschluß daran bringen wir unser Ergebnis

von vorhin wieder in den Akku mit dem Befehl, LDE (40) und speichern diesen Wert Pointer 1 relativ im Display ab. Der Befehl lautet: STO (1) (C900). Danach lassen wir den Prozessor anhalten mit HALT (00). Hier noch einmal das Programm komplett:

Adresse	Befehl
0C00	C403
0C02	F404
0C04	01
0C05	C407
0C07	35
0C08	C400
0C0A	31
0C0B	40
0C0C	C900
0C0E	00

Geben Sie dieses Programm mit der Modify-Routine ein und starten Sie es mit der RUN-Routine. Achten Sie auf das rechte Display, auf dem das Ergebnis erscheinen muß.

Das auf dem Display nun wirklich 7 steht ist reiner Zufall. Ändern Sie das Programm so ab, daß es die Zahlen 5 + 4 addiert. Sie müssen dazu nur das zweite Byte (Adresse 0C01) von 03 in 05 ändern. Starten Sie das Programm erneut. Es sollte auf dem Display 9 stehen, es erscheinen aber nur zwei Striche. Dies hat folgenden Grund: Wenn Sie eine Zahl unter der Adresse des Displays abspeichern, wird diese von Display nicht selbständig übersetzt, d.h. für jedes Bit in der Zahl, die Sie dem Display schicken, leuchtet ein Segment auf, wenn das Bit gleich 1 ist. Die Bits sind folgenden Segmenten zugeordnet:

Bit 7 (MSB=höchste Bit)	= DP	
Bit 6	= Segm.	G
Bit 5	= "	F
Bit 4	= "	E
Bit 3	= "	D
Bit 2	= "	C
Bit 1	= "	B
Bit 0 (LSB=niedrigstes Bit)	= "	A

Wo die Segmente liegen entnehmen Sie der HEX I/O-Beschreibung. Wie Sie sehen, weshalb es Zufall war, daß bei dem Ergebnis 3 + 4 wirklich 7 auf dem Display stand. Bei allen anderen Zahlen steht auf dem Display nur Unsinn. Um jede Zahl auf dem Display darstellen zu können, müssen wir den Binär-Code in ein 7-Segment-Code übersetzen. Dies erreicht man am einfachsten mit einer Tabelle. Wir legen als Basis-Adresse für die Tabelle 0D00 fest. Die Tabelle hat folgenden Aufbau:

Unter der Adresse 0D00 steht der 7-Segment-Code für die Zahl 0 (3F). Unter der nächsten Adresse (0D01) der Code für die Zahl 1 (06). So geht es weiter bis zur HEX-Zahl F, mit dem 7-Segment-Code 71 unter der Adresse 0D0F.

Nehmen wir wieder unser altes Programm, daß die Zahlen 5 + 4 addiert. Das Programm bleibt bis zur Adresse 0C0A erhalten. Als nächstes laden wir die Adresse unserer Tabelle in das PTR 2. Die Befehle dazu lauten:

LDI OD (C40D); XPAH 2 (36); LDIO (C400); XPAL 2 (32).

Das Ergebnis der Addition steht noch im E-Register. Wir benutzen nun einen indirekten Ladebefehl: LD X'80 (2) (CZ 80).

Mit diesem Befehl wir der dem Ergebnis entsprechene 7-Segment-Code aus der Tabelle geladen. Hieran hängen wir nun den Teil des alten Programms, den wir vorhin weggelassen haben.

Das komplette Programm lautet:

0C00	C405
0C02	F404
0C04	01
0C05	C407
0C07	35
0C08	C400
0C0A	31
0C0B	C40D
0C0D	36
0C0E	C400
0C10	32
0C11	C280

0C13	C900
0C15	00
0D00	3F
0D01	06
0D02	5B
0D03	4F
0D04	66
0D05	6D
0D06	7D
0D07	07
0D08	7F
0D0	6F
0D0A	77
0D0B	7C
0D0C	58
0D0D	5E
0D0E	79
0D0F	71

Eigentlich wäre es nicht nötig gewesen, diese Tabelle in das Programm aufzunehmen, denn das Monitor-Programm Elbug enthält die gleiche Tabelle. Sie liegt bei der Adresse /11F (Erläuterungen Elbug).

Es würde den Rahmen dieses Buches sprengen, alle Vorgänge so zu erklären, wie die Addition zweier Zahlen. Im folgenden finden Sie jedoch einige Programmlistings, deren Analyse Ihnen bei der Lösung Ihrer Aufgaben helfen will.

Wie schreibe ich ein Programm?

Das Problem

Zuerst wird das spezielle Programm, das zu einem Computer-Programm werden soll, definiert.

1. Es werden die Eingangswerte festgelegt.
2. Es werden Ausgangswerte = Das Ergebnis = festgelegt.
3. Es wird festgelegt, auf welche Art diese beiden Werte voneinander ab-

hängig sind.
4. Es werden Störmeldungen festgelegt, die bei fehlerhafter Eingabe oder anderen Störungen warnen.
5. Es wird eine Zeit festgelegt, innerhalb der das System antworten muß.

Das Flußdiagramm

Der Programmentwurf wird zur besseren Übersicht in einem Flußdiagramm dargestellt. Seine Vorteile sind:

1. mehrfach geschachtelte Programme sind besser zu überblicken.
2. überflüssige Verzweigungen werden sichtbar gemacht.
3. überflüssige Funktionen sind besser zu erkennen.

Flußdiagramm-Symbole sind standardisiert.

Die Befehle

Das im Flußdiagramm dargestellte Programm wird jetzt in die Assemblersprache (Mnemonics) übertragen. Hierzu bedient man sich eines Programmierformulars. Danach wird dieses Quellprogramm (Source-Programm) in die Maschine eingegeben. Dies kann sowohl über eine ASCII-Tastatur auf dem direkten Wege erfolgen oder indirekt über einen zu erstellenden Datenträger wie z.B. den Lochstreifen.

Das Quellprogramm ist das vom Programmierer in irgendeiner Assemblersprache geschriebene Programm. Hieraus wird vom Assembler das Objektprogramm gefertigt. Dies ist das endgültige Programm in der Maschinensprache. Zur Korrektur während des Programmierens wird ein Editor gebraucht; zum Austesten benötigt man ein Debug-Programm, und um das Programm, bevor man es in die Maschine lädt, von den nun überflüssigen Kommentaren, Zeilennummern und Adressen zu befreien, benötigt man einen Loader. Dieser dient dazu, eine lückenlose Aneinanderreihung von Maschinenbefehlen in das Arbeitsspeicher-Medium zu laden.

Es dürfen Buchstaben, Ziffern und Sonderzeichen verwendet werden. Buchstaben von A - Z, Ziffern von 0 - 9 und folgende Sonderzeichen:

$	:	@	SHIFT P
%	;		
&	!		
'	"		
(	,		
)	-		
*	.		
+	/		

Vom Fernschreiber werden verschiedene Sonderzeichen durch zwei Zeichen dargestellt. Siehe Bild 3

Eine Assemblerzeile besteht aus:

1. Labelfeld
2. Operationsteilfeld
3. Operandenfeld
4. Kommentarfeld
5. Identifikationsfeld

Der Assembler verlangt kein festes Format für die Zeileneinteilung.

Zwischen dem zweiten und dritten Feld muß immer ein Leerzeichen stehen.

Das Labelfeld

Das Labelfeld muß nicht beschrieben werden. Es kann eine symbolische Adresse (Label) enthalten, auf die der Programmierer zurückgreifen kann. Das Feld kann mehrere Labels enthalten. Diese bezeichnen dann alle dieselbe Zeile. Symbolische Adressen können aus bis zu 32 Buchstaben oder Ziffern bestehen. Sie müssen als Abschluß einen Doppelpunkt haben. Der Assembler benutzt allerdings nur die ersten 6 Zeichen, um verschiedene Labels zu unterscheiden. Das erste Zeichen muß ein Buchstabe oder ein Dollarzeichen sein. Innerhalb eines Labels dürfen

keine Leerstellen vorkommen, auch nicht vor dem Doppelpunkt. Es ist zu empfehlen, für lokale Ansprungadressen den Labels ein Dollarzeichen vorzustellen und fortlaufend zu numerieren. Z.B. $ 1, $ 2 usw. Der Assembler besitzt keine Lokalanweisung. Deshalb müssen auch die $ - Labels eindeutig definiert sein.

Das Operationsfeld

Dieses Feld enthält den mnemonischen Operationscode oder eine Assembleranweisung. Assembleranweisungen dienen zur Steuerung des Assembliervorganges, Befehle dienen zur Programmgestaltung für den Rechner.

Die Assemblieranweisung

Alle Assemblieranweisungen beginnen mit einem Punkt.

. TITLE

Die Title-Anweisung dient dazu, ein Programm mit einem Namen zu versehen. Zusätzlich kann innerhalb zweier Apostrophs (String) eine Erklärung zum Titel gegeben werden.

. END
Dies ist die letzte Anweisung eines Programms. Nach der Endanweisung springt der Assembler in den nächsten Durchlauf.

. PAGE
Dient als Überschrift und kann mit einem zusätzlichen String versehen sein.

. BYTE
Diese Anweisung belegt für dezimale bzw. hexadezimale Eingaben im Bereich von - 128 bis + 255 (dezimal ein Byte). Mit einer Byte-Anweisung können auch mehrere Bytes bereitgestellt werden, wenn man die Operanden durch Komma trennt.

. DBYTE
Diese Anweisung bewirkt dasselbe wie die BYTE-Anweisung. Hier werden allerdings zwei Byte reserviert (von - 32768 bis + 65535). Beide Anweisungen können mit einem oder mehreren Labels versehen werden. Das Label bezieht sich immer auf das erste reservierte Byte.

. ADDR
Diese Anweisung bewirkt dasselbe wie die DBYTE-Anweisung, jedoch mit automatischer Verminderung um 1 zur Benutzung durch die SC/MP-Sprungbefehle.

. ASCII
Diese Anweisung speichert in hintereinander liegenden Bytes die Druckzeichen eine Zeichenkette im 7 Bit ASCII-Code

. SPACE
Die Space-Anweisung veranlaßt den Assembler, beim Ausdruck zwei Leerzeilen einzufügen.

Einige Anweisungen können mit nur einem Buchstaben gegeben werden.

.PAGE = .P
.BYTE = .B
.DBYTE = .D
.SPACE = .S

Pseudo-Befehl (Makro)

Ein Pseudo-Befehl wird vom Assembler wie ein normaler Befehl akzeptiert. Er hat aber im Gegensatz zu letzterem die Erzeugung mehrerer Maschinenbefehle zur Folge.

Im HSA ist ein Pseudo-Befehl enthalten. Es ist der Unterprogramm-Sprung: JS = Jump to Subroutine / Springe ins Unterprogramm.

Jump to Subroutine

JS PTR, Ausdruck
Platzbedarf 7 Bytes
LDI H (Ausdruck)
XOAH PTR
LDI L (Ausdruck)-1
XPAL PTR
XPPC PTR

Infolge der Seitenstruktur beim SC/MP sind Programme, welche eine 4 K Seite überschreiten, nicht lauffähig.

Aus diesem Grunde gibt der Assembler bei Adressierungen über die Seitengrenze hinaus die Error-Meldung "Error 3". Allerdings ist es möglich, mit PC relativer Adressierung vom Anfang einer Page die Bytes am Ende dieser Page, rückwärts gehend, zu erreichen.

Das Operandenfeld

Mit den in diesem Feld stehenden Operanden werden die im Operationsteilfeld stehenden Operationen ausgeführt. Es können mehrere Operanden zugelassen werden.

Als Operand ist ein selbsterklärender Begriff in dezimaler oder hexadezimaler Form oder in Form einer Zeichenkette (String) zulässig.

Positive Zahlen von 1 - 65535 dezimal dürfen benutzt werden. Die erste Ziffer darf nicht 0 sein, damit der Assembler die Zahl nicht hexadezimal auffaßt.

Hexadezimale Zahlen werden durch Voranstellen von X' oder 0 dargestellt. Hexadezimal ist der Bereich von 0000 bis FFFF benutzbar.

Eine Folge von Druckzeichen wird String genannt. Ein String muß von Apostrophen eingeschlossen sein. Alle druckbaren Zeichen einschließlich der Blanks sind erlaubt.

Weiterhin kann ein symbolischer Name benutzt werden, für den der Assembler erst während des Assembliervorganges einen Wert ermittelt.

Ebenso kann als Operand ein mathematischer Ausdruck verwendet werden. Der Assembler berechnet (nur Addition und Subtraktion dezimal oder hexadezimal und auch gemischt sind zulässig) den Wert eines Ausdrucks und setzt diesen für den Operanden ein.

Es besteht die Möglichkeit von 2-Byte Ausdrücken das höhere und das niedere Byte getrennt zu laden.
Die Anweisungen dazu lauten:
LDI H (Ausdruck)
LDI L (Ausdruck)

Das Kommentarfeld

Ein Kommentar muß mit einem Semikolon beginnen. Innerhalb eines Kommentars sind alle druckbaren Zeichen einschließlich eines Blanks erlaubt. Ein Kommentar kann über mehrere Zeilen gehen, jedoch muß jede Zeile mit einem Semikolon beginnen.

Der Kommentar dient zur Erläuterung des Programms. Er wird im Assemblerprotokoll wiedergegeben, hat jedoch keinen Einfluß auf den Assembliervorgang.

Das Identifikationsfeld

Dieses Feld ist nur bei einer Eingabe über Lochkarte von Bedeutung.

Bei fehlerhafter Eingabe gibt der Assembler "error-Meldungen".

Diese sind:

error 1 = fehlende Argumente
z.B.: LDI ohne den Wert

error 2 = fehlerhafter Wert
z.B.: LDI 1000 (Wert ist zu groß)

error 3 = Adressierungsfehler
z.B.: JMP X' 0150, wenn der PC auf 0010 steht

error 4 = nicht zulässiger Assemblerbefehl
z.B.: .IF

error 5 = Syntax error
z.B.: 1 CR

error 6 = zu viele Operanden
z.B.: JMP LDI

error 7 = a) Doppelt definierte symbolische Namen
a = 1
a = 2
b) Zu viele symbolische Namen
bei Assemblierung mit RAM-Bereich ca 350,
bei Lochstreifen ca. 600 für jede 4 K Karte

error Nr. R = nur bei Assemblierung mit RAM-Bereich zeigt Überschreiten des vorhandenen Bereichs an.

error Nr. E = keine Fehlermeldung, nur Hinweis (bei ST, LD usw.), daß für die Adressierung nicht der Akkumulator, sondern das Extension-Register benutzt wird.

Der SC/MP hat 46 Befehle. Diese sind unterteilt in 24 Single Byte und 22 Double Byte Befehle.

Es gibt:

8 speicherbezogene Befehle

LD	LOAD	Laden
ST	STORE	Speichern
AND	AND	Und
OR	OR	Oder
XOR	EXCLUSIVE OR	Ungleich (exclusiv oder)
DAD	DEZIMAL ADD	Dezimale Addition
ADD	ADD	Addition binär
CAD	COMPLEMENT AND ADD	Komplement und Addition

2 Speicherbefehle zum Erhöhen oder Erniedrigen

ILD	INCREMENT AND LOAD	Erhöhen und Laden
DLD	DECREMENT AND LOAD	Erniedrigen und Laden

7 unmittelbare Befehle

LDI	LOAD IMMEDIATE	Laden unmittelbar
ANI	AND IMMEDIATE	Und unmittelbar

ORI	OR IMMEDIATE	Oder unmittelbar
XRI	EXCLUSIVE OR IMMEDIATE	Ungleich unmittelbar
DAI	DECIMAL AND IMMEDIATE	Dezimal Addition unmittelbar
ADI	ADD IMMEDIATE	Addition unmittelbar
CAI	COMPLEMENT AND ADD IMMEDIATE	Komplement und Addition unmittelbar

4 bedingte und unbedingte Sprünge

JMP	JUMP	Sprung (unbedingt)
JP	JUMP IF POSITIV	Sprung, wenn AC positiv
JZ	JUMP IF ZERO	Sprung, wenn AC = 0
JNZ	JUMP IF NOT ZERO	Sprung, wenn AC = 0

1 Zeitverzögerungsbefehl

DLY	DELAY	Zeitverzögerung

8 Befehle mit dem Erweiterungsregister

LDE	LOAD AC FROM EXTENSION	AC laden von E
XAE	EXCHANGE AC AND EXTENSION	AC und E tauschen
ANE	AND EXTENSION	Und mit E Register
ORE	OR ECTENSION	OR mit E Register
XRE	EXCLUSIV OR EXTENSION	Ungleich mit E Register
DAE	DEZIMAL ADD EXTENSION	Dezimal Addition mit E Register
ADE	ADD EXTENSION	Binär Addition mit E Register
CAE	COMPLEMENT AND ADD EXTENSION	Komplement und Addition mit E Register

3 Manipulations-Befehle mit dem Pointerregister

XPAL	EXCHANGE POINTER LOW	Wechsel Pointer Lower Byte
XPAH	EXCHANGE POINTER HIGH	Wechsel Pointer Higher Byte
XPPC	EXCHANGE POINTER WITH PC	Wechsel Pointer mit PC

5 Befehle für Schieben, Rotation und serielle Ein- und Ausgabe

SIO	SERIAL INPUT/OUTPUT	Serielle Ein- und Ausgabe
SR	SHIFT HIGH	Rechts schieben
SRL	SHIFT RIGHT WITH LINK	Rechts schieben mit Link
RR	ROTATE RIGHT	Rechts im Kreise schieben
RRL	ROTATE RIGHT WITH LINK	Rechts im Kreise schieben mit Link

8 sonstige Befehle

HALT	HALT	Halt
CCL	CLEAR CARRY/LINK BIT	Carry/Link Bit löschen
SCL	SER CARRY/LINK BIT	Carry/Link Bit setzen
DINT	DISABLE INTERRUPT	Interrupt sperren
IENT	ENABLE INTERRUPT	Interrupt ermöglichen
CSA	COPY STATUS TO AC	Status Information in AC übertragen
CAS	COPY AC TO STATUS	AC in Status Register übertragen
NOP	NO OPERATION	Leer Befehl

Beispiel für ein Assemblerprogramm

1. Durchlauf

```
pt/ra? pt
kb/me? kb



bnoe m a/b



h s a
homecomputer gmbh , ahnfeldstr.55 , 4000 duesseldorf


  0001                .title demo1  'si-druck-iv'
  0002                .,kl. betriebsprogramme, programm (9)
                      .s

  0003                .page  1
  0004                .,definition der veraenderlichen
                      .s

  0005       0020     blank  =x'20            a .,ascii zeichen'blank'
  0006       1807     sbr    =x'1807          .,addr. unterprogramm
  0007       0000     xx     =x'00            .,startaddr. hi+lo des
  0008       0000     yy     =x'00            .,auszulesenden progr.
                      .s

  0009                .page  2
  0010       1790     .=     x'1790           .,pc auf startaddresse
  0011 1790 c400      ldi    xx               .,start hi+lo laden
  0012 1792 35        xpah 1
  0013 1793 c400      ldi    yy               .,und in ptr 1 setzen.
  0014 1795 31        xpal 1
                      .s

  0015                4 ?/1:                     .,erstes ansprunglabel
  0016 1796 c404      ldi    x'04             .,ldi 4 auch erlaubt.
  0017 1798 0808      st     stack+1          .,label stack wird bei
error nr.  2
  0018 179a c400      ldi    00               .,addr. 1803 definiert
  0019 179c 32        xpal 2                  .,addr. tv lo in ptr 2
                      .s

  0020                ?/2:
  0021 179d c4fc      ldi    x'fc             .,addr. tv hi
  0022 179f 36        xpah 2
  0023                .,im originalprogramm wird an dieser stelle
  0024                .,das ptr 3 mit der addr. der sbr-1 geladen.
  0025                .,dies erreicht man beim assembler mit dem
  0026                .,macro befehl 'js' einfacher
  0027 17a0 c410      ldi    16               .,(x'10 oder 010)
  0028 17a2 0808      st     stack
error nr.  2
                      .s

  0029  -             ?/3:
  0030 17a4 c420      ldi    blank            .,ascii zeichen



  0031 17a6 ce01      st     @1(2)            .,auto indexed addr.
  0032 17a8 ce01      st     @1(2)            .,(tv loeschen)
  0033 17aa 35        xpah 1                  .,inhalt von ptr 1 hi
  0034 17ab 01        xae                     .,in e-register retten
  0035 17ac 40        lde
  0036 17ad 35        xpah 1
  0037 17ae c418      js 3,  sbr              .,macro befehl
       17b0 37c4
       17b2 0633
       17b4 3f
  0038                .,vorsicht bei befehlen, in dener ein ptr an-
  0039                .,gegeben wird. (z.b.xpal 2, xppc 3, js 3,xyz)
  0040                .,es muss zwischen befehl und ptr-angabe
  0041                .,genau ein 'blank' stehen.
  0042 17b5 31        xpal 1                  .,inhalt von ptr 1 lo
```

```
  0043 17b6 01     xae                    .,in e-register retter
  0044 17b7 40     lde                    .,(befehlsaddrgt .)
  0045 17b8 31     xpal 1
  0046 17b9 3f     xppc 3                 .,erneuter sbr anspr.
  0047              .,'js' nicht noetig, da am ende der sbr ein
  0048              .,ruecksprungbefehl vorhanden ist.
  0049 17ba c420   ldi    blank
  0050 17bc ce01   st     @1(2)          .,2 a 'blank'
  0051 17be ce01   st     @1(2)
  0052 17c0 c501   ld     @1(1)          .,1. befehlsbyte
  0053 17c2 01     xae
  0054 17c3 3f     xppc 3
  0055 17c4 c420   ldi    blank
  0056 17c6 ce01   st     @1(2)
  0057 17c8 c5ff   ld     @-1(1)
  0058 17ca 0808   jp     ?/4             .,2-byte befehl?
error nr.  2
  0059 17cc c501   ld     @1(1)          .,2. befehlsbyte
  0060 17ce 01     xae
  0061 17cf 3f     xppc 3
  0062 17d0 0808   jmp    ?/5
error nr.  2
                  .s

  0063              ?/4:                  .,bei 1-byte befehl
  0064 17d2 c420                .,ascii z. 'blank'
  0065 17d4 ce01   st     @1(2)
  0066 17d6 ce01   st     @1(2)



  0067              ?/5:
  0068 17d8 c420   ldi    blank
  0069 17da ce01   st     @1(2)
  0070 17dc ce01   st     @1(2)
  0071 17de ce31   st     @49(2)
  0072 17e0 0808   dld    stack          .,addr. x'1803
error nr.  2
  0073 17e2 9cc0   jnz    ?/3
  0074 17e4 ce10   st     @16(2)
  0075 17e6 0808   dld    stack+1        .,addr. x'1804
error nr.  2
  0076 17e8 e403   xri    3
  0077 17ea 0808   jz     ?/6
error nr.  2
  0078              .,im 1. durchgang errormeldung, da dem assem.
  0079              .,?/6 nov ch nicht bekannt ist.
  0080 17ec 0808   ld     stack+1
error nr.  2
  0081 17ee 9cad   jnz    ?/2
                  .s


  0082 17f0 00     halt                   .,programmunterbrech.
  0083                                    .,(bildschirm voll)
  0084 17f1 0808   ld     stack+2
error nr.  2
  0085 17f3 35     xpah 1
  0086 17f4 0808   ld     stack+3
error nr.  2
  0087 17f6 31     xpal 1
  0088 17f7 909d   jmp    ?/1
                  .s

  0089              ?/6:
  0090 17f9 35     xpah 1
  0091 17fa 0808   st     stack+2
error nr.  2
  0092 17fc 35     xpah 1
  0093 17fd 31     xpal 1
  0094 17fe 0808   st     stack+3
error nr.  2
  0095 1800 31     xpal 1
  0096 1801 909a   jmp    ?/2
                  .s



  0097              stack:                .,4 speicher byte
  0098 1803 55     .byte  x'55, x'55
       1804 55
  0099 1805 5555   .dbyte x'5555
                  .s

  0100              .page  sbr     'unterprogramm'
  0101 1807 40     lde                    .,umwandlung hex in
  0102 1808 1c     sr                     .,ascii.
```

```
  0103 1809 1c      sr
  0104 180a 1c      sr
  0105 180b 1c      sr
                    .s

  0106              ?/8:
  0107 180c 02      ccl
  0108 180d f4f6    adi     0f6
  0109 180f 0808    jp      ?/9               .,sprung bei bus chst.
error nr.  2
  0110 1811 f4f9    adi     0f9
                    .s

  0111              ?/9:
  0112 1813 f440    adi     040               .,umwandlung in ascii
  0113 1815 ce01    st      ø1(2)             .,und anzeige
  0114 1817 06      csa
  0115 1818 e401    xri     1                 .,flag 0 invertieren
  0116 181a 07      cas
  0117 181b d40f    ani     0f
  0118 181d 90ed    jmp     ?/8
                    .s

  0119              ?/10:
  0120 181f 3f      xppc 3                    .,ruecksprung ins
  0121 1820 90e5    jmp     x'1807            .,hauptprogramm.
  0122              .end
kb
```

2. Durchlauf

```
0001              .title demo1  'si-druck-iv'
0002              .,kl. betriebsprogramme, programm (9)

0003              .page  1
0004              .,definition der veränderlichen

0005      0020    blank   =x'20             .,ascii zeichen'blank'
0006      1807    sbr     =x'1807           .,addr. unterprogramm
0007      0000    xx      =x'00             .,startaddr. hi+lo des
0008      0000    yy      =x'00             .,auszulesenden progr.

0009              .page  2
0010      1790    .=      x'1790            .,ab auf startaddresse
0011 1790 c400    ldi     xx                .,start hi+lo laden
0012 1792 35      xpah 1
0013 1793 c400    ldi     yy                .,und in ptr 1 setzen.
0014 1795 31      xpal 1

0015              ?/1:                       .,erstes ansprunglabel
0016 1796 c404    ldi     x'04              .,ldi 4 auch erlaubt.
0017 1798 c96b    st      stack+1           .,label stack wird bei
0018 179a c400    ldi     00                .,addr. 1803 definiert
0019 179c 32      xpal 2                    .,addr. tv lo in ptr 2

0020              ?/2:
0021 179d c4fc    ldi     x'fc              .,addr. tv hi
0022 179f 36      xpah 2
0023              .,im originalprogramm wird an dieser stelle
0024              .,das ptr 3 mit der addr. der spr-1 geladen.
0025              .,dies erreicht man beim assembler mit dem
0026              .,macro befehl 'js' einfacher.
0027 17a0 c410    ldi     16                .,(x'10 oder 010)
0028 17a2 c860    st      stack

0029              ?/3:
0030 17a4 c420    ldi     blank             .,ascii zeichen
0031 17a6 ce01    st      ø1(2)             .,auto indexed addr.
0032 17a8 ce01    st      ø1(2)             .,(tv loeschen)
```

```
0033 17aa 35      xpah 1                    .,inhalt von ptr 1 hi
0034 17ab 01      xae                       .,in e-register retten
0035 17ac 40      lde
0036 17ad 35      xpah 1
0037 17ae c418    js 3,  sbr                .,macro befehl
     17b0 37c4
     17b2 0633
     17b4 3f
0038              .,vorsicht bei befehlen, in denen ein ptr an-
0039              .,gegeben wird. (z.b.xpal 2, xppc 3, js 3,xyz)
0040              .,es muss zwischen befehl und ptr-angabe
0041              .,genau ein 'blank' stehen.
0042 17b5 31      xpal 1                    .,inhalt von ptr ' lo
0043 17b6 01      xae                       .,in e-register retten
0044 17b7 40      lde                       .,(befehlsaddr.)
0045 17b8 31      xpal 1
0046 17b9 3f      xppc 3                    .,ernelter sbr anspr.
0047              .,'js' nicht noetig, da am ende der sbr ein
0048              .,rueckspringbefehl vorhanden ist.
0049 17ba c420    ldi    blank
0050 17bc ce01    st     ø1(2)              .,? o 'blank'
0051 17be ce01    st     ø1(2)
0052 17c0 c501    ld     ø1(1)              .,1. befehlsbyte
0053 17c2 01      xae
0054 17c3 3f      xppc 3
0055 17c4 c420    ldi    blank
0056 17c6 ce01    st     ø1(2)
0057 17c8 c5ff    ld     ø-1(1)
0058 17ca 9406    jp     ?/4                 .,2-byte befehl?
0059 17cc c501    ld     ø1(1)              .,?  befehlsbyte
0060 17ce 01      xae
0061 17cf 3f      xppc 3
0062 17d0 9006    jmp    ?/5


0063              ?/4:                       .,bei 1-byte befehl
0064 17d2 c420    ldi    blank              .,ascii z. 'blank'
0065 17d4 ce01    st     ø1(2)
0066 17d6 ce01    st     ø1(2)
0067              ?/5:
0068 17d8 c420    ldi    blank
0069 17da ce01    st     ø1(2)
0070 17dc ce01    st     ø1(2)



0071 17de ce31    st     ø49(2)
0072 17e0 b822    dld    stack              .,addr. x'1803
0073 17e2 9cc0    jnz    ?/3
0074 17e4 ce10    st     ø16(2)
0075 17e6 b81d    dld    stack+1            .,addr. x'1804
0076 17e8 e403    xri    3
0077 17ea 930d    jz     ?/6
0078              .,im 1. durchgang errormeldung, da dem assem.
0079              .,?/6 noch nicht bekannt ist.
0080 17ec c017    ld     stack+1
0081 17ee 9cad    jnz    ?/2


0082 17f0 00      halt                      .,programmunterbrech.
0083                                        .,(bildschirm voll)
0084 17f1 c013    ld     stack+2
0085 17f3 35      xpah 1
0086 17f4 c011    ld     stack+3
0087 17f6 31      xpal 1
0088 17f7 909d    jmp    ?/1


0089              ?/6:
0090 17f9 35      xpah 1
0091 17fa c80a    st     stack+2
0092 17fc 35      xpah 1
0093 17fd 31      xpal 1
0094 17fe c807    st     stack+3
0095 1800 31      xpal 1
0096 1801 909a    jmp    ?/2


0097              stack:                    .,4 speicher byte
0098 1803 55      .byte  x'55, x'55
     1804 55
0099 1805 5555    .doyte x'5555


0100              .page  sor      'unterprogramm'
0101 1807 40      lde                       .,umwandlung hex in
0102 1808 1c      sr                        .,ascii.
```

```
0103 1809 1c      sr
0104 180a 1c      sr
0105 180b 1c      sr



0106              ?/8:
0107 180c 02      ccl
0108 180d f4f6    adi     0f6
0109 180f 9402    jp      ?/9               .,sprung bei buchst.
0110 1811 f4f9    adi     0f9


0111              ?/9:
0112 1813 f440    adi     040               .,umwandlung in ascii
0113 1815 ce01    st      @1(2)             .,und anzeige
0114 1817 06      csa
0115 1818 e401    xri     1                 .,flag 0 invertieren
0116 181a 07      cas
0117 181b d40f    ani     0f
0118 181d 90ed    jmp     ?/8


0119              ?/10:
0120 181f 3f      xppc 3                    .,rueecksprung ins
0121 1820 90e5    jmp     x'1807            .,hauptprogramm.
0122              .end
```

Das Monitorprogramm Elbug

Um Mikroprozessoren zur besseren, flexibleren und preisgünstigeren Lösung von Aufgaben der Digital-Elektronik einsetzen zu können, benötigt man zunächst eine Einrichtung zum Entwickeln und Austesten von Programmen - ein Entwicklungs-System.

Das 7-Segment-Hexadezimal-Ausbaustadium des SC/MP - Mikrocmputers zusammen mit der Betriebs-Software "ELBUG" stellt bereits ein einfaches Entwicklungssystem dar. Die etwas bescheidene Architektur und Befehlsstruktur des SC/MP wird durch dieses Gesamt-System so erweitert (LIFO-Stack, Block-Umlade-Kommando, 8 Interrupt-Eingänge usw.), daß das Komfort-Niveau wesentlich teurer uP's erreicht wird.

Es hat sich als sinnvoll erwiesen, Programme für einen bestimmten Mikroprozessor auf einem Mikrocomputer, der eben diesen uP als CPU beinhaltet, zu entwickeln weil:

- der Anwender nicht gleich zwei verschiedene Befehlssätze (Programmiersprachen) erlernen muß
- die fertigen Programme auf dem Entwicklungssystem, vor Erstellen der Hardware für die betreffende Anwendung bereits getestet werden können und weil
- beim Aufbau dieses Systems schon spezifische Kenntnisse für den betreffenden uP-Typ erworben wurden.

Arbeiten mit der Elbug

Beim Anlegen der Versorgungsspannungen (- 12 V jetzt auch erforderlich) erscheint "... ELBUG" auf dem Display. Falls nicht, liegt ein Fehler in der Hardware vor.

Das Display ist funktionell in 4 Felder aufgeteilt:

Digit 0 und 1 (die beiden rechten Anzeigen)	= Datenfeld
Digit 2 und 5 (die vier mittleren Anzeigen)	= Adressfeld
Digit 6 und 7 (die beiden linken Anzeigen)	= Kommandofeld

Dezimalpunkte auf dem Display bedeuten, daß das Programm eine Eingabe über die Tastatur erwartet; die beiden Punkte von " .. EL-BUG" verlangen das Betätigen einer Kommando-Taste.

Diese sind innerhalb des Programms wie folgt definiert:

F0 = RUN
E0 = MODIFY
D0 = SUBTRAKTION
C0 = KASSETTE
B0 = BLOCK-TRANSFER
A0 = CPU-REGISTER
90 = DOWN
80 = UP

Die beiden letzten sind keine echten Kommando-Tasten, sondern "Suffix"-Tasten, welchen nach Betätigung einer Kommando-Taste eine bestimmte Bedeutung haben, ähnlich wie die von manchen Taschenrechnern bekannten "Präfix"-Tasten, die vor den eigentlichen Kommando-Tasten zu betätigen sind.

Modify-Kommando

Wird nun (nach ".. ELBUG") die Taste "MODIFY" gedrückt, so erscheint auf dem Display "MO ....". Das Programm erwartet eine Adresseingabe über das hexadezimale Tastenfeld. Nach Eingabe des letzten Adress-Digits erscheint sofort auch der Inhalt dieser Adresse auf dem Datenfeld des Display. Nun hat der Benutzer 4 Möglichkeiten:

– Er will den Inhalt dieser Adresse ändern. Dies geschieht durch Eingabe des gewünschten neuen Bytes auf der Hex-Tastatur. Nach der ersten Hex-Taste erscheint das betreffende Hex-Zeichen auf Display 1, Display 0 erlischt, Nach der 2. Hex-Taste erscheint das Hex-Zeichen auf Display 0, aber nur dann, wenn man in die betreffende Adresse auch tatsächlich etwas hineinschreiben kann. Unbeschaltete Adressbereiche (werden von der CUP als X'FF interpretiert) oder PROM-Adressen melden sich wieder mit ihrem (nicht überschreibbaren) Inhalt.

Hat man sich vertippt, so kann man beliebig oft ein anderes Byte (kein einzelnes Hex-Digit = 1/2 Byte) auf die gleiche Adresse eintasten.

– Man will den Inhalt der nächsthöheren Adresse auf die Anzeige bekommen: "UP" Taste. Natürlich auch dann, wenn der Inhalt der vorherigen Adresse nicht mit Hex-Eingaben geändert wurde. Auf diese Weise kann recht komfortabel ein Speicherbereich durchgelesen werden.

– Mit der "DOWN" - Taste ruft man den Inhalt der nächstniedrigeren Adresse ab.

– Die Modify-Routine kann nur mit NRST verlassen werden: "..ELBUG".

Mit Modify bewegt man sich immer in der gleichen Page. Nach 4 FFF folgt 4000. Auf Adr. 5000 kommt man mit NRST, MODIFY und 5000.

Run-Kommando

Hat man z.B. mit Modify ein Programm in ein RAM-Bereich geladen und will es starten, bedient man sich dieses Kommandos.
Zunächst muß mit NRST wieder "..ELBUG" auf das Display gebracht werden. Nach Betätigen der RUN-Taste erscheint "RU ...." auf dem Display. Die Startadresse des Programms (die Adresse des ersten Befehlsbytes) wird nun eingetastet. Zum eigentlichen Starten kann jede beliebige Taste (Hex oder Kommando) benutzt werden. Beim Loslassen wird "RU" auch auf Digit 0-1 des Display geschrieben und das Programm gestartet. "RU ADR RU" bleibt stehen, sofern das betreffende Programm nicht selbst auf das Display zugreift. Geschieht in dem angesprungenen Programm ein XPPC 3 (programmiert oder durch Interrupt) bevor PTR3 neu geladen ist, so erscheint "..ELBUG" auf dem Display; Zeichen dafür, daß das User-Programm verlassen wurde und die CPU auf ein neues Kommando wartet.

Hexadezimale Subtraktion

Dieses Kommando dient zum Errechnen von Displacement-Werten für

eigene Programme, als Assemblier-Hilfe.
Nach Drücken dieser Taste zeigt das Display: "SH...." = Subtraktion, hexadezimal. (Nicht "5 H....", in "7-Segment-Schrift" sehen "5" und "S" nur gleich aus).

Jetzt verlangen die Dezimalpunkte keine Adresse, sondern den Minuenden. Wenn dieser eingegeben ist, erlöschen Digit 6 und 7 und auf Digit 1 erscheint ein Minus-Zeichen. Es folgt die Eingabe des Subtrahenden. Die Differenz bekommt man durch Betätigen einer beliebigen Taste. Sie ist mit einem "=" versehen. Hier läuft das Programm in eine tote Schleife:

```
JMP:
JMP JMP 90FE
```

Diese kann nur mit NRST verlassen werden.
Negative Differenzen erhält man, so wie sie gebraucht werden: als Zweier-Komplement.

Kassetten-Routinen

Zur Ausgabe des Inahltes eines Speicherbereichs geht man so vor:
NRST "..ELBUG"
KA-Taste "CA . . . ."

Jetzt kann man mit der Modify-Taste die gewünschte Ausgabegeschwindigkeit eingegeben werden, s. Tabelle.

MO-Taste	"CA . . . . MO"
SPEED (HexKB)	"Ca00XXMO"

Wenn von dieser Möglichkeit kein Gebrauch gemacht wird, so erfolgt die Ausgabe mit 600 Baud (= Bit/sec.). Man kann also ohne "Modify Speed" fortfahren.

Beliebige Taste (außer MO, UP, DOWN)	"Ca ....ad"
Anfangsadr. des Speicherbereichs	"Caxxxx.."
Endadr. des Speicherbereichs (Hex)	"Cayyyy.."

Spätestens jetzt muß der Kassetten-Rekorder angeschlossen, auf Aufnahme eingestellt und gestartet werden. Vor dem nächsten Schritt

sollte der Rekorder bereits 1 min. laufen, damit ein Stück hohe Frequenz (= log 1 = Stopbit) aufgenommen wird. Dieses verlängerte Stopbit wird beim Laden von Kassette dringend benötigt.

DOWN nach einiger Zeit, wenn der Datenblock fertig ausgegeben ist, erscheint auf dem Display "..ELBUG"
Die CPU wartet auf das nächste Kommando.

Bei der Ausgabe wird die Anfangs- und Endadresse auf das Band geschrieben. Außerdem wird nach je 32 Datenbytes eine Checksumme (= Arithmetische Summe dieser 32 Bytes, unter Außerachtlassung des Übertrags, also nur 1 Byte) an die Kassette ausgegeben. Der letzte Datenblock kann kürzer sein als 32 Byte (ist aber trotzdem mit einer Checksumme versehen). Es ist erlaubt, Datenblocks auszugeben, welche Page-Grenzen überschreiten. Z.B. Anfangsadr. = 0FF0, Endadr. = 1654.

Laden von der Kassette:

NRST "..ELBUG"
KA-Taste "Ca . . . ."

Spätestens hier muß der Rekorder laufen und in seinem Lautsprecher das o.a. verlängerte Stopbit zu hören sein.

UP-Taste nach einer Weile, wenn der Datenblock (auch hörbar) zu Ende ist, erscheint auf dem Display wieder "..ELBUG".
Wenn irgend etwas nicht gestimmt hat – Band beschädigt, Kassetten Interface nicht in Ordnung usw. - erscheint noch während der Datenblock im Lautsprecher zu hören ist
"Ca Error" auf dem Display.

Das bedeutet, daß die selbst errechnete Checksumme während der Datenübernahme nicht mit der Checksumme vom Band übereinstimmt. Nach "Error" läuft das Programm in eine tote Schleife, die nur mit NRST verlassen werden kann. Wenn einige Wiederholungsversuche (der Fehler kann auch bei der Ausgabe an die Kassette geschehen sein) nichts bringen, müssen der Logiktester und der Lötkolben in Aktion treten.

Das auf diese Weise geladene Programm wird genau an den Adressen stehen, von wo es an die Kassette ausgegeben wurde; die Anfangs-

und Endadressen stehen ja auf dem Band! Soll ein Datenblock von der Kassette auf andere Adressen geladen werden, wird folgendermaßen vorgegangen:

NRST	"..ELBUG"
KA-Taste	"Ca.... "
Beliebige Taste (außer MO,UP,DOWN)	"Ca.... ad"
Anfangsadr.	"Caxxxx.."
Endadr.	"Cayyyy.."
Rekorder laufen lassen	
UP	

Wenn das Interface total defekt ist und ein ewiges Startbit an SIN anliegt, erscheint kein Error weil 0 + 0 = 0 und die Checksumme ja stimmt. Liegt ein dauerndes Stopbit an SIN, so kommt die Kassettenroutine erst gar nicht in Gang, obwohl der Rekorder schon längst fertig ist.

Wenn nur ein Teil eines Blocks von der Kassette geladen wird (z.B. auf Kassette sind Daten von 1F00 bis 1 FFF und der erste Teil dieses Blocks soll auf 1F20 bis 1F80 geladen werden), so erscheint nach korrekt beendeter Übernahme doch "Error", weil naturgemäß die letzte Checksumme nicht stimmt.

Bei der bisherigen Arbeitsweise wurde angenommen, daß der Datenblock mit 600 Baud an die Kassette ausgegeben wurde. Ist das nicht der Fall, so muß auch hier über Modify die tatsächliche Geschwindigkeit eingegeben werden:

NRST, KA-Taste, MO-Taste, SPEED, (beliebige Taste, Anfangsadr. Endadr.) Rekorder laufen lassen und UP-Taste.

Block-Transfer-Kommando

Während des Austestens von Programmen tritt oft die Situation auf, daß an irgendeiner Stelle noch Befehle eingefügt werden müssen, oder daß einige Befehle überflüssig sind. Um nicht noch einmal mühe- und fehlervoll das restliche Programm ab dieser Stelle eintippen zu müssen, bedient man sich dieses Kommandos.

Hanhabung:

NRST	"..ELBUG"
Transfer-Taste	"BL.... "
Anfangsadr. des Blocks	"BLxxxx.."
Endadr. des Blocks	"BLyyyy.."
Neue Anfangsadr.	"BLzzzz.."
Beliebige Taste	"..ELBUG"

Damit ist der Block umgeladen.
Ein Umladen auf eine andere Page ist möglich, aber nicht von oder auf einen Bereich, der eine Page-Grenze überschneidet. Das wäre auch nicht sinnvoll, da ein Programm an der Page-Grenze nur dann weiterläuft, wenn dort mit einem XPPC dieser Sprung ermöglicht wird. Die neue Anfangsadr. kann kleiner oder größer sein als die ursprüngliche; ein aufwärts oder abwärts-Umladen ist demnach möglich.

CPU-Register-Kommando

Dieses Kommando ist auch eine Debug-Hilfe. Damit kann an einer beliegiben Stelle eines zu testenden Programms der Inhalt der CPU-Register auf das Display gebracht werden. Das betreffende Programm wird in ein RAM-Bereich geladen. Dann:

NRST	„ .. ELBUG "	
CPU-Taste	„CP .... "	
Startadr. User-Progr.	„CPxxxx.. "	
Stopadr.(Byte nach dem Befehl der als letzter ausgeführt werden soll)	„CPyyyy.. "	
Taste A	„ CP AA "	AA = (A)
Taste E	„CP EE "	EE = (E)
Taste 5 (5 = S 7-Segm.)	„CP 55 "	= (SR)
Taste 1	„CP 1111 "	= (PTR 1)
Taste 2	„CP 2222 "	= (PTR 2)

Die Schleife zum Abfragen dieser 5 Tasten (A, E, 5, 1, 2) wiederholt sich endlos und kann nur mit NRST verlassen werden. Zu beachten ist, daß das Programm nicht läuft, wenn (PTR3) im User-Programm verändert wird, und daß nachher in der Stopadresse nicht der vorherige Inhalt sondern X,3F (= XPPC 3) steht. Dieses muß vor dem weiteren Austesten wieder in den ursprünglichen Zustand gebracht werden (mit Modify).

RAM-Organisation des LIFO-Stack

ADRESSE	REGISTER	STATUS
0FDF	AC	
0FDE	E	
0FDD	SR	
0FDC	PTR 1 L	
0FDB	PTR 1 H	
0FDA	PTR 2 L	ERSTER STATUS
0FD9	PTR 2 H	
0FD8	PTR 3 L	
0FD7	PTR 3 H	
0FD6	ROUTAD L	
0FD5	ROUTAD H	
0FD4	AC	
0FD3	E	
0FD2	SR	
0FD1	PTR 1 L	
0FD0	PTR 1 H	
0FCF	PTR 2 L	ZWEITER STATUS
0FCE	PTR 2 H	
0FCD	PTR 3 L	
0FCC	PTR 3 H	
0FCB	ROUTAD L	
0FCA	ROUTAD H	
0FC9	AC	
		DRITTER STATUS usw.

Format der Daten auf Kassette

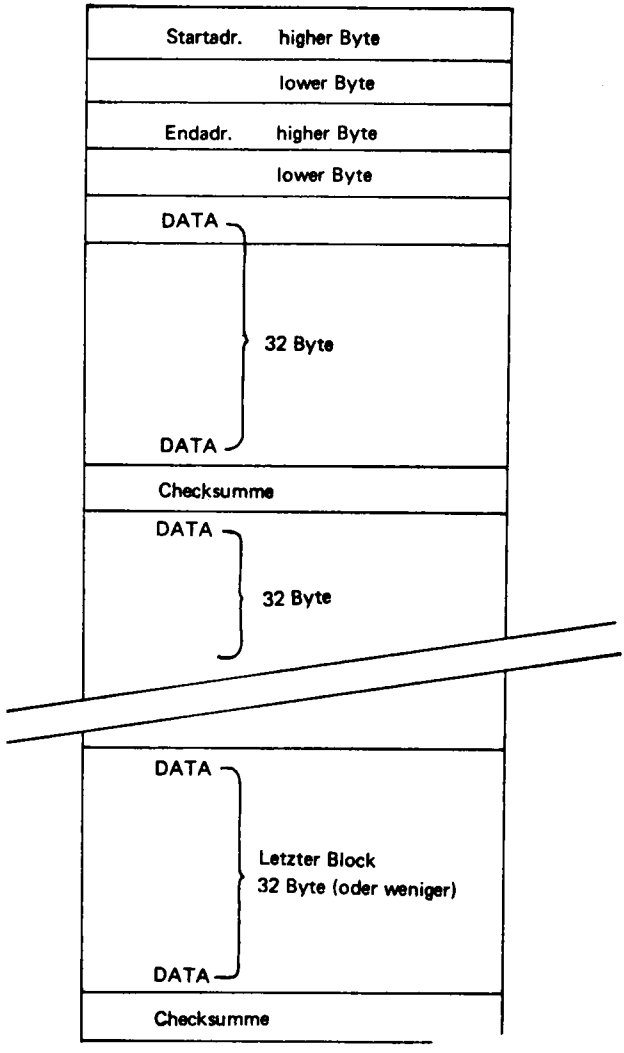


Tabelle mit den Werten zum Laden des Bytes "SPEED" für verschiedene Geschwindigkeiten der Kassetten-Routinen.

Geschwindigkeit	SPEED	
2400 Bit/sec	ØØØ2	
1200	ØØØ8	
600	ØØ15	(wird vom ELEBUG
300 (Cansas City)	ØØ2E	automatisch geladen)
110 (TTY)	ØØ85	

* Die Werten gelten für SC/MP I mit 1 MHz-Quarz oder SC/MP II mit 2 MHz-Quarz. Für SC/MP II mit 4 MHz-Quarz gelten doppelte Geschwindigkeiten oder die o.a. Werte halbiert (Vorsicht es sind Hex-Zahlen).

```
                .title          ' e l b u g '
                .page           'stack-routinen'
                .local          /abgrenzung eines '$'-adr.-bereichs
      0700      displ   = 0700   /e.a. des display
      0fff      stakpt  = 0fff   /2 byte f. aktuellen stackpointerstand
      0ffd      routad  = 0ffd   /2 byte adresse der subroutine
      0ffb      stfull  = 0ffb   /'stack-full'-flag
      0ffa      stdeep  = 0ffa   /1 byte f.einstellbare stacktiefe
      0ff9      stkeff  = 0ff9   /aktueller stack-fuellstand
      0ff8      ac      = 0ff8   /zwischenspeicher fuer (ac)
      0ff7      ptr     = 0ff7   /zwischenspeicher fuer (pointer)
      0ff5      speed   = 0ff5   /geschwindigkeit kassetten-routinen

      0fe0      stkbse  = 0fe0   /'stack-base'
      0000      .=0000
                stack:           /programm-anfang
0000  08         nop
0001  c415       ldi x'15        /kassetten-geschwindigkeit auf
0003  c8f1       st speed        /600 bit/sec voreinstellen
0005  c4e0       ldi l(stkbse)  /'ram-stackpointer' auf 'stack-
0007  c8f7       st stakpt       /base' stellen
0009  c40f       ldi h(stkbse)
000b  c8f2       st stakpt-1
000d  c400       ldi 00
000f  c8e9       st stkeff       /stack-zaehler auf 0 stellen
0011  c8e9       st stfull       /stack-full-anzeige = 0
0013  903d       jmp $1          /eigentliches 'elbug' anspringen
                pull:            /status aus stack 'ziehen'
0015  c0e9       ld stakpt       /ptr 1  mit aktuellem stackpointer-
0017  31         xpal 1          /stand laden
0018  c0e5       ld stakpt-1
001a  35         xpah 1
001b  c501       ld @1(1)        /routinen-adr. aus stack in
001d  c8de       st routad-1     /'routad' laden
001f  c501       ld @1(1)
0021  c8db       st routad
0023  c501       ld @1(1)        /ptr 3 aus stack laden
0025  37         xpah 3
0026  c501       ld @1(1)
0028  33         xpal 3
0029  c501       ld @1(1)        /ptr 2 aus stack laden
002b  36         xpah 2
002c  c501       ld @1(1)
002e  32         xpal 2
002f  c501       ld @1(1)        /(ptr 1) aus stack in zwischen-
0031  c8c4       st ptr-1        /speicher
0033  c501       ld @1(1)
0035  c8c1       st ptr
0037  c501       ld @1(1)        /s-reg. aus stack laden
0039  07         cas
003a  c501       ld @1(1)        /e-reg. aus stack laden
003c  01         xae
003d  c501       ld @1(1)        /(ac) aus stack in zwischenspeicher
003f  c8b8       st ac
```

```
0041 c0b4   ld ptr-1        /aktuellen stackpointerstand in
0043 35     xpah 1          /'stkpt' laden und  ptr 1  aus
0044 c8b9   st stakpt-1     /zwischenspeicher laden
0046 c0b0   ld ptr
0048 31     xpal 1
0049 c8b5   st stakpt
004b b8ad   dld stkeff      /stackfuellstandszeiger up to date
004d c0aa   ld ac           /ac aus zwischenspeicher laden
004f 3f     xppc 3          /ruecksprung in verlassenes programm
0050 9004   jmp push .      /notwendig  (s.text)
            $1:
0052 904d   jmp start       /'hilfs-jump'
            $2:
0054 90bf   jmp pull        /dito
            push:           /status in stack ablegen
0056 c8a1   st ac           /(ac) in zwischenspeicher ablegen
0058 c0a6   ld stakpt
005a 33     xpal 3          /(ptr 3) in zwischenspeicher und
005b c89b   st ptr          /ptr 3 als stackpointer laden
005d c0a0   ld stakpt-1
005f 37     xpah 3
0060 c895   st ptr-1
0062 c4ff   ldi l(stakpt)   /(ptr 1) in stack ablegen und
0064 31     xpal 1          /ptr 1  als rampointer laden
0065 cffc   st @-4(3)
0067 c40f   ldi h(stakpt)
0069 35     xpah 1
006a cfff   st @-1(3)
006c 01     xae             /(e) in stack ablegen
006d cb03   st 3(3)
006f 06     csa             /(sr) in stack ablegen
0070 cb02   st 2(3)
0072 c1f9   ld -7(1)        /(ac) aus zwischenspeicher in stack
0074 cb04   st 4(3)
0076 32     xpal 2          /(ptr 2) in stack ablegen
0077 cfff   st @-1(3)
0079 36     xpah 2
007a cfff   st @-1(3)
007c c1f8   ld -8(1)        /(ptr 3) aus zwischenspeicher
007e cfff   st @-1(3)       /holen, in stack ablegen
0080 c1f7   ld -9(1)
0082 cfff   st @-1(3)
0084 c1fe   ld -2(1)        /routinenadr. aus 'routad' in
0086 cfff   st @-1(3)       /stack ablegen
0088 c1fd   ld -3(1)
008a cfff   st @-1(3)       /und  ptr 3 mit routinenadr. laden,
008c 37     xpah 3          /gleichzeitig aktuellen stack-
008d c9ff   st -1(1)        /pointerstand (aus ptr 3)  in
008f c1fe   ld -2(1)        /'stkpt' ablegen.
0091 33     xpal 3
0092 c900   st 0(1)
0094 a9fa   ild -6(1)       /fuellstandszaehler -  up to date
0096 e1fb   xor -5(1)       /mit eingestellter tiefe vergleichen
0098 9c04   jnz $3
009a c4ff   ldi x'ff        /'stack-full'-flag setzen
009c c9fc   st -4(1)
            $3:
009e 3f     xppc 3          /subroutine anspringen
009f 90b3   jmp $2
```

```
                .page           'kommando-schleife'
                .local
                start:
00a1 c400       ldi l(displ)    /ptr 1 zur adressierung des
00a3 31         xpal 1          /display und keyboard laden
00a4 c407       ldi h(displ)
00a6 35         xpah 1
00a7 c4e0       ldi l(stkbse) /ptr 2 als ram-pointer laden
00a9 32         xpal 2
00aa c40f       ldi h(stkbse)
00ac 36         xpah 2
00ad c42f       ldi l(tab1)     /ptr 3  auf tabelle '..elbug'
00af 33         xpal 3
00b0 c401       ldi h(tab1)
00b2 37         xpah 3
00b3 c408       ldi 08          /zaehler laden
00b5 ca0b       st 0b(2)
                $1:
00b7 c701       ld @1(3)        / '..elbug' an display
00b9 cd01       st @1(1)
00bb ba0b       dld 0b(2)
00bd 9cf8       jnz $1          /fertig ?
00bf c40a       ldi l(ldkb)-1
00c1 ca1d       st x'1d(2)
00c3 c402       ldi h(ldkb)
00c5 ca1c       st x'1c(2)
00c7 c400       js 3(push)      /ldkb, ueber stack, anspringen
00c9 37c4
00cb 5533
00cd 3f
00ce c480       ldi x'80        / '.'  an display 3-7
00d0 cdfd       st @-3(1)
00d2 cdff       st @-1(1)
00d4 cdff       st @-1(1)
00d6 cdff       st @-1(1)
00d8 c400       ldi 00          /display 1 loeschen
00da cdff       st @-1(1)
00dc c208       ld 8(2)         /keyboard gesamtes byte, holen
00de 01         xae
00df 40         lde
00e0 e4e0       xri x'e0        / = x'e0 ?
00e2 9853       jz modify
00e4 40         lde
00e5 e4f0       xri x'f0
00e7 9c07       jnz $2          / = x'f0 ?
00e9 c401       js 3(run)
00eb 37c4
00ed a033
00ef 3f
                $2:
00f0 40         lde
00f1 e4d0       xri x'd0        / = x'd0 ?
00f3 9c07       jnz $3
00f5 c403       js 3(subtr)
00f7 37c4
00f9 ea33
00fb 3f
                $3:
00fc 40         lde
```

```
00fd e4c0    xri x'c0       / = x'c0 ?
00ff 9c07    jnz $4
0101 c402    js 3(kass)
0103 37c4
0105 f133
0107 3f
             $4:
0108 40      lde
0109 e4b0    xri x'b0       / = x'b0 ?
010b 9c07    jnz $5.
010d c405    js 3(transf)
010f 37c4
0111 4933
0113 3f
             $5:
0114 40      lde
0115 e4a0    xri x'a0       / = x'a0 ?
0117 9c88    jnz start
0119 c404    js 3(cpu)
011b 37c4
011d 3533                   /keine kommando-taste
011f 3f


             .page           'tabellen'
             tab:           / 7-segm.-code 1-f (0=3f=tab-1)
0120 06      .byte  06,x'5b,x'4f,x'66,x'6d,x'7d,07
0121 5b
0122 4f
0123 66
0124 6d
0125 7d
0126 07
0127 7f      .byte x'7f,x'6f,x'77,x'7c,x'58,x'5e,x'79,x'71
0128 6f
0129 77
012a 7c
012b 58
012c 5e
012d 79
012e 71
             tab1:          / 7-segm.-code '..elbug'
012f 00      .byte 00,x'3d,x'1c,x'7c,x'38,x'79,x'80,x'80
0130 3d
0131 1c
0132 7c
0133 38
0134 79
0135 80
0136 80
             .page           'modify-kommando'
             .local
             modify:
0137 c45c    ldi x'5c       / 'mo' an display
0139 c905    st 5(1)
013b c454    ldi x'54
013d c906    st 6(1)
013f c43e    ldi l(gethex)-1
```

```
0141 ca1d    st x'1d(2)
0143 3f      xppc 3          /'adr' von keyboard holen
            $1:
0144 c201    ld 1(2)         /ptr 3 auf 'adr' stellen
0146 33      xpal 3
0147 c202    ld 2(2)
0149 37      xpah 3
014a c300    ld 0(3)         / (adr) holen
014c ca00    st 0(2)
014e c4a0    ldi l(puthex)-1
0150 ca1d    st x'1d(2)
0152 c400    js 3(push)      / adr und (adr) auf display anzeigen
0154 37c4
0156 5533
0158 3f
0159 c40a    ldi l(ldkb)-1
015b ca1d    st x'1d(2)
015d 3f      xppc 3          /eine taste von kb holen
015e c201    ld 1(2)
0160 33      xpal 3          /ptr 3 wieder auf 'adr'
0161 c202    ld 2(2)
0163 37      xpah 3
0164 c208    ld 8(2)         /gesamtes byte resp.taste holen
0166 e480    xri x'80        / = 'up'-taste ?
0168 980a    jz $2
016a e480    xri x'80        /byte wieder herstellen
016c e490    xri x'90        / = 'down'-taste ?
016e 9c0e    jnz $4
0170 c7ff    ld e-1(3)       /ptr 3 decrementieren
0172 9002    jmp $3
            $2:
0174 c701    ld e1(3)        /ptr 3 incrementieren
            $3:
0176 33      xpal 3
0177 ca01    st 1(2)         / (ptr 3) in ram aufbewahren
0179 37      xpah 3
017a ca02    st 2(2)
017c 90c6    jmp $1
            $4:
017e c207    ld 7(2)         / 7-segm.-code fuer  erste taste
0180 c900    st 0(1)         /an display 1 geben
0182 c400    ldi 00          /display 0 loeschen
0184 c9ff    st -1(1)
0186 c209    ld 9(2)         /binaerwert der taste
0188 1e      rr              /in bit 4-7 e
0189 1e      rr
018a 1e      rr
018b 1e      rr
018c 01      xae
018d c400    js 3(push)      / 2. hex.-taste abfragen
018f 37c4
0191 5533
0193 3f
0194 c201    ld 1(2)
0196 33      xpal 3          /ptr 3 wieder auf 'adr' stellen
0197 c202    ld 2(2)
0199 37      xpah 3
019a c209    ld 9(2)
019c 58      ore             /byte zusammenstellen und
```

```
019d cb00    st 0(3)          /in 'adr' ablegen
019f 90a3    jmp $1
             .page             'run-kommando'
             .local
             run:
01a1 c450    ldi x'50         /'ru'  an display 6,7
01a3 c906    st 6(1)
01a5 c41c    ldi x'1c
01a7 c905    st 5(1)
01a9 c43e    ldi l(gethex)-1
01ab ca1d    st x'1d(2)       /startadr. von keyboard holen
01ad c400    js 3(push)
01af 37c4
01b1 5533
01b3 3f
01b4 c40a    ldi l(ldkb)-1 /auf beliebige taste warten
01b6 ca1d    st x'1d(2)
01b8 3f      xppc 3
01b9 c201    ld 1(2)
01bb 33      xpal 3           /startadr. in ptr 3
01bc c202    ld 2(2)
01be 37      xpah 3
01bf c7ff    ld @-1(3)        /startadr.-1
01c1 c450     ldi x'50
01c3 c900    st 0(1)          /'ru' an display 0,1
01c5 c41c    ldi x'1c
01c7 c9ff    st -1(1)
01c9 3f      xppc 3           /user-programm anspringen
01ca c40f    js 3(stack)      /bei ruecksprung aus user-programm
01cc 37c4
01ce ff33
01d0 3f
             .local
             .page
             ldbyte:          /routine: 1 byte von kassette holen
01d1 c215    ld x'15(2)
01d3 1c      sr               /speed:2 in ram
01d4 ca14    st x'14(2)
             $1:
01d6 c4ff    ldi x'ff
01d8 01      xae
01d9 19      sio              /stopbit ausgeben
01da 40      lde
01db 9402    jp $2            /auf startbit warten
01dd 90f7    jmp $1
             $2:
01df c4ff    ldi x'ff
01e1 01      xae
01e2 c214    ld x'14(2)       /speed/2  kopieren
01e4 ca0a    st 10(2)
             $3:
01e6 ba0a    dld 10(2)        / 1/2 bit delay
01e8 9cfc    jnz $3
01ea c408    ldi 08           /bit-zaehler laden
01ec ca08    st 8(2)
             $4:
01ee c215    ld x'15(2)       /speed kopieren
01f0 ca09    st 9(2)
01f2 c416    ldi 22           /dly 114 us (sc/mp 1)
```

```
01f4 8f00   dly 00
           $5:
01f6 ba09   dld 9(2)        /speed herunterzaehlen
01f8 9cfc   jnz $5
01fa 19     sio             /bit uebernehmen
01fb ba08   dld 8(2)
01fd 9cef   jnz $4          /8 bit uebernommen?
01ff c215   ld x'15(2)
0201 ca09   st 9(2)
           $6:
0203 ba09   dld 9(2)        /speed herunterzaehlen (1 x = 66 us)
0205 9cfc   jnz $6          / (sc/mp 1)
8207 40     lde             /fertiges byte in ac laden
0208 3f     xppc 3          /ruecksprung
0209 90c6   jmp ldbyte      /fuer erneuten ansprung
                            'routine load keyboard'
           .page
           .local
           ldkb:
020b c414   ldi l(pull)-1 /ptr 3 vorbereiten
020d 33     xpal 3
020e c400   ldi h(pull)
0210 37     xpah 3
           ldkb1:           /label zum ansprung ohne stack
0211 c40.   ldi l(displ)+1
0213 31     xpal 1
0214 c407   ldi h(displ)  /ptr 1 und ptr 2 vorbereiten
0216 35     xpah 1
0217 c4e0   ldi l(stkbse)
0219 32     xpal 2
021a c40f   ldi h(stkbse)
021c 36     xpah 2
           $1:
021d c108   ld 8(1)
021f 94fc   jp $1           /auf tastendruck warten
0221 8f1e   dly 30          /entprellzeit - ca. 30 ms
0223 c108   ld 8(1)
0225 ca08   st 8(2)         /keyboard-ausgang (gesamt) in ram
0227 d40f   ani 0f
0229 ca09   st 9(2)         /binaer-wert der taste in ram und
022b 01     xae             /in e
           $2:
022c c108   ld 8(1)
022e 9402   jp $3           /auf loslassen der taste warten
0230 90fa   jmp $2
           $3:
0232 8f1e   dly 30          /enprellzeit
0234 c41f   ldi l(tab)-1
0236 31     xpal 1
0237 c401   ldi h(tab)
0239 35     xpah 1
023a c180   ld -128(1)      /7-segm.-code holen

023c ca07   st 7(2)         /in ram ablegen
023e 3f     xppc 3          /ruecksprung

           .page            'routine gethex'
           .local
           gethex:
023f c406   ldi l(displ)+6
```

```
0241 31     xpal 1          /ptr 1 auf display 6 stellen
0242 c407   ldi h(displ)
0244 35     xpah 1
0245 c4e7   ldi l(stkbse)+7
0247 32     xpal 2          /ptr 2 auf stack-base + 7
0248 c40f   ldi h(stkbse)
024a 36     xpah 2
024b c404   ldi 04          /tastenzaehler laden
024d caf9   st -7(2).
           $1:
024f c455   ldi l(push)-1
0251 33     xpal 3
0252 c400   ldi h(push)
0254 37     xpah 3
0255 c40a   ldi l(ldkb)-1
0257 cba8   st -88(3)
0259 c402   ldi h(ldkb)
025b cba7   st -89(3)
025d 3f     xppc 3          /' ldkb' (ueber stack) anspringen
025e c4e0   ldi l(stkbse)
0260 33     xpal 3          /ptr 3 jetzt ram-pointer
0261 c40f   ldi h(stkbse)
0263 37     xpah 3
0264 c307   ld 7(3)         / 7-segm.-code holen
0266 cdff   st @-1(1)       /auf display 5 (4,3,2) schreiben
0268 c400   ldi 00
026a c9ff   st -1(1)        /alle display-stellen unterhalb
026c c9fe   st -2(1)        /loeschen
026e c9fd   st -3(1)
0270 c9fc   st -4(1)
0272 c9fb   st -5(1)
0274 c309   ld 9(3)         /binaer-wert der taste in ram-tabelle
0276 ceff   st @-1(2)
0278 bb00   dld 0(3)
027a 9cd3   jnz $1          /4 tasten?
027c c480   ldi x'80
027e c9ff   st -1(1)        / '.'  an display 0,1
0280 c9fe   st -2(1)
0282 c306   ld 6(3)         /higher byte zusammenstellen
0284 1e     rr
0285 1e     rr
0286 1e     rr
0287 1e     rr
0288 01     xae
0289 c305   ld 5(3)
028b 58     ore
028c cb02   st 2(3)
028e c304   ld 4(3)         /lower byte zusammenstellen
0290 1e     rr
0291 1e     rr
0292 1e     rr
0293 1e     rr
0294 01     xae
0295 c303   ld 3(3)
0297 58     ore
0298 cb01   st 1(3)
           jspull:          /hilfslabel zum 'trittbrettfahren'
029a c400   js 3(pull)      /ueber 'pull' zurueck ins hauptprogr
029c 37c4
```

```
Ø29e 1433
Ø2aØ 3f

                .page           'routine puthex'
                .local
                puthex:
Ø2a1 c4eØ       ldi l(stkbse)
Ø2a3 33         xpal 3          /ptr 3 als fixen ram-pointer laden
Ø2a4 c4Øf       ldi h(stkbse)
Ø2a6 37         xpah 3
Ø2a7 c4eØ       ldi l(stkbse)
Ø2a9 32         xpal 2          /ptr 2 u. 1 fuer autoindexierte
Ø2aa c4Øf       ldi h(stkbse)   /adressierung vorbereiten
Ø2ac 36         xpah 2
Ø2ad c4e3       ldi l(stkbse)+3
Ø2af 31         xpal 1
Ø2bØ c4Øf       ldi h(stkbse)
Ø2b2 35         xpah 1
Ø2b3 c4Ø3       ldi Ø3          /byte-zaehler laden
Ø2b5 cbØf       st Øf(3)
                $1:
Ø2b7 c2ØØ       ld Ø(2)         /erstes (naechstes) byte holen
Ø2b9 d4Øf       ani Øf          /bit Ø-3 alleine
Ø2bb cdØ1       st @1(1)        /in ram ablegen
Ø2bd c6Ø1       ld @1(2)        /gleiches byte wieder holen
Ø2bf 1c         sr
Ø2cØ 1c         sr              /bit 4-7
Ø2c1 1c         sr
Ø2c2 1c         sr
Ø2c3 cdØ1       st @1(1)        /in naechste ram-adresse
Ø2c5 bbØf       dld Øf(3)
Ø2c7 9cee       jnz $1          / 3 byte zerlegt?
Ø2c9 c41f       ldi l(tab)-1
Ø2cb 31         xpal 1          /ptr 1 fuer indirekte adressierung
Ø2cc c4Ø1       ldi h(tab)      /vorbereiten
Ø2ce 35         xpah 1
Ø2cf c4Ø6       ldi Ø6          /hex-zeichen-zaehler laden
Ø2d1 cbØf       st Øf(3)
                $2:
Ø2d3 c6Ø1       ld @1(2)        /erstes (naechstes) halbe byte holen
Ø2d5 Ø1         xae
Ø2d6 c18Ø       ld -128(1)      / 7-segm.-code holen

Ø2d8 caØ5       st 5(2)         /in ram zwischenspeichern
Ø2da bbØf       dld Øf(3)
Ø2dc 9cf5       jnz $2          / 6 stellen fertig?
Ø2de c4ØØ       ldi l(displ)
Ø2eØ 31         xpal 1          /ptr 1 auf display stellen
Ø2e1 c4Ø7       ldi h(displ)
Ø2e3 35         xpah 1
Ø2e4 c4Ø6       ldi Ø6          /zaehler laden
Ø2e6 cbØf       st Øf(3)
                $3:
Ø2e8 c6Ø1       ld @1(2)        / 7-segm.-code
Ø2ea cdØ1       st @1(1)        /an display Ø-5
Ø2ec bbØf       dld Øf(3)
Ø2ee 9cf8       jnz $3          /fertig?
Ø2fØ 9Øa8       jmp jspull      /ueber hilfslabel nach pull
```

```
                .page           'kassetten-routinen'
                .local
                kass:
02f2 c439       ldi x'39        /'ca' an display 6,7
02f4 c906       st 6(1)
02f6 c45f       ldi x'5f
02f8 c905       st 5(1)
02fa 01         xae
02fb 19         sio             /stopbit ausgeben
02fc c4ff       ldi x'ff
02fe ca00       st 0(2)         /flag 'adr. von kassette' setzen
0300 c400       js 3(push)      /auf taste warten
0302 37c4
0304 5533
0306 3f
0307 c45f       ldi x'5f        / 'ad' an display 0,1
0309 c900       st 0(1)
030b c45e       ldi x'5e
030d c9ff       st -1(1)
030f c208       ld 8(2)
0311 e4e0       xri x'e0        /taste 'modify' ?
0313 9c1e       jnz $1
0315 c454       ldi x'54        / 'mo' an display 0,1
0317 c900       st 0(1)
0319 c45c       ldi x'5c
031b c9ff       st -1(1)
031d c43e       ldi l(gethex)-1
031f ca1d       st x'1d(2)
0321 3f         xppc 3          / 'speed' von keyboard holen
0322 c201       ld 1(2)
0324 ca15       st x'15(2)
0326 c40a       ldi l(ldkb)-1
0328 ca1d       st x'1d(2)
032a 3f         xppc 3          /auf taste warten
032b c45f       ldi x'5f
032d c900       st 0(1)
032f c45e       ldi x'5e        / 'ad' an display 0,1
0331 c9ff       st -1(1)
                $1:
0333 c208       ld 8(2)
0335 e480       xri x'80        /taste 'up' ?
0337 982c       jz kassup
0339 c43e       ldi l(gethex)-1
033b ca1d       st x'1d(2)
033d 3f         xppc 3          /anfangsadresse von kb holen
033e c201       ld 1(2)
0340 ca0b       st 0b(2)
0342 c202       ld 2(2)
0344 ca0c       st 0c(2)
0346 3f         xppc 3          /end-adrese holen
0347 c40a       ldi l(ldkb)-1
0349 ca1d       st x'1d(2)
034b 3f         xppc 3          /auf taste warten
034c c208       ld 8(2)
034e e480       xri x'80        /taste 'up' ?
0350 9c04       jnz $2
0352 ca00       st 0(2)         /flag 'adressen von kb' setzen
0354 900f       jmp kassup
                $2:
```

```
0356 e480    xri x'80
0358 e490    xri x'90       /taste 'down' ?
035a 9802    jz $3
035c 9050    jmp jsstak     /wenn andere taste: '..elbug'
             $3:
035e c404    js 3(kassdo)
0360 37c4
0362 e333
0364 3f

             kassup:        /routine: laden von kassette in ram
              local
0365 c41c    ldi x'1c       /'up' an display 0,1
0367 c900    st 0(1)
0369 c473    ldi x'73
036b c9ff    st -1(1)
036d c4d0    ldi l(ldbyte)-1
036f 33      xpal 3         /ptr 3 vorbereiten
0370 c401    ldi h(ldbyte)
0372 37      xpah 3
0373 c200    ld 0(2)
0375 980e    jz $1          /lade-adresse von kassette?
0377 3f      xppc 3         /ladeadresse, higher byte,
0378 ca0c    st 0c(2)
037a 3f      xppc 3         /lower byte,
037b ca0b    st 0b(2)
037d 3f      xppc 3         /stop-adresse, higher byte,
037e ca02    st 2(2)
0380 3f      xppc 3         /lower byte von kassette holen
0381 ca01    st 1(2)
0383 9004    jmp $2
             $1:
0385 3f      xppc 3         / 4 byte von kassette ignorieren
0386 3f      xppc 3
0387 3f      xppc 3
0388 3f      xppc 3
             $2:
0389 c420    ldi 32         /block-laengenzaehler laden
038b ca05    st 5(2)
038d c400    ldi 00
038f ca06    st 6(2)        /ram-byte fuer checksumme loeschen
0391 02      ccl
             $3:
0392 c20b    ld 0b(2)       /ptr 1 auf erste (naechste) lade-
0394 31      xpal 1         /adresse stellen
0395 c20c    ld 0c(2)
0397 35      xpah 1
0398 3f      xppc 3         /daten-byte von kassette holen und
0399 c900    st 0(1)        /in lade-adresse ablegen
039b f206    add 6(2)       /check-summe bilden
039d ca06    st 6(2)
039f 35      xpah 1
03a0 e202    xor 2(2)
03a2 9c11    jnz $4         /stopadresse erreicht?
03a4 31      xpal 1
03a5 e201    xor 1(2)
03a7 9c0c    jnz $4
03a9 3f      xppc 3         /checksumme von kassette holen
03aa e206    xor 6(2)       /mit eigener checksumme identisch?
```

```
03ac 9c21    jnz error
             jsstak:          /hilfs-label (befehls-byte sparen)
03ae c40f    js 3(stack)      /ruecksprung '..elbug'
03b0 37c4
03b2 ff33
03b4 3f
             $4:
03b5 06      csa              / cy/l (von checksumme) in e retten
03b6 01      xae
03b7 02      ccl
03b8 c20b    ld 0b(2)
03ba f401    adi 01           /ptr 1 ueber page-grenze hinweg,
03bc ca0b    st 0b(2)         /incrementieren
03be c20c    ld 0c(2)
03c0 f400    adi 00           /wegen uebertrag (cy/l)
03c2 ca0c    st 0c(2)
03c4 40      lde              / cy/l von checksumme zurueck in sr
03c5 07      cas
03c6 ba05    dld 5(2)         /block-laenge erreicht?
03c8 9cc8    jnz $3
03ca 3f      xppc 3           /checksumme von kassette holen
03cb e206    xor 6(2)         /uebereinstimmung?
03cd 98ba    jz $2
             error:           /'error' an display
03cf c401    ldi l(displ)+1
03d1 31      xpal 1
03d2 c407    ldi h(displ)
03d4 35      xpah 1
03d5 c400    ldi 00
03d7 c904    st 4(1)
03d9 c479    ldi x'79
03db c903    st 3(1)
03dd c450    ldi x'50
03df c902    st 2(1)
03e1 c901    st 1(1)
03e3 c9ff    st -1(1)
03e5 c45c    ldi x'5c
03e7 c900    st 0(1)
             $5:
03e9 90fe    jmp $5           /tote schleife
             .page             'subtraktion'
             .local
             subtr:
03eb c46d    ldi x'6d         / 'sh' an display 6,7
03ed c906    st 6(1)
03ef c476    ldi x'76
03f1 c905    st 5(1)
03f3 c43e    ldi l(gethex)-1
03f5 ca1d    st x'1d(2)
03f7 c400    js 3(push)       /minuend holen
03f9 37c4
03fb 5533
03fd 3f
03fe c440    ldi x'40         / '-' an display 1
0400 c900    st 0(1)
0402 c400    ldi 00
0404 c9ff    st -1(1)
0406 c906    st 6(1)
0408 c905    st 5(1)
```

```
040a c202   ld 2(2)         /minuend in ram ablegen
040c ca14   st x'14(2)
040e c201   ld 1(2)
0410 ca13   st x'13(2)
0412 3f     xppc 3          /subtrahend holen
0413 03     scl             /wegen 2-er komplement
0414 c213   ld x'13(2)
0416 fa01   cad 1(2)        /subtraktion
0418 ca01   st 1(2)
041a c214   ld x'14(2)
041c fa02   cad 2(2)
041e ca02   st 2(2)
0420 c40a   ldi l(ldkb)-1
0422 ca1d   st x'1d(2)
0424 3f     xppc 3          /auf beliebige taste warten
0425 c4a0   ldi l(puthex)-1
0427 ca1d   st x'1d(2)
0429 3f     xppc 3          /differenz an display
042a c400   ldi 00
042c c9ff   st -1(1)
042e c900   st 0(1)
0430 c448   ldi x'48        / '=' an display 6
0432 c905   st 5(1)
            $1:
0434 90fe   jmp $1          /tote schleife
            .page           'cpu-register'
            .local
            cpu:
0436 c439   ldi x'39
0438 c906   st 6(1)         / 'cp' an display
043a c473   ldi x'73
043c c905   st 5(1)
043e c43e   ldi l(gethex)-1
0440 ca1d   st x'1d(2)
0442 c400   js 3(push)      /startadresse user-programm holen
0444 37c4
0446 5533
0448 3f
0449 c201   ld 1(2)
044b ca0e   st 0e(2)        /startadresse umladen
044d c202   ld 2(2)
044f ca0d   st 0d(2)
0451 3f     xppc 3          /stopadresse holen
0452 c201   ld 1(2)
0454 31     xpal 1          /ptr 1  auf stopadresse
0455 c202   ld 2(2)
0457 35     xpah 1
0458 c43f   ldi x'3f        /xppc 3 (=x'3f) in stopadresse
045a c900   st 0(1)
045c c471   ldi l($1)-1     / '$1'  in routad laden
045e ca1d   st x'1d(2)
0460 c404   ldi h($1)
0462 ca1c   st x'1c(2)
0464 c20e   ld 0e(2)        /ptr 2 zum ansprung des user-
0466 01     xae             /programms vorbereiten
0467 c20d   ld 0d(2)
0469 36     xpah 2
046a 40     lde
046b 32     xpal 2
```

```
046c c5ff    ld @-1(2)
046e c455    ldi l(push)-1 /ptr 3  fuer den ruecksprung
0470 33      xpal 3        /nach 'push' vorbereiten
0471 3e      xppc 2        /user-programm-ansprung
             $1:
0472 c4e0    ldi l(stkbse) /ptr 2 wieder als ram-pointer laden
0474 32      xpal 2
0475 c40f    ldi h(stkbse)
0477 36      xpah 2
0478 c4d5    ldi l(stkbse)-11
047a ca1f    st x'1f(2)    /'stkpt' um eine etage verfaelschen
047c c40a    ldi l(ldkb)-1
047e ca1d    st x'1d(2)
0480 c402    ldi h(ldkb)
0482 ca1c    st x'1c(2)
0484 c400    js 3(push)    /tastendruck abwarten
0486 37c4
0488 5533
048a 3f
048b c208    ld 8(2)       /keyboard-byte in e
048d 01      xae
048e c4a0    ldi l(puthex)-1
0490 ca1d    st x'1d(2)    /'routad' mit puthex-1 laden
0492 c401    ldi l(displ)+1
0494 31      xpal 1        /ptr 1 wieder auf display
0495 c407    ldi h(displ)
0497 35      xpah 1
0498 40      lde
0499 e4fa    xri x'fa      /taste 'a' ?
049b 9816    jz $2
049d 40      lde
049e e4fe    xri x'fe      /taste 'e' ?
04a0 9815    jz $3
04a2 40      lde
04a3 e4f5    xri x'f5      /taste '5' ?  (fuer sr)
04a5 9814    jz $4
04a7 40      lde
04a8 e4f1    xri x'f1      /taste '1' ?
04aa 9813    jz $5
04ac 40      lde
04ad e4f2    xri x'f2      /taste '2' ?
04af 9815    jz $6
             $11:
04b1 90bf    jmp $1
             $2:
04b3 c2ff    ld -1(2)      / (ac) aus stack holen
04b5 901c    jmp $8
             $3:
04b7 c2fe    ld -2(2)      / (e) aus stack holen
04b9 9018    jmp $8
             $4:
04bb c2fd    ld -3(2)      / (sr) aus stack holen
04bd 9014    jmp $8
             $5:
04bf c2fc    ld -4(2)
04c1 01      xae
04c2 c2fb    ld -5(2)      / (ptr 1) aus stack holen
04c4 9005    jmp $7
             $6:
```

```
Ø4c6 c2fa    ld -6(2)       / (ptr 2) aus stack holen
Ø4c8 Ø1      xae
Ø4c9 c2f9    ld -7(2)
             $7:
Ø4cb caØ2    st 2(2)        /register-inhalte fuer puthex
Ø4cd 4Ø      lde            /umladen
Ø4ce caØ1    st 1(2)
Ø4dØ 3f      xppc 3         /puthex-ansprung
Ø4d1 9ØØ9    jmp $9
             $8:
Ø4d3 caØ1    st 1(2)        /register-inhalte umladen
Ø4d5 3f      xppc 3         /puthex-ansprung
Ø4d6 c4ØØ    ldi ØØ
Ø4d8 c9Ø3    st 3(1)        /restliches display loeschen
Ø4da c9Ø4    st 4(1)
             $9:
Ø4dc c4ØØ    ldi ØØ
Ø4de c9ØØ    st Ø(1)
Ø4eØ c9ff    st -1(1)
Ø4e2 3Øcd    jmp $11        /sprung nach $1 (ueber $11)
             .page          'ausgabe an kassette'
             .local
             kassdo:
Ø4e4 c45e    ldi x'5e
Ø4e6 c9ØØ    st Ø(1)        / 'do' an display Ø,1
Ø4e8 c45c    ldi x'5c
Ø4ea c9ff    st -1(1)
Ø4ec c2Øb    ld Øb(2)       /ptr 1  auf anfangs-lade-adresse
Ø4ee 31      xpal 1
Ø4ef c2Øc    ld Øc(2)
Ø4f1 35      xpah 1
Ø4f2 c4d7    ldi l(bytout)-1
Ø4f4 33      xpal 3         /ptr 3 auf byte-ausgaberoutine
Ø4f5 c4Ø5    ldi h(bytout)
Ø4f7 37      xpah 3
Ø4f8 c2Øc    ld Øc(2)       /higher byte anfangs-ladeadresse
Ø4fa 3f      xppc 3         /an kassette
Ø4fb c2Øb    ld Øb(2)
Ø4fd 3f      xppc 3         /lower byte
Ø4fe c2Ø2    ld 2(2)
Ø5ØØ 3f      xppc 3         /higher byte end-adresse
Ø5Ø1 c2Ø1    ld 1(2)
Ø5Ø3 3f      xppc 3         /lower byte
             $1:
Ø5Ø4 c42Ø    ldi 32         /block-laengen-zaehler laden
Ø5Ø6 caØ5    st 5(2)
Ø5Ø8 c4ØØ    ldi ØØ
Ø5Øa caØ6    st 6(2)        /ram-byte fuer checksumme loeschen
Ø5Øc Ø2      ccl
             $2:
Ø5Ød c1ØØ    ld Ø(1)        /erstes (naechstes) datenbyte holen
Ø5Øf Ø1      xae
Ø51Ø c2Ø6    ld 6(2)
Ø512 7Ø      ade            /checksumme bilden
Ø513 caØ6    st 6(2)
Ø515 4Ø      lde
Ø516 3f      xppc 3         /datenbyte an kassette ausgeben
Ø517 35      xpah 1
Ø518 e2Ø2    xor 2(2)       /ptr 1  =  end-adresse ?
```

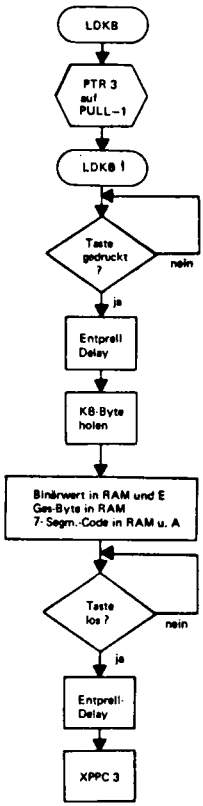
```
051a 01      xae
051b 40      lde
051c e202    xor 2(2)
051e 35      xpah 1
051f 40      lde
0520 9c08    jnz $3
0522 31      xpal 1
0523 e201    xor 1(2)
0525 9819    jz $4
0527 e201    xor 1(2)
0529 31      xpal 1
             $3:
052a 06      csa             / cy/l (von checksumme) in e retten
052b 01      xae
052c 02      ccl
052d 31      xpal 1          /ptr 1 ueber page-grenze
052e f401    adi 01          /hinweg incrementieren
0530 31      xpal 1
0531 35      xpah 1
0532 f400    adi 00
0534 35      xpah 1
0535 40      lde
0536 07      cas             / cy/l zurueck
0537 ba05    dld 5(2)        /block fertig ?
0539 9cd2    jnz $2
053b c206    ld 6(2)
053d 3f      xppc 3          /checksumme an kassette
053e 90c4    jmp $1
             $4:
0540 c206    ld 6(2)
0542 3f      xppc 3          /checksumme an kassette
             jsstac:
0543 c40f    js 3(stack)     /ruecksprung: '..elbug'
0545 37c4
0547 ff33
0549 3f

             .page
             .local           'block-transfer-kommando'
             transf:
054a c47c    ldi x'7c        / 'bl' an display 6,7
054c c906    st 6(1)
054e c438    ldi x'38
0550 c905    st 5(1)
0552 c43e    ldi l(gethex)-1
0554 ca1d    st x'1d(2)
0556 c400    js 3(push)      /block-anfangs-adresse holen (ba)
0558 37c4
055a 5533
055c 3f
055d c201    ld 1(2)
055f ca10    st x'10(2)
0561 c202    ld 2(2)
0563 ca0f    st 0f(2)
0565 3f      xppc 3          /block-ende-adresse holen (be)
0566 c201    ld 1(2)
0568 ca0e    st 0e(2)
056a c202    ld 2(2)
056c ca0d    st 0d(2)
056e 3f      xppc 3          /neue block-anfangs-adresse holen
```

```
 856f c40a   ldi l(ldkb)-1 / = 'nba'
0571 ca1d     st x'1d(2)
0573 3f       xppc 3          /auf beliebige taste wartenk
0574 03       scl
0575 c201     ld 1(2)         /subtraktion: nba - ba
0577 fa10     cad x'10(2)
0579 ca0c     st 0c(2)
057b c202     ld 2(2)
057d fa0f     cad 0f(2)
057f ca0b     st 0b(2)
0581 9429     jp $3           /differenz = positiv?
0583 c210     ld x'10(2)
0585 31       xpal 1          /ptr 1 auf 'ba'
0586 c20f     ld 0f(2)
0588 35       xpah 1
0589 c201     ld 1(2)
058b 33       xpal 3          /ptr 3 auf 'nba'
058c c202     ld 2(2)
058e 37       xpah 3
              $1:
058f c501     ld @1(1)
0591 cf01     st @1(3)        /block umladen
0593 c5ff     ld @-1(1)       /ptr 1 -1
0595 31       xpal 1
0596 e20e     xor 0e(2)       / = be?
0598 01       xae
0599 40       lde
059a e20e     xor 0e(2)       / (ptr 1 l) wieder herstellen
059c 31       xpal 1
059d 40       lde
059e 9c08     jnz $2
05a0 35       xpah 1
05a1 e20d     xor 0d(2)
              $11:            /hilfslabel
05a3 989e     jz jsstac       /ruecksprung '..elbug'
05a5 e20d     xor 0d(2)       / (ptr 1 h) wieder herstellen
05a7 35       xpah 1
              $2:
05a8 c501     ld @1(1)        /ptr 1 +1
05aa 90e3     jmp $1
              $3:
05ac c20e     ld 0e(2)        /ptr 1 auf be
05ae 31       xpal 1
05af c20d     ld 0d(2)
05b1 35       xpah 1
05b2 c501     ld @1(1)        /ptr 1 +1
05b4 03       scl
05b5 c20e     ld 0e(2)        /ptr 3 auf  be + nba - ba + 1
05b7 f20c     add 0c(2)
05b9 33       xpal 3
05ba c20d     ld 0d(2)
05bc f20b     add 0b(2)
05be 37       xpah 3
              $4:
05bf c5ff     ld  @-1(1)      /block umladen
05c1 cfff     st @-1(3)
05c3 31       xpal 1
05c4 e210     xor x'10(2)     /ptr 1 = ba ?
05c6 01       xae
```

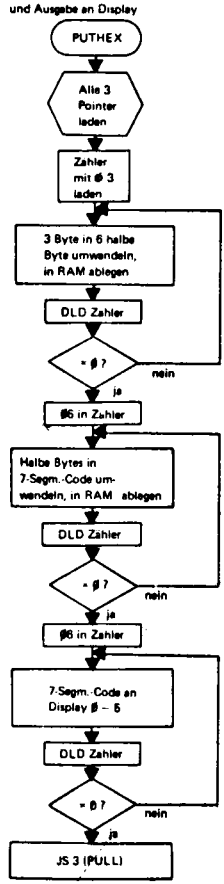
```
Ø5c7 4Ø      lde
Ø5c8 e21Ø    xor x'1Ø(2)
Ø5ca 31      xpal 1
Ø5cb 4Ø      lde
Ø5cc 9cf1    jnz $4
Ø5ce 35      xpah 1
Ø5cf e2Øf    xor Øf(2)
Ø5d1 98dØ    jz $11
Ø5d3 e2Øf    xor Øf(2)
Ø5d5 35      xpah 1
Ø5d6 9Øe7    jmp $4
             .local
             bytout:         / 1 byte an kassette ausgeben
Ø5d8 caØ7    st 7(2)         /byte in ram aufbewahren
Ø5da c4Øb    ldi 11
Ø5dc caØ8    st 8(2)         /bit-zaehler laden
Ø5de c4ØØ    ldi ØØ
Ø5eØ Ø1      xae
Ø5e1 19      sio             /startbit ausgeben
Ø5e2 Ø1      xae             /toter befehl
Ø5e3 ba2Ø    dld x'2Ø(2)     /toter befehl zum zeitausgleich
Ø5e5 c2Ø7    ld 7(2)
Ø5e7 Ø1      xae             /byte in e
             $1:
Ø5e8 c4Øb    ldi 11
Ø5ea 8fØØ    dly ØØ          /dly 7Ø us (sc/mp 1)
Ø5ec c215    ld x'15(2)      /'speed' kopieren
Ø5ee caØ9    st 9(2)
             $2:
Ø5fØ baØ9    dld 9(2)        /speed decrementieren
Ø5f2 9cfc    jnz $2
Ø5f4 19      sio             /bitausgabe
Ø5f5 4Ø      lde
Ø5f6 dc8Ø    ori x'8Ø        /stopbit an byte an fuegen
Ø5f8 Ø1      xae
Ø5f9 baØ8    dld 8(2)
Ø5fb 9ceb    jnz $1          /wenn bitzaehler = Ø ,weiter
Ø5fd 3f      xppc 3          /ruecksprung
Ø5fe 9Ød8    jmp bytout      /fuer erneuten ansprung
             .end
```

4. Flußdiagramme

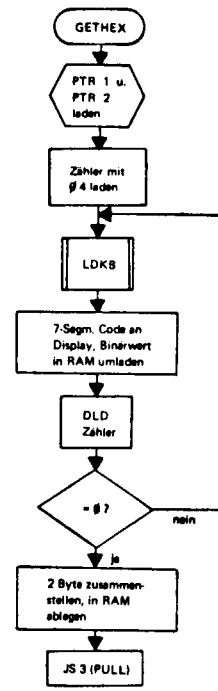
Routine zum Abfragen des Keyboard

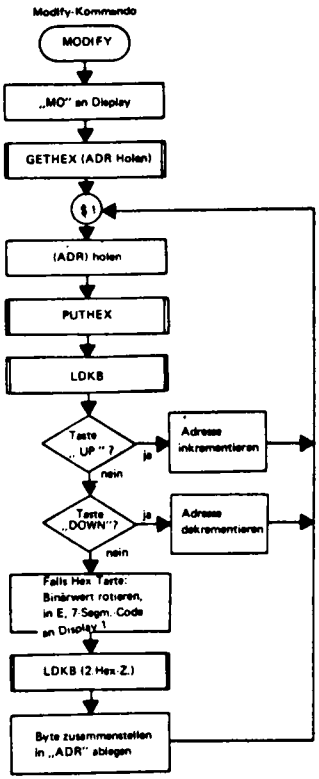


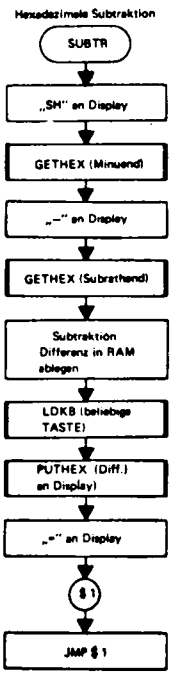
Routine zum Umwandeln von 3 binären Bytes in 6 Hex-Zeichen und Ausgabe an Display

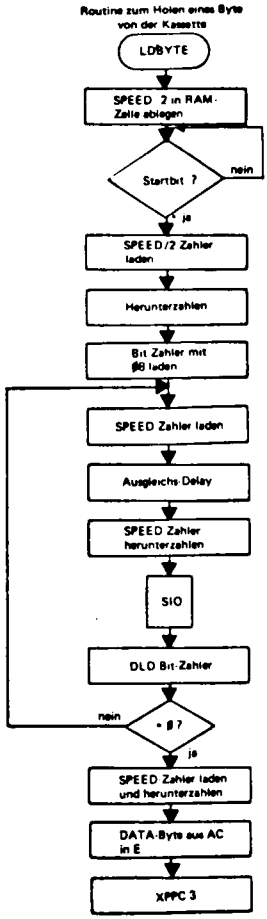


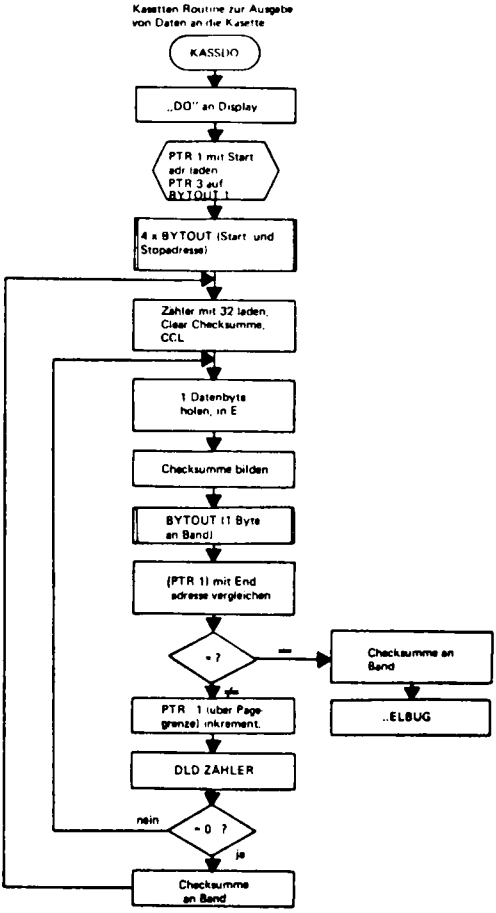
Routine zum Umwandeln von 4 Hex-Zeichen in 2 binäre Byte

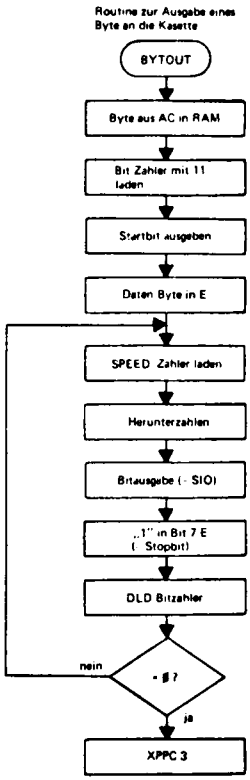


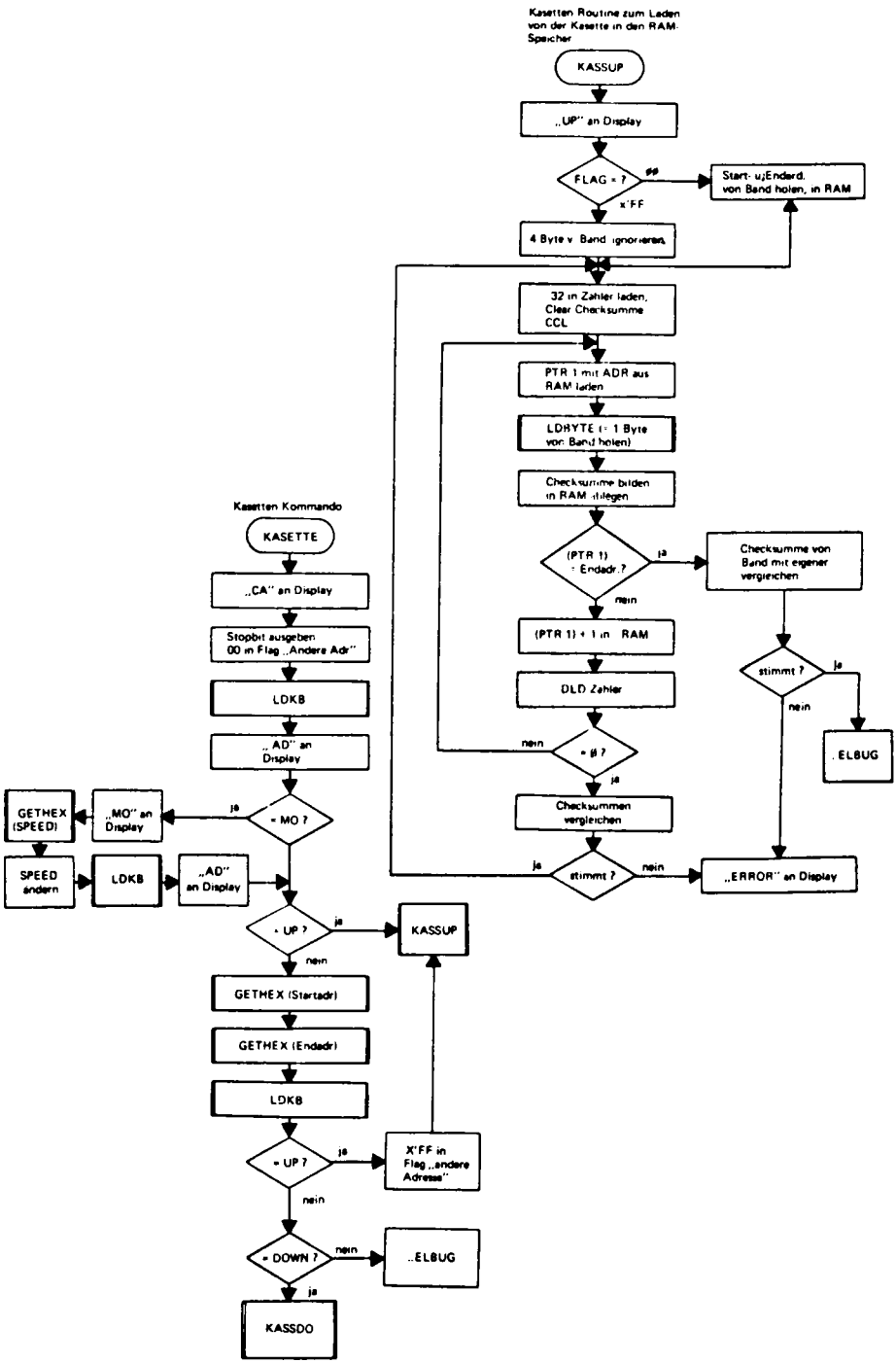

Modify-Kommando
MODIFY
„MO" an Display
GETHEX (ADR Holen)
$1
(ADR) holen
PUTHEX
LDKB
Taste „UP" ?
ja
Adresse inkrementieren
nein
Taste „DOWN"?
ja
Adresse dekrementieren
nein
Falls Hex Taste: Binärwert rotieren, in E, 7-Segm.-Code an Display
LDKB (2 Hex-Z.)
Byte zusammenstellen in „ADR" ablegen

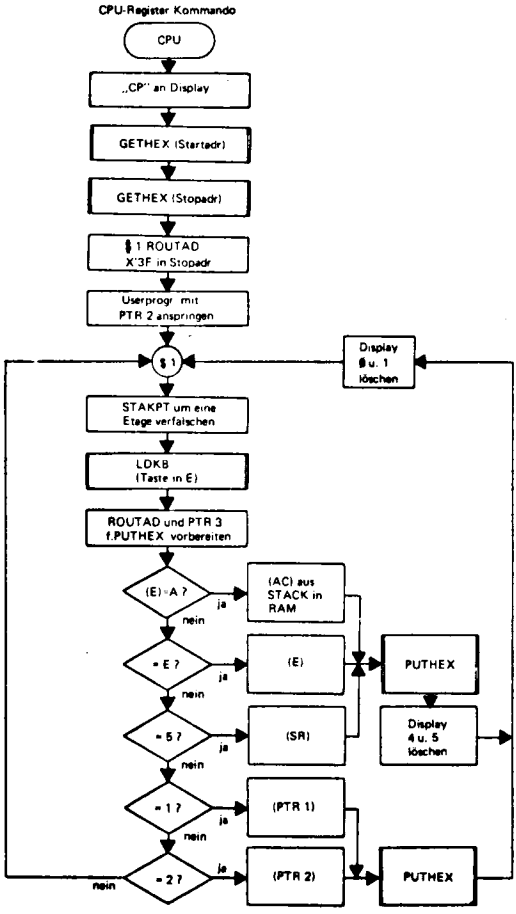

Hexadezimale Subtraktion
SUBTR
„SH" an Display
GETHEX (Minuend)
„–" an Display
GETHEX (Subtrahend)
Subtraktion Differenz in RAM ablegen
LDKB (beliebige TASTE)
PUTHEX (Diff.) an Display
„=" an Display
$1
JMP $1

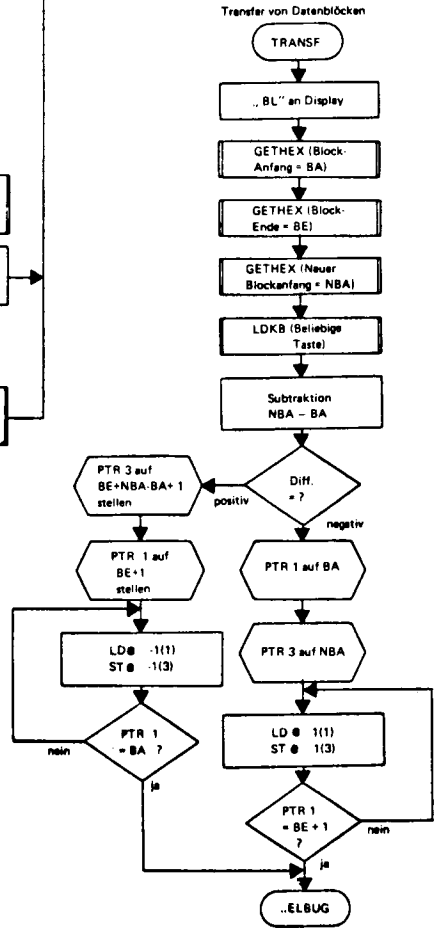

Routine zum Holen eines Byte von der Kassette
LDBYTE
SPEED 2 in RAM-Zelle ablegen
Startbit ?
nein
ja
SPEED/2 Zähler laden
Herunterzählen
Bit Zähler mit $8 laden
SPEED Zähler laden
Ausgleichs-Delay
SPEED Zähler herunterzählen
SIO
DLD Bit-Zähler
nein
= $7
ja
SPEED Zähler laden und herunterzählen
DATA-Byte aus AC in E
XPPC 3


Kassetten Routine zur Ausgabe von Daten an die Kasette
KASSDO
„DO" an Display
PTR 1 mit Start adr laden PTR 3 auf BYTOUT 1
4 x BYTOUT (Start und Stopadresse)
Zähler mit 32 laden, Clear Checksumme, CCL
1 Datenbyte holen, in E
Checksumme bilden
BYTOUT (1 Byte an Band)
(PTR 1) mit End adresse vergleichen
= ?
=
Checksumme an Band
..ELBUG
≠
PTR 1 (über Page grenze) inkrement.
DLD ZÄHLER
nein
= 0 ?
ja
Checksumme an Band


Routine zur Ausgabe eines Byte an die Kasette
BYTOUT
Byte aus AC in RAM
Bit Zähler mit 11 laden
Startbit ausgeben
Daten Byte in E
SPEED Zähler laden
Herunterzählen
Bitausgabe (→ SIO)
„1" in Bit 7 E (= Stopbit)
DLD Bitzähler
nein
= 0 ?
ja
XPPC 3


Kassetten Routine zum Laden von der Kassette in den RAM-Speicher
KASSUP
„UP" an Display
FLAG = ?
$99
Start- u. Enderd. von Band holen, in RAM
x'FF
4 Byte v. Band ignorieren
32 in Zähler laden, Clear Checksumme CCL
PTR 1 mit ADR aus RAM laden
LDBYTE (= 1 Byte von Band holen)
Checksumme bilden in RAM ablegen
(PTR 1) = Endadr.?
ja
Checksumme von Band mit eigener vergleichen
nein
(PTR 1) + 1 in RAM
DLD Zähler
= 0 ?
nein
ja
Checksummen vergleichen
stimmt ?
ja
nein
„ERROR" an Display
stimmt ?
ja
nein
„ELBUG
Kassetten Kommando
KASETTE
„CA" an Display
Stopbit ausgeben 00 in Flag „Andere Adr"
LDKB
„AD" an Display
= MO ?
ja
„MO" an Display
GETHEX (SPEED)
SPEED ändern
LDKB
„AD" an Display
= UP ?
ja
KASSUP
nein
GETHEX (Startadr)
GETHEX (Endadr)
LDKB
= UP ?
ja
X'FF in Flag „andere Adresse"
nein
= DOWN ?
nein
„ELBUG
ja
KASSDO


CPU-Register Kommando
CPU
„CP" an Display
GETHEX (Startadr)
GETHEX (Stopadr)
§1 ROUTAD
X'3F in Stopadr
Userprogr. mit
PTR 2 anspringen
§1
Display
§ u. 1
löschen
STAKPT um eine
Etage verfalschen
LDKB
(Taste in E)
ROUTAD und PTR 3
f.PUTHEX vorbereiten
(E)=A ?
ja
nein
(AC) aus
STACK in
RAM
= E ?
(E)
= 5 ?
(SR)
= 1 ?
(PTR 1)
= 2 ?
(PTR 2)
PUTHEX
Display
4 u. 5
löschen
PUTHEX


Transfer von Datenblöcken
TRANSF
„BL" an Display
GETHEX (Block-
Anfang = BA)
GETHEX (Block-
Ende = BE)
GETHEX (Neuer
Blockanfang = NBA)
LDKB (Beliebige
Taste)
Subtraktion
NBA – BA
Diff.
= ?
positiv
negativ
PTR 3 auf
BE+NBA-BA+1
stellen
PTR 1 auf
BE+1
stellen
PTR 1 auf BA
PTR 3 auf NBA
LD@ -1(1)
ST@ -1(3)
PTR 1
= BA ?
nein
ja
LD@ 1(1)
ST@ 1(3)
PTR 1
= BE + 1
?
nein
ja
..ELBUG

5. Erläuterungen zu den einzelnen Kommandos und Routinen

Die Reihenfolge der Kommandos und Subroutinen innerhalb des ELBUG scheint zunächst etwas kunterbunt gemischt: dies hat aber einen bestimmten Zweck.

- Innerhalb des ersten K (OOOO - O3FF) sind alle Subroutinen und die Kommandos MODIFY RUN und KASSUP (= Laden eines RAM-Bereichs von der Kassette) untergebracht. Dadurch ist eine hardwaremäßige Anwender-Konfiguration, bestehend aus Hex-I/O-Platine, CPU-Platine, einer 4k RAM-Platine und Kassetten-Interface möglich. Der Adressdekoder(von der Input-Unit-Platine) müßte noch auf einer kleinen Wire-Wrap-Platine (und eventuelle Interfaces) aufgebaut werden. Mit dieser „ Minimal-Konfiguration " ließe sich z.B. eine Maschinen-Steuerung realisieren. Auf dem „ großen System " wird das notwendige Programm entwickelt getestet,auf die Kassette geschrieben und dann „ vor Ort " in das kleine System geladen und mit RUN gestartet,kleine Änderungen wären mit MODIFY noch möglich.

- Die Stack-Routinen *müssen* sich nahe am Anfang des Programms befinden, weil diese zum Teil die PC-relative Adressierung benötigen. Durch den Page-Mechanismus wird das RAM von Adr. OFOO damit erreicht, so als ob es sich direkt unterhalb von Adr. OOOO befinden würde.

- Dadurch, daß die Subroutinen LDKB, GETHEX und PUTHEX innerhalb des gleichen 1/4-Bereichs liegen (higher Byte der Adr. identisch = O2) können etliche Befehlsbyte eingespart werden. Näheres s. weiter unten.

- Kommandos,welche mit einem Rücksprung in die Kommandoschleife beendet sind, wurden nahe zusammengelegt, um mit einem kurzen Sprung (PC-relativ) zum langen Sprung (PTR-relativ) des Nachbarn zu gelangen. Dieses „ Trittbrett-Fahren " spart wieder einige Befehlsbyte.
  Die ersten Befehle stellen eine sogenannte „ *Initialize* "-Routine dar (bis „ PULL "). Hier werden einige RAM-Byte definiert geladen.
  Anschließend wird über einen Hilfsjump die Kommando-Schleife angesprungen. Die Stack-Routinen werden dabei übersprungen.

  Die *Kommando-Schleife* hat folgende Aufgaben:

  - PTR 1 und PTR 2 laden
  - ..ELBUG auf das Display schreiben (das Programm „ meldet sich")
  - die Keyboard-Routine anspringen und je nach gedrückter Taste
  - die entsprechende Kommando-Routine anspringen.

  Bei Verlassen der Kommandoschleife enthält PTR 1 die EA des Display (Digit 1), PTR 2 ist als RAM-Pointer geladen (STKBSE) und auf dem Display steht „ ..... " (nicht sichtbar, weil unmittelbar danach das betreffende Kommando auf das Kommando-Feld geschrieben wird.

  Die *Stack-Routinen* des ELBUG stellen einen Software-LIFO-Stack mit Abspreichern aller CPU-Register-Inhalte, außer PC, dar, der für programmiertes oder asynchrones (INTERRUPT) Anspringen von Subroutinen, sequentiell oder geschachtelt (nested) verwendbar ist. Für die Funktion des ELBUG wäre diese Komplexität nicht erforderlich gewesen, ist aber für Anwender-Programme, besonders bei Interrupt-Verarbeitung sehr nützlich.
  Um diese Aufgaben zu erfüllen, ist zunächst ein RAM notwendig. Die LIFO-Funktion wird nicht so erfüllt, daß die aufeinanderfolgend abgespeicherten STATEN im RAM hinunter und hinauf geschoben werden, sondern ein Pointer wird als STACK-POINTER benutzt. Er zeigt jeweils auf die RAM-Adr. des zuletzt abgespeicherten STATUS. Damit ist aber (die nicht sehr üppige) Anzahl der Index-Register des SC/MP bereits auf 2 zusammengeschmolzen.

R A M – T A B E L L E

ADR	STACK	LDKB	GETHEX	PUTHEX	MODIFY	SUBTR	KASS	CPU	TRANSF
0FFF	STAKPT,lower								
0FFE	STAKPT,higher								
0FFD	ROUTAD,lower								
0FFC	ROUTAD,higher								
0FFB	STFULL								
0FFA	STDEEP								
0FF9	STKEFF								
0FF8	AC								
0FF7	PTR,lower								
0FF6	PTR,higher								
0FF5	SPEED						SPEED		
0FF4						Minuend,high.	0,5 SPEED		
0FF3						Minuend,lower			
0FF2									
0FF1									
0FF0									BA,lower
0FEF				Zähler					BA,higher
0FEE				}				Start,lower	BE,lower
0FED				}				Start,higher	BE,higher
0FEC				} 7-Segm-			Start,higher		Diff,lower
0FEB				} Code	Zähler		Start,lower		Diff,higher
0FEA				}	1/2 Byte		0,5 SPEED		
0FE9		Taste,binär		}			SPEED		
0FE8		Taste,gesamt		}			Count Bit		
0FE7		7-Segm-Code		}			DATA		
0FE6			}	} halbe			Checksumme		
0FE5			} Tasten	} Bytes			Count,Byte		
0FE4			} binär	}					
0FE3			}	}					
0FE2			Byte,higher	ADR,higher	ADR,higher	Diff.,higher	Stop.,higher		NBA,higher
0FE1			Byte,lower	ADR,lower	ADR,lower	Diff.,lower	Stop.,lower		NBA,lower
0FE0	STKBSE		Zähler	DATA	DATA		Flag „ADR"		

Abb. 1 Prinzipielle Arbeitsweise des Software-LIFO-Stack

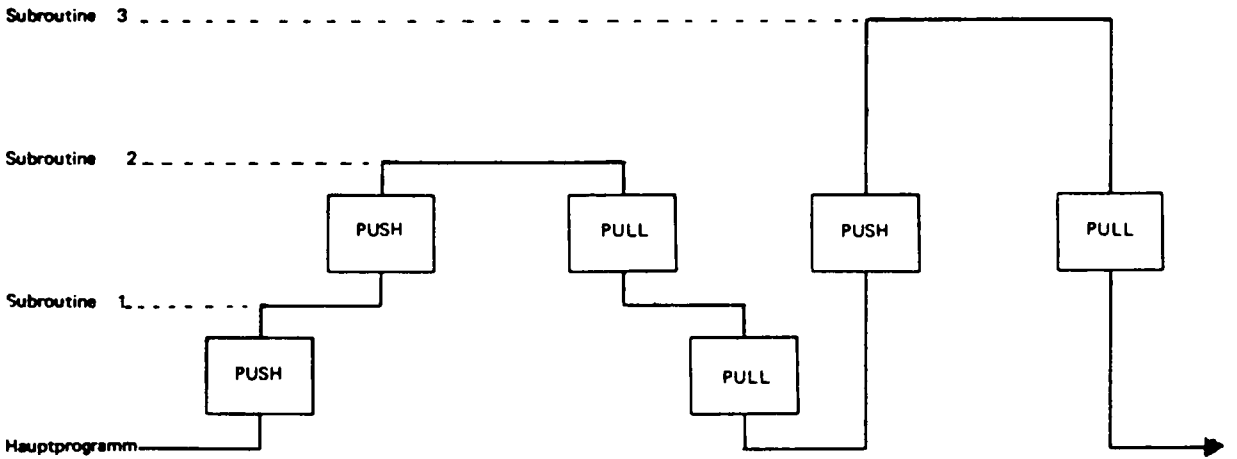
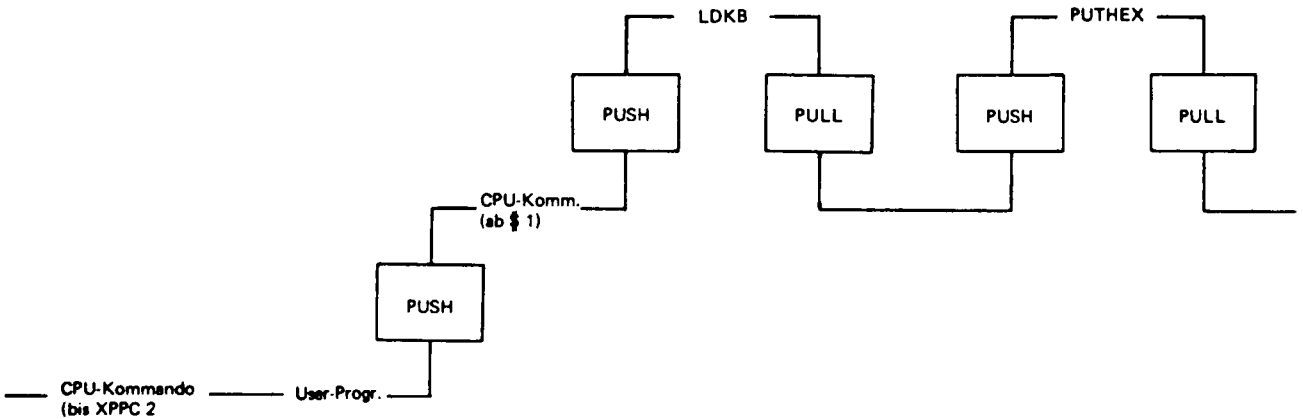


Abb. 2 Verwendung des STACK in dem CPU-Kommando



Um den betreffenden Pointer außerhalb der Stack-Routinen frei verwenden zu können, wird sein Inhalt vor Verlassen der Stack-Routinen in eine fixe RAM-Adr. geladen (STAKPT = OFFF und OFFE). In diese Adr. wird in der „ Initialize " die Adr. des Stack-Anfangs (Stack-Base = STKBSE) geschrieben.
Bei Interrupt-Verarbeitung kann u.U. eine beachtliche Stack-Tiefe zusammenkommen, die dann entweder in unbeschaltete Adressbereiche oder in RAM-Bereiche hineinläuft in welchen andere Daten oder Programme stehen. Um das zu vermeiden, wird ein Stack-Zähler (STKEFF) mitgeführt, der bei jeder Stack-Operation in- bzw. dekrementiert wird. (Updating). Außerdem ist ein über Moldify einzustellendes RAM-Byte vorgesehen (STDEEP) welches die maximal zulässige Stack-Tiefe enthält und bei jedem Status-Abspeichern mit der effektiv erreichten Tiefe (STKEFF) verglichen wird. Ist die maximale Tiefe erreicht, so wird ein „ Stack-Full "-Flag, ein RAM-Byte (= STFULL = OFFB) mit X'FF geladen.

Dieses Byte kann aus Anwender-Programmen abgefragt und falls gesetzt, ein weiterer Interrupt gesperrt werden.
Ein STATUS benötigt 11 RAM-Byte. Die Einstellung der maximal zulässigen Tiefe (STDEEP) wird für jeden kompletten STATUS um 1 erhöht.

Aus Abb. 1 ist ersichtlich daß 2 getrennte Stack-Routinen: PUSH und PULL notwendig sind. PUSH bewerkstelligt das Abspeichern der CPU-Registerinhalte in das RAM. Am Ende von PUSH befinden sich Befehle zum Ansprung der Subroutine. Deren Adresse (−1) muß vorher in die fixe RAM-Adr. „ ROUTAD " (= Adr. der Routine) geschrieben werden. Dazu bedient man sich entweder der Tatsache, daß zum Ansprung von PUSH PTR 3 mit PUSH−1 geladen werden muß und ROUTAD somit Pointer-(3)-relativ adressierbar ist (s. Listing Adr. 0255) oder eines bereits vorher geladenen RAM-Pointers (s. Listing Adr. OOBF).

Hier können einige Befehlsbyte gespart werden, wenn der vorherige Inhalt von ROUTAD zum Teil identisch mit dem neuen erforderlichen Inhalt ist. Im ELBUG liegen die Subroutinen LDKB, GETHEX und PUTHEX auf der gleichen 1/4 PAGE. Das higher Adr.-Byte ist für alle 3 identisch (= Ø2). Wenn jetzt in ROUTAD noch die Adr. LDKB−1 steht und es soll GETEX angesprungen werden, so braucht das higher Byte von ROUTAD nicht mehr geladen zu werden (s. Listing Adr. 013F).

Jede Subroutine selbst muß mit JS 3 (PULL) (= Makro, s. Assembler-Manual) beendet sein, damit das Hauptprogramm (oder die Subroutine von der vorherigen „ Etage ") erst angesprungen wird, wenn der betr. STATUS aus dem STACK wieder in die CPU-Reg. geladen ist. Dieses Zurück-Laden der CPU wird durch die Stack-Routine PULL besorgt.

Falls in einer Subroutine PTR 3 nicht verwendet wird, kann sie mit XPPC 3 (anstatt „ JS 3 (PULL) ") beendet werden. PTR 3 zeigt ja dann noch auf das Ende von PUSH und hier befindet sich ein Sprung nach PULL.

Ebenso kann eine gewisse Subroutine aus einer anderen Programm-Ebene zum 2. oder jedem weiteren mal mit XPPC 3 angesprungen werden, wenn (PTR 3) inzwischen nicht verändert worden ist (s. Listing Adr. Ø37A).

Dieser befehlssparende Ansprung ist möglich, weil PULL mit einem Sprung nach PUSH beendet ist und weil (ROUTAD) genau wie die CPU-Register auch in den Stack geschoben wird (= fast ein 4. Index-Register des SC/MP).

Die hier beschriebene Handhabung der Stack-Routinen hat natürlich volle Gültigkeit auch für Anwender-Programme. Falls in diesen eine Interrupt-Verarbeitung vorgesehen ist, muß dieser solange gesperrt bleiben, bis die o.a. organisatorischen Aufgaben erledigt sind (PTR 3 auf PUSH—1 stellen und Subroutinenadresse in ROUTAD laden). Nach Freigabe wird dann das durch Interrupt ausgelöste XPPC 3 wirksam. Bei mehreren Interrupt-Eingängen muß noch eine kleine Erkennungs-Software (s. ELEKTOR) verwirklicht werden.
Zu den anderen Routinen und Kommandos ist eigentlich nicht mehr viel zu sagen. Alles wesentliche geht aus dem Listing, den Flußdiagrammen und der RAM-Tabelle hervor.
Nach PUSH stehen dem Anwender *alle* 7 Register des SC/MP zur Verfügung; sie können, je nach Bedarf neu geladen werden.

Innerhalb des ELBUG werden die Routinen bis maximal zu einer 2-fachen Verschachtelung benutzt, sodaß das RAM von Adr. OFC9 abwärts frei zur Verfügung steht. Stack-Routinen, Kommando-Schleife und alle Kommandos befinden sich auf Hauptprogramm-Ebene, die Subroutinen LDKB, GETHEX und PUTHEX auf Ebene 1 und LDKB (von GETHEX aufgerufen) auf Ebene 2. LDBYTE und BYTOUT werden nicht über den Stack angesprungen.

LDKB kann auch auf Hauptprogramm-Ebene, unter Verwendung des Ansprung-Labels LDKB1 aufgerufen werden. Der Binär-Wert der Taste wird im E-Reg. und der 7-Segm.-Code im AC „ mitgebracht ". Ein wiederholtes Anspringen mit XPPC 3 (ohne Stack) ist leider nicht möglich, weil am Ende von LDKB (aus Speicher-Platzgründen) der Befehl JMP LDKB1 fehlt.

GETHEX holt durch 4-maliges Anspringen (über den Stack) von LDKB, 4 Hex-Tasten, schreibt diese auf Display 5 bis 2, stellt die beiden Byte zusammen und speichert sie in Adr. ØFE1 — 0FE2.

PUTHEX holt 3 Byte von Adr. ØFEØ bis ØFE2, wandelt sie in 7-Segm.-Code und schreibt diesen auf Display Ø — 5. Wenn aus Anwender-Programmen Hex-Zahlen auf das Display gebracht werden sollen, geht man so vor:

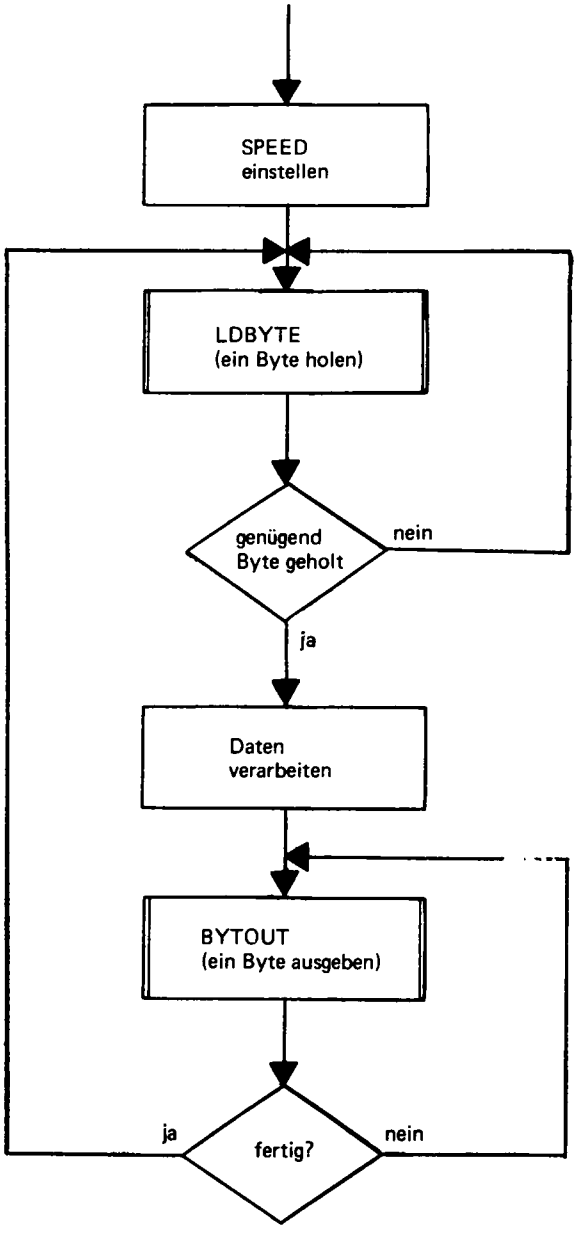
- 3 (oder weniger) Byte in Adr. ØFEØ — ØFE2 schreiben
- PTR 3 auf PUSH—1 stellen,
- PTUHEX—1 in ROUTAD schreiben (PTR-3-relativ oder über einen bereits geladenen RAM-Pointer) und
- XPPC 3.

Falls die gewünschte Zahl weniger als 6 Stellen hat, müssen anschließend die überflüssigen Display-Stellen gelöscht werden.
Die Verwendung von GETHEX geschieht genauso, nur daß die 2 Byte nach dem Rücksprung von Adr. OFE1 — OFE2 geholt werden.

In dem *CPU-KOMMANDO* ist ein kleiner Programmier-Trick enthalten. Das User-Programm wird mit XPPC 3 verlassen. PTR 3, der innerhalb dieses Programms nicht benutzt werden darf, wurde vorher auf PUSH—1 gestellt. Dadurch wird der „ STATUS " am betreffenden Punkt des User-Programms in den Stack geschoben. Anschließend springt das Programm nach „ $ 1 "-CPU-Kommando ($ 1 wurde vorher von der CPU-Kommando-Software in ROUTAD geladen).
Damit dieser „ Anwender-Status " nicht verlorengeht, wird die STACK-BASE jetzt um eine Etage „ verfälscht ". Dadurch werden die „ STATEN " der folgenden Routinen (LDKB und PTUHEX eine Etage tiefer in das RAM geschrieben und somit bleibt der „ Anwender-Status " erhalten. (Abb. 2)

Die Verwendung der Kassetten-Routinen LDBYTE und BYTOUT aus Anwenderprogrammen geschieht mit „ JS 3 (LDBYTE)" oder „ JS 3 (BYTOUT)". Ein wiederholter Ansprung mit einem einfachen XPPC 3 ist möglich.

Für ein gleichzeitiges Arbeiten mit 2 Rekordern (notwendig beim Assemblieren, Listen bearbeiten usw) kann ein eigenes Programm nach dem Flußdiagramm erstellt werden.



Verwendung von LDBYTE und BYTOUT aus User-Programmen für simultanes Arbeiten mit 2 Kassetten-Rekordern.

Programmbeschreibung

1.
Das neue verbesserte 4K-Diagnoseprogramm. Es belegt die Adresse 1000 - 1228. Es ist so geschrieben, daß es auf Seite 0 lauffähig ist, damit auch die Page 1 ausgetestet werden kann. Sie müssen vor Benutzung einen Blocktransfer auf die Seite 0, d.h. auf die Adresse 0C00 machen. Das benutzbare Programm liegt dann von 0C00 - 0E26. Start bei 0C00.

Das Programm meldet sich ähnlich wie bei ELBUG-Kommandos mit dG. Nun geben Sie die Anfangs-Adressen Ihrer Speicherkarte, z.B. 2000, und folgend die Endadressen, z.B. 2FFF, ein. Bei Verwendung des Programms ist zu beachten, daß die beiden letzten Hex-Zeichen immer FF sein müssen. Somit ist es möglich, jedes 1/4 K zu überprüfen. Das Programm ist nicht auf eine Page begrenzt und man kann beliebig große Speicher damit testen. Es benötigt, um eine Seite durchzutesten, ca. 1 Minute. Bei größeren Speicherbereichen verdoppelt sich die Zeit nicht, sondern steigt exponentiell.

Nach dem Start des Programms wird der gesamte Testbereich mit Nullen geladen. Dann wird in das erste und bei den nächsten Durchläufen jeweils das nächste 1/4K diese 55 geschrieben. Nachdem in 1/4K diese 55 geschrieben worden ist, werden alle übrigen Bereiche abgefragt, ob auch hier eine 55 auftritt. Trifft dies zu, steigt das Programm aus und schreibt "error 1" auf das Display. Nach dem error erscheint die Adresse des Bereiches, in dem die 55 geschrieben wurde und darauf die Adresse, in der eigentlich keine 55 stehen dürfte.

Hiermit kann eine fehlerhafte Adress-Decodierung, kurzgeschlossene CHIP Enable-Leitungen vom Adress-Decoder etc. leicht herausgefunden werden.

Wenn diese Schleife bis zum Ende des RAM-Bereiches durchgelaufen ist und keine vorgen. Fehler gefunden wurden, wird nochmals der gesamte Bereich mit Nullen geladen und abgefragt, ob tatsächlich überall Nullen vorhanden sind. Ist dies nicht der Fall, schreibt das

Programm "error 2" und anschließend die Adresse des Speichers, in dem keine Null gefunden wurde. Anschließend erscheint eine zweistellige Hex-Zahl. Diese Hex-Zahl gibt das Bit an, welches eine falsche Information enthält.

Wenn das Programm ausgestiegen ist und error an das Display gegeben hat, läuft es in einer Schleife und forscht die weiteren Speicher-Bereiche nicht mehr durch. Der Hardware-Fehler muß mittels Lötkolben oder neuen IC's zunächst ausgemerzt werden. Dann kann die Karte neu von vorn getestet werden.

Sind in den Bereichen 1 und 2 keine Fehler vorhanden, schreibt das Programm FF in den gesamten Speicherbereich und fragt ab, ob überall ein FF vorhanden ist. Falls nicht, erscheint "error 3" und die Adresse, von wo die falsche Antwort kam, sowie wieder die 2-stellige Hex-Zahl, die wieder das betreffende Bit angibt. Wenn die gesamte Speicherkarte durchlaufen ist und kein Fehler gefunden wurde, kommt "error no" auf's Display, gefolgt von der Anfangs- und Endadresse des gewünschten Speicher-Bereiches.

2.
Auf Adresse 1230 - 12E3 liegt ein memory compare = Speicherbereichs-Vergleichsprogramm. Es meldet sich nach dem Start mit co... Ähnlich wie beim Blocktransfer wird jetzt die Anfangs- und Endadresse des zu testenden Blocks und sofort folgend die Anfangsadresse des zu vergleichenden Blocks eingegeben. Bei Nichtübereinstimmen von zwei Blöcken erscheint die Adresse des Fehlers auf den mittleren vier Displays. Auf den letzten beiden Displays steht die fehlerhafte Date unter dieser Adresse; bei Fehlerlosigkeit erscheint auf dem Display "no error".

3.
Unter der Adresse 1300 - 136E finden Sie ein verbessertes hexadezimales Subtraktionsprogramm. Nach dem Start meldet es sich mit ..-..=.. Es werden nun zunächst der Minuhend und folgend der Subtrahend zweistellig eingegeben. Die Differenz erscheint sofort, so daß auch nach dem Ergebnis die gesamte Rechnung noch überprüft werden kann. Hat man sich bei der Eingabe vertippt, kann an beliebiger Stelle durch

Drücken der S-Taste neu eingegeben werden. Am Ende der Rechnung ist dies jedoch nicht erforderlich, es kann sofort mit der nächsten Subtraktion begonnen werden.

4.
Unter 1370 - 14C7 befindet sich das Hex-Matrix-Programm für den Kleindrucker. Nach dem Start erscheint auf dem Display PR.
Bei Drücken der Taste T erscheint PR ...ti. Jetzt stehen Ihnen 8 Byte zum Einschreiben eines Titels zur Verfügung. Der Titel wird im 5 Bit-ASCII-Code geschrieben, z.B. 01 für A. nach erfolgter Titeleingabe erscheint auf dem Display ...ad und Sie können jetzt die Anfangs und Endadresse des auszudruckenden Blockes einschreiben. Nach Eingabe der Endadresse beginnt der Drucker automatisch mit dem Druckvorgang. Wollen Sie auf einen Teil verzichten, drücken Sie nach dem Start sofort eine beliebige Taste außer T und es erscheint ... ad.

5.
Adresse 14D0 - 1669 enthalten den Si-Druck. Nach dem Start meldet es sich wie das Hex-Matrix Programm und wird auch genauso bedient. Der Ausdruck erfolgt hier jedoch nach 1 und 2 Byte Befehlen getrennt.

6.
Um mit der ASCII-Tastatur einen Text vom Kleindrucksystem ausdrucken zu lassen, bedient man sich des unter Adresse 1670 - 16C5 befindlichen Programms. Es können nach dem Start ASCII-Buchstaben, Zahlen und Sonderzeichen (nur Großbuchstaben mit Shift) eingegeben werden. Nach dem Einschreiben von 32 Zeichen beginnt der Drucker automatisch mit dem Ausdruck. Sollen in eine Zeile weniger als 32 Zeichen geschrieben werden, startet man den Drucker mit CR. Ebenfalls kann mit CR eine Leerzeile gedruckt werden. Mit BS (Backspace) läßt sich ein vorangegangener Tipfehler korrigieren.

7.
Unter 16D0 - 170A befindet sich ein Programm, mittels dessen man von der ASCII-Tastatur auf das 7-Segment Display der Hex I/O Karte schreiben kann. Ausgegeben wird der Hex-Coce des ASCII-Zeichen

8.
Das Programm steht unter 1710 - 178F. Nach dem Start wird der Bild-

schirm gelöscht und der Cursor (home) nach links oben gesetzt. Jetzt können Sie mit der ASCII-Tastatur auf den Bildschirm schreiben (Großbuchstaben mit Shift).
Folgende Tasten sind mit Sonderfunktionen belegt:

BS = Backspace Cursor nach links und Löschen des vorangegangenen Zeichens
CR = return und feed line= setzt den Cursor an den Anfang der nächsten Zeile
ENQ = cursor home = Cursor nach links oben an den Bildschirmanfang
EM = erase memory = Löschen des ganzen Bildschirmes und Cursor home
FS = Cursor eine Stelle nach rechts
GS = Cursor eine Stelle nach unten
RS = Cursor eine Stelle nach links
US = Cursor eine Stelle aufwärts

9.
Adresse 1790 - 1826. Mit diesem Programm können Sie einen Speicherbereich ähnlich wie bei dem Sidruck nach 1 und 2 Byte Befehlen getrennt in 4 Spalten auf den Bildschirm bringen. Zunächst schreiben Sie mit Modify unter den Adressen 17A1 das higher order Byte und unter 17A4 das lower order Byte der Startadresse des auszulesenden Speicherbereichs ein. Nach dem Starten zeigt sich die gewünschte Tabelle auf dem Bildschirm und das Halt LED geht an. Nach dem Drücken der Taste HR auf der Hex Tastatur erscheint die nächste Spalte auf dem Bildschirm.

10.
Bei 1830 - 1843 steht ein Programm zum Löschen des Bildschirmes, d.h. der Bildspeicher wird mit 20 geladen.

11.
Mit dem unter 1850 - 1860 zu findenden Programm können Sie alle möglichen ASCII-Zeichen auf dem Bildschirm darstellen.

12.
Adresse 1870 - 188E Programm wie vor, nur ständiger Wechsel des Bildes.

13.
Unter der Adresse 1890 -18CF ist ein Programm, welches den Dialog zwischen einem Terminal und dem Prozessor ermöglicht. Im On line Betrieb schreibt das Terminal den eingeschriebenen Text auf den Bildschirm. Gleichzeitig wird dieser in den RAM-Bereich von 0C00 an aufwärts bis maximal 0FC9 gespeichert. Er kann von hier aus mit dem Programm beliebig oft wieder ausgelesen werden.

14.
Adresse 18D0 - 1910. Dieses Programm dient zum Auslesen eines RAM-speicherbereiches und Sichtbarmachen auf dem Bildschirm eines Terminals. Der RAM-Bereich 0C00-0FC9 kann per Modify geladen werden oder per Blocktransfer oder durch Anwenderprogramm wie das folgende unter 15. Die Länge des abzuspeichernden Programms bestimmen Sie, indem Sie die Adressen 190F (high order) und 1910 (low order) laden. Soll z.B. das Programm 256 Byte lang sein, d.h. die Anfangsadresse soll bei 0C00 sein und die Endadresse bei 0D00. In diesem Falle ist die high order 0D und die low order 00.

15.
1920 - 194E. Dieses Programm schreibt in den RAM-Bereich von 0C00-0FB0 alle möglichen ASCII-Zeichen ein.

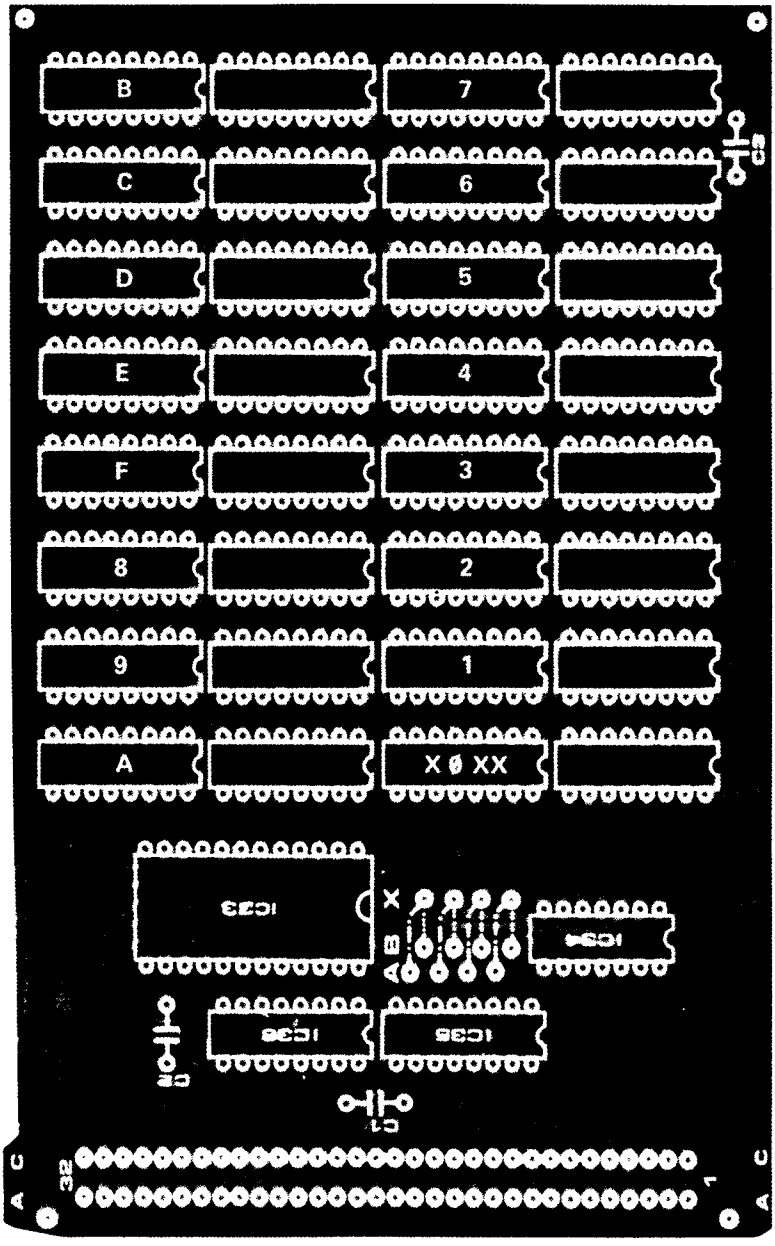
16.
1950 - 1A4C. Es hat sich als vorteilhaft erwiesen, Cassetten mit aufzunehmenden Programmen mehrfach zu bespielen, und zwar mit unterschiedlichen Geschwindigkeiten. Um nicht ständig den Prozessor mit Anfang-, Endadressen und Geschwindigkeit füttern zu müssen, geschieht dies im vorliegenden Programm automatisch. Man gibt Start und Endadresse an, startet den Rekorder und drückt die Taste down. Jetzt gibt der Prozessor 2 Aufnahmen mit 600 Baud, und je eine mit 110 und 55 Baud aus. Zwischen den Ausgaben liegt je eine Pause von 10 sec.

Nach Beendigung der Ausspielung leuchtet das Haltled auf und an Flag 1 liegt ein Signal an. Ein angeschlossener Summer (siehe Schaltvorschlag) gibt Signal. Nach dem Start meldet sich das Programm mit CM. Will man nach Beendigung des Programms eine zweite Aufnahme machen, genügt es, den Halttaster zu drücken und mit "down" neu zu starten.

17.
Bei 1A50 - 1B74 liegt das BIN List Programm. Es dient dazu. Speicherbereiche in binärer Form auszudrucken. Dies ist beim Programmieren mit einem Prommer z.B. manchmal erforderlich. Das Programm meldet sich mit bi..... und wird nach Anfangs- und Endadresseneingabe gestartet.
Alle Programme, bei denen die USART, TIMER Karte verwendet wird, sind so ausgelegt, daß der 8 bit DIL-Schalter auf der Adresse 8000 steht.

SPEICHERADRESSEN auf der Elektor 4 K RAM Karte

BIT 7-4	BIT Ø-3	BIT 7-4	BIT Ø-3



15) ALPHABET IN RAM
1920 - 194F

```
1920 C4 0C 37 C4 00 33 C4 20
1928 C8 26 90 04 A8 22 CF 01
1930 37 01 40 37 C4 0F 60 9C
1938 0A 33 01 40 33 C4 B0 60
1940 9C 01 00 C0 8B E4 5A 9C
1948 E3 C4 0A CF 01 90 D7 55

      $ 0
1920  C40C  LDI 0C
1922  37    XPAH 3
1923  C400  LDI 00
1925  33    XPAL 3
      $ 1
1926  C420  LDI 20
1928  C826  ST 26 =194F
192A  9004  JMP $ 3
      $ 2
192C  A822  ILD 22 =194F
192E  CF01  ST @01 (3)
      $ 3
1930  37    XPAH 3
1931  01    XAE
1932  40    LDE
1933  37    XPAH 3
1934  C40F  LDI 0F
1936  60    XRE
1937  9C0A  JNZ $ 4
1939  33    XPAL 3
193A  01    XAE
193B  40    LDE
193C  33    XPAL 3
193D  C4B0  LDI B0
193F  60    XRE
1940  9C01  JNZ $ 4
1942  00    HALT
      $ 4
1943  C00B  LD 0B =194F
1945  E45A  XRI 5A
1947  9CE3  JNZ $ 2
1949  C40A  LDI 0A
194B  CF01  ST @01 (3)
194D  90D7  JMP $ 1
194F  55    BYTE
```

16) KASSETTENBESPIELPROGRAMM
1950 - 1A4C

```
1950 C4 39 C9 06 01 19 C4 37
1958 C9 05 C4 80 C9 04 C9 03
1960 C9 02 C9 01 C4 00 C9 00
1968 C9 FF C4 00 37 C4 4F 33
1970 3F C3 99 E4 90 98 1B C4
1978 5F C9 00 C4 5E C9 FF C4
1980 3E CB AE 3F C3 92 CB 9C
1988 C3 93 CB 9D 3F C4 0A CB
1990 AE 3F C4 04 C8 4C C4 19
1998 36 C4 E3 32 C4 5E C9 00
19A0 C4 5C C9 FF C6 01 CB A6
19A8 C4 19 CB AD C4 E7 CB AE
19B0 3F B8 2F 98 0C C4 10 C8
19B8 2A 8F FF B8 26 9C FA 90
19C0 E3 8F FF C4 AB 01 19 8F
19C8 FF A8 17 E4 08 9C F7 C4
19D0 01 07 00 C4 02 CB AD C4
19D8 0A CB AE C4 00 07 32 92
19E0 4F 55 55 15 15 85 FE 00
19E8 C4 0F 36 C4 E0 32 C2 08
19F0 31 C2 0C 35 C4 D7 33 C4
19F8 05 37 C2 0C 3F C2 0B 3F
1A00 C2 02 3F C2 01 3F C4 20
1A08 CA 05 C4 00 CA 06 02 C1
1A10 00 01 C2 06 70 CA 06 40
1A18 3F 35 01 40 35 40 E2 02
1A20 9C 09 31 01 40 31 40 E2
1A28 01 98 16 06 01 02 31 F4
1A30 01 31 35 F4 00 35 40 07
1A38 BA 05 9C D3 C2 06 3F 90
1A40 C5 C2 06 3F C4 00 37 C4
1A48 53 33 3F 90 9B

      $ 0
1950  C439  LDI 39
1952  C906  ST 06 (1)
1954  01    XAE
1955  19    SIO
1956  C43F  LDI 3F
1958  C905  ST 05 (1)
195A  C480  LDI 80
195C  C904  ST 04 (1)
195E  C903  ST 03 (1)
1960  C902  ST 02 (1)
1962  C901  ST 01 (1)
1964  C400  LDI 00
1966  C900  ST 00 (1)
1968  C9FF  ST FF (1)
196A  C400  LDI 00
196C  37    XPAH 3
196D  C44F  LDI 4F
196F  33    XPAL 3
1970  3F    XPPC 3
1971  C399  LD 99 (3)
1973  E490  XRI 90
1975  981B  JZ $ 1
1977  C45F  LDI 5F
1979  C900  ST 00 (1)
197B  C45E  LDI 5E
197D  C9FF  ST FF (1)
197F  C43E  LDI 3E
1981  CBAE  ST AE (3)
1983  3F    XPPC 3
1984  C392  LD 92 (3)
1986  CB9C  ST 9C (3)
1988  C393  LD 93 (3)
198A  CB9D  ST 9D (3)
198C  3F    XPPC 3
198D  C40A  LDI 0A
198F  CBAE  ST AE (3)
1991  3F    XPPC 3
      $ 1
1992  C404  LDI 04
1994  CB4C  ST 4C =19E1
1996  C419  LDI 19
1998  36    XPAH 2
1999  C4E3  LDI E3
199B  32    XPAL 2
199C  C45E  LDI 5E
199E  C900  ST 00 (1)
19A0  C45C  LDI 5C
19A2  C9FF  ST FF (1)
      $ 2
19A4  C601  LD @01 (2)
19A6  CBA6  ST A6 (3)
19A8  C419  LDI 19
19AA  CBAD  ST AD (3)
19AC  C4E7  LDI E7
19AE  CBAE  ST AE (3)
19B0  3F    XPPC 3
19B1  B82F  DLD 2F =19E1
19B3  980C  JZ $ 4
19B5  C410  LDI 10
19B7  C82A  ST 2A =19E2
      $ 3
19B9  8FFF  DLY FF
19BB  B826  DLD 26 =19E2
19BD  9CFA  JNZ $ 3
19BF  98E3  JMP $ 2
      $ 4
19C1  8FFF  DLY FF
19C3  C4AB  LDI AB
19C5  01    XAE
      $ 5
19C6  19    SIO
19C7  8FFF  DLY FF
19C9  A817  ILD 17 =19E1
19CB  E408  XRI 08
19CD  9CF7  JNZ $ 5
19CF  C401  LDI 01
19D1  07    CAS
19D2  00    HALT
19D3  C402  LDI 02
19D5  CBAD  ST AD (3)
19D7  C40A  LDI 0A
19D9  CBAE  ST AE (3)
19DB  C400  LDI 00
19DD  07    CAS
19DE  32    XPAL 2
19DF  924F  JMP 4F (2)
19E1  55    BYTE
19E2  55    BYTE
      TAB:
19E3 15 15 85 FE 00
      $ 6
19E8  C40F  LDI 0F
19EA  36    XPAH 2
19EB  C4E0  LDI E0
19ED  32    XPAL 2
19EE  C00B  LD 00 =
19F0  31    XPAL 1
19F1  C000  LD 00 (2)
19F3  35    XPAH 1
19F4  C407  LDI 07
```

```
19F6  33    XPAL 3
19F7  C405  LDI 05
19F9  37    XPAH 3
19FA  C20C  LD 0C (2)
19FC  3F    XPPC 3
19FD  C208  LD 08 (2)
19FF  3F    XPPC 3
1A00  C202  LD 02 (2)
1A02  3F    XPPC 3
1A03  C201  LD 01 (2)
1A05  3F    XPPC 3
      $ 7
1A06  C420  LDI 20
1A08  CA05  ST 05 (2)
1A0A  C400  LDI 00
1A0C  CA06  ST 06 (2)
1A0E  02    CCL
      $ 8
1A0F  C100  LD 00 (1)
1A11  01    XAE
1A12  C206  LD 06 (2)
1A14  70    ADE
1A15  CA06  ST 06 (2)
1A17  40    LDE
1A18  3F    XPPC 3
1A19  35    XPAH 1
1A1A  01    XAE
1A1B  40    LDE
1A1C  35    XPAH 1
1A1D  40    LDE
1A1E  E202  XOR 02 (2)
1A20  9C09  JNZ $ 9
1A22  31    XPAL 1
1A23  01    XAE
1A24  40    LDE
1A25  31    XPAL 1
1A26  40    LDE
1A27  E201  XOR 01 (2)
1A29  9816  JZ $ 10
      $ 9
1A2B  06    CSA
1A2C  01    XAE
1A2D  02    CCL
1A2E  31    XPAL 1
1A2F  F401  ADI 01
1A31  31    XPAL 1
1A32  35    XPAH 1
1A33  F400  ADI 00
1A35  35    XPAH 1
1A36  40    LDE
1A37  07    CAS
1A38  BA05  DLD 05 (2)
1A3A  9CD3  JNZ $ 8
1A3C  C206  LD 06 (2)
1A3E  3F    XPPC 3
1A3F  98C5  JMP $ 7
      $ 10
1A41  C206  LD 06 (2)
1A43  3F    XPPC 3
1A44  C400  LDI 00
1A46  37    XPAH 3
1A47  C453  LDI 53

1A49  33    XPAL 3
1A4A  3F    XPPC 3
1A4B  909B  JMP $ 6


17) BIN-LIST
1A50 - 1B74

1A50 C4 7C C9 06 C4 05 C9 05
1A58 C4 00 C9 04 C9 03 C9 02
1A60 C9 01 C9 00 C9 FF C4 00
1A68 37 C4 55 33 C4 02 CB A7
1A70 C4 3E CB A8 3F C4 1B 35
1A78 C4 70 31 C2 02 C9 00 C2
1A80 01 C9 01 3F C2 02 C9 02
1A88 C2 01 C9 03 C1 00 C9 FF
1A90 C4 06 37 C4 55 33 C4 1A
1A98 CB A7 C4 E4 CB A8 3F C4
1AA0 0F 36 C4 00 32 C1 FE CA
1AA8 00 C1 FF CA 01 C1 01 C9
1AB0 FF 3F C1 FE CA 02 C1 FF
1AB8 CA 03 C4 60 CA 04 C1 00
1AC0 36 C1 01 32 C6 01 C9 FF
1AC8 36 C9 00 32 C9 01 3F C4
1AD0 0F 36 C4 03 32 C1 FE CA
1AD8 FD C1 FF CA FE C4 60 CA
1AE0 [illegible]
1AE8 [illegible]
1AF0 32 C2 00 14 F0 1E 1E 1E
1AF8 1E 01 C1 00 CA FF C2 20
1B00 C4 0F 01 C1 00 C4 00 3F
1B08 50 D6 76 71 72 73 74 75
1B10 76 77 78 79 41 42 43 44
1B18 45 46 C1 00 37 C1 01 33
1B20 C3 FF C9 FF C4 08 C9 04
1B28 C4 1B 37 C4 65 33 C1 FF
1B30 01 C1 04 01 03 00 98 04
1B38 C4 71 30 02 C4 70 CE 01
1B40 B9 04 9C EA C4 FF CA 20
1B48 C4 00 37 C4 55 33 C4 F0
1B50 CB A7 C4 0E CB A8 3F C1
1B58 00 E1 02 9C 86 C1 01 E1
1B60 03 9C 80 06 90 FD 01 02
1B68 04 08 10 20 40 80 55 55
1B70 55 55 55 55 55

      $ 0
1A50  C47C  LDI 7C
1A52  C906  ST 06 (1)
1A54  C405  LDI 05
1A56  C905  ST 05 (1)
1A58  C400  LDI 00
1A5A  C904  ST 04 (1)
1A5C  C903  ST 03 (1)
1A5E  C902  ST 02 (1)
1A60  C901  ST 01 (1)
1A62  C900  ST 00 (1)
1A64  C9FF  ST FF (1)
1A66  C400  LDI 00
1A68  37    XPAH 3
1A69  C455  LDI 55

1A6B  33    XPAL 3
1A6C  C402  LDI 02
1A6E  CBA7  ST A7 (3)
1A70  C43E  LDI 3E
1A72  CBA8  ST A8 (3)
1A74  3F    XPPC 3
1A75  C41B  LDI 1B
1A77  35    XPAH 1
1A78  C470  LDI 70
1A7A  31    XPAL 1
1A7B  C202  LD 02 (2)
1A7D  C900  ST 00 (1)
1A7F  C201  LD 01 (2)
1A81  C901  ST 01 (1)
1A83  3F    XPPC 3
1A84  C202  LD 02 (2)
1A86  C902  ST 02 (1)
1A88  C201  LD 01 (2)
1A8A  C903  ST 03 (1)
      $ 1
1A8C  C100  LD 00 (1)
1A8E  C9FF  ST FF (1)
1A90  C400  LDI 00
1A92  37    XPAH 3
1A93  C455  LDI 55
1A95  33    XPAL 3
1A96  C41A  LDI 1A
1A98  CBA7  ST A7 (3)
1A9A  C4E4  LDI E4
1A9C  CBA8  ST A8 (3)
1A9E  3F    XPPC 3
1A9F  C40F  LDI 0F
1AA1  36    XPAH 2
1AA2  C400  LDI 00
1AA4  32    XPAL 2
1AA5  C1FE  LD FE (1)
1AA7  CA00  ST 00 (2)
1AA9  C1FF  LD FF (1)
1AAB  CA01  ST 01 (2)
1AAD  C101  LD 01 (1)
1AAF  C9FF  ST FF (1)
1AB1  3F    XPPC 3
1AB2  C1FE  LD FE (1)
1AB4  CA02  ST 02 (2)
1AB6  C1FF  LD FF (1)
1AB8  CA03  ST 03 (2)
1ABA  C460  LDI 60
1ABC  CA04  ST 04 (2)
1ABE  C100  LD 00 (1)
1AC0  36    XPAH 2
1AC1  C101  LD 01 (1)
1AC3  32    XPAL 2
1AC4  C601  LD @01 (2)
1AC6  C9FF  ST FF (1)
1AC8  36    XPAH 2
1AC9  C900  ST 00 (1)
1ACB  32    XPAL 2
1ACC  C901  ST 01 (1)
1ACE  3F    XPPC 3
1ACF  [illegible]
1AD1  [illegible]
1AD2  [illegible]
```

```
1AD4 32   XPAL 2
1AD5 C1FE LD FE (1)
1AD7 CAFD ST FD (2)
1AD9 C1FF LD FF (1)
1ADB CAFE ST FE (2)
1ADD C460 LDI 60
1ADF CAFF ST FF (2)
1AE1 9037 JMP $ 4
     $ 2
1AE3 90A7 JMP $ 1
     $ 3
1AE5 C41B LDI 1B
1AE7 35   XPAH 1
1AE8 C40A LDI 0A
1AEA 31   XPAL 1
1AEB C41B LDI 1B
1AED 36   XPAH 2
1AEE C46F LDI 6F
1AF0 32   XPAL 2
1AF1 C200 LD 00 (2)
1AF3 D4F0 ANI F0
1AF5 1E   RR
1AF6 1E   RR
1AF7 1E   RR
1AF8 1E   RR
1AF9 01   XAE
1AFA C180 LD 80 (1)
1AFC CAFF ST FF (2)
1AFE C200 LD 00 (2)
1B00 D40F ANI 0F
1B02 01   XAE
1B03 C180 LD 80 (1)
1B05 CA00 ST 00 (2)
1B07 3F   XPPC 3
1B08 900B JMP $ 3
     TAB:
1B0A 70 71 72 73 74 75 76 77
1B12 78 79 41 42 43 44 45 46
     $ 4
1B1A C100 LD 00 (1)
1B1C 37   XPAH 3
1B1D C101 LD 01 (1)
1B1F 33   XPAL 3
1B20 C3FF LD FF (3)
1B22 C9FF ST FF (1)
1B24 C408 LDI 08
1B26 C904 ST 04 (1)
1B28 C41B LDI 1B
1B2A 37   XPAH 3
1B2B C465 LDI 65
1B2D 33   XPAL 3
     $ 5
1B2E C1FF LD FF (1)
1B30 01   XAE
1B31 C104 LD 04 (1)
1B33 01   XAE
1B34 D380 AND 80 (3)
1B36 9804 JZ $ 6
1B38 C471 LDI 71
1B3A 9002 JMP $ 7
     $ 6
1B3C C470 LDI 70
     $ 7
1B3E CE01 ST @01 (2)
1B40 B904 DLD 04 (1)
1B42 9CEA JNZ $ 5
1B44 C4FF LDI FF
1B46 CA00 ST 00 (2)
1B48 C400 LDI 00
1B4A 37   XPAH 3
1B4B C455 LDI 55
1B4D 33   XPAL 3
1B4E C4F0 LDI F0
1B50 CBA7 ST A7 (3)
1B52 C40E LDI 0E
1B54 CBA8 ST A8 (3)
1B56 3F   XPPC 3
1B57 C100 LD 00 (1)
1B59 E102 XOR 02 (1)
1B5B 9C06 JNZ $ 2
1B5D C101 LD 01 (1)
1B5F E103 XOR 03 (1)
1B61 9C00 JNZ $ 2
     $ 8
1B63 00   HALT
1B64 90FD JMP $ 8
     TAB:
1B66 01 02 04 08 10 20 40 80
1B6E 55   BYTE
1B6F 55   BYTE
1B70 55   BYTE
1B71 55   BYTE
1B72 55   BYTE
1B73 55   BYTE
1B74 55   BYTE
```

[illegible]
14D0 - 1669

```
14D0 08 C4 73 C9 06 C4 50 C9
14D8 05 C4 80 C9 04 C9 03 C9
14E0 02 C9 01 C9 00 C9 FF C4
14E8 00 CA 0F 37 C4 55 33 3F
14F0 C2 08 E4 B0 9C 43 C4 78
14F8 C9 00 C4 10 C9 FF CA 0F
1500 C4 3E CA 1D C4 04 CA 0E
1508 C4 D3 CA 12 3F C2 02 D4
1510 3F DC 80 01 AA 12 01 CA
1518 80 C2 01 D4 3F DC 90 01
1520 AA 12 01 CA 80 BA 0E 9C
1528 E3 AA 12 01 C4 FF CA 80
1530 90 02 CA 0F C4 0A CA 1D
1538 3F C4 5F C9 00 C4 5E C9
1540 FF C4 3E CA 1D 3F C2 02
1548 CA 11 C2 01 CA 10 3F C4
1550 00 CA CA C4 00 CA CB C4
1558 20 CA CC C4 0A CA 1D 3F
1560 C2 08 E4 E0 98 0C C4 0F
1568 CA CA C4 40 CA CB C4 60
1570 CA CC C4 FF CA 1D C4 FF
1578 CA 1C C4 B4 CA 13 C4 0F
1580 CA 14 C2 0F 98 01 3F C2
1588 10 31 C2 11 35 C4 D3 CA
1590 12 35 CA 0D 35 C4 02 CA
1598 0E C4 16 37 C4 30 33 3F
15A0 AA 12 01 C2 0C DA CB CA
15A8 80 AA 12 01 C2 0B DA CB
15B0 CA 80 BA 0E 98 06 31 CA
15B8 0D 31 90 DD AA 12 01 C4
15C0 08 CA 0E C2 CC CA 80 C2
15C8 CA 98 16 C1 00 94 06 C4
15D0 02 CA 0E 90 04 C4 01 CA
15D8 0E C2 CC CA 80 90 02 90
15E0 AC C1 00 CA 0D 3F C2 0C
15E8 DA CB 01 AA 12 01 CA 80
15F0 C2 0B DA CB 01 AA 12 01
15F8 CA 80 AA 12 01 31 E2 01
1600 9C 0D E2 01 31 35 E2 02
1608 98 1B E2 02 35 90 03 E2
1610 01 31 C5 01 BA 0E 9C C1
1618 C4 FF CA 80 C4 00 37 C4
1620 55 33 3F 90 BA C4 FF CA
1628 80 C4 00 37 C4 55 33 3F
1630 00 C4 16 37 CA 11 C4 5A
1638 33 CA 10 C2 0D D4 F0 1E
1640 1E 1E 1E 01 C3 80 CA 0C
1648 C2 0D D4 0F 01 C3 80 CA
1650 0B C2 11 37 C2 10 33 3F
1658 90 D7 30 31 32 33 34 35
1660 36 37 38 39 01 02 03 04
1668 05 06
```

```
     $ 0
14D0 08   NOP
14D1 C473 LDI 73
14D3 C906 ST 06 (1)
14D5 C450 LDI 50
14D7 C905 ST 05 (1)
14D9 C480 LDI 80
14DB C904 ST 04 (1)
14DD C903 ST 03 (1)
14DF C902 ST 02 (1)
14E1 C901 ST 01 (1)
14E3 C900 ST 00 (1)
14E5 C9FF ST FF (1)
14E7 C400 LDI 00
14E9 CA0F ST 0F (2)
14EB 37   XPAH 3
14EC C455 LDI 55
14EE 33   XPAL 3
14EF 3F   XPPC 3
14F0 C208 LD 08 (2)
14F2 E4B0 XRI B0
14F4 9C43 JNZ $ 3
14F6 C478 LDI 78
14F8 C900 ST 00 (1)
14FA C410 LDI 10
14FC C9FF ST FF (1)
14FE CA0F ST 0F (2)
1500 C43E LDI 3E
1502 CA1D ST 1D (2)
1504 C404 LDI 04
1506 CA0E ST 0E (2)
1508 C4D3 LDI D3
150A CA12 ST 12 (2)
```

```
     $ 1
150C 3F   XPPC 3
150D C202 LD 02 (2)
150F D43F ANI 3F
1511 DC80 ORI 80
1513 01   XAE
1514 AA12 ILD 12 (2)
1516 01   XAE
1517 CA00 ST 00 (2)
1519 C201 LD 01 (2)
151B D43F ANI 3F
151D DC80 ORI 80
151F 01   XAE
1520 AA12 ILD 12 (2)
1522 01   XAE
1523 CA00 ST 00 (2)
1525 BA0E DLD 0E (2)
1527 9CE3 JNZ $ 1
1529 AA12 ILD 12 (2)
152B 01   XAE
152C C4FF LDI FF
152E CA00 ST 00 (2)
1530 9002 JMP $ 2
1532 CA0F ST 0F (2)
     $ 2
1534 C40A LDI 0A
1536 CA1D ST 1D (2)
1538 3F   XPPC 3
     $ 3
1539 C45F LDI 5F
153B C900 ST 00 (1)
153D C45E LDI 5E
153F C9FF ST FF (1)
1541 C43E LDI 3E
1543 CA1D ST 1D (2)
1545 3F   XPPC 3
1546 C202 LD 02 (2)
1548 CA11 ST 11 (2)
154A C201 LD 01 (2)
154C CA10 ST 10 (2)
154E 3F   XPPC 3
154F C400 LDI 00
1551 CACA ST CA (2)
1553 C400 LDI 00
1555 CACB ST CB (2)
1557 C420 LDI 20
1559 CACC ST CC (2)
155B C40A LDI 0A
155D CA1D ST 1D (2)
155F 3F   XPPC 3
1560 C208 LD 08 (2)
1562 E4E0 XRI E0
1564 980C JZ $ 4
1566 C40F LDI 0F
1568 CACA ST CA (2)
156A C440 LDI 40
156C CACB ST CB (2)
156E C460 LDI 60
1570 CACC ST CC (2)
     $ 4
1572 C4FF LDI FF
1574 CA1D ST 1D (2)
1576 C4FF LDI FF
1578 CA1C ST 1C (2)
157A C4B4 LDI B4
157C CA13 ST 13 (2)
157E C40F LDI 0F
1580 CA14 ST 14 (2)
1582 C20F LD 0F (2)
1584 9801 JZ $ 5
1586 3F   XPPC 3
     $ 5
1587 C210 LD 10 (2)
1589 31   XPAL 1
158A C211 LD 11 (2)
158C 35   XPAH 1
     $ 6
158D C4D3 LDI D3
158F CA12 ST 12 (2)
1591 35   XPAH 1
1592 CA0D ST 0D (2)
1594 35   XPAH 1
1595 C402 LDI 02
1597 CA0E ST 0E (2)
     $ 7
1599 C416 LDI 16
159B 37   XPAH 3
159C C430 LDI 30
159E 33   XPAL 3
159F 3F   XPPC 3
15A0 AA12 ILD 12 (2)
15A2 01   XAE
15A3 C20C LD 0C (2)
15A5 DACB OR CB (2)
15A7 CA00 ST 00 (2)
15A9 AA12 ILD 12 (2)
15AB 01   XAE
15AC C208 LD 08 (2)
15AE DACB OR CB (2)
15B0 CA00 ST 00 (2)
15B2 BA0E DLD 0E (2)
15B4 9806 JZ $ 8
15B6 31   XPAL 1
15B7 CA0D ST 0D (2)
15B9 31   XPAL 1
15BA 90DD JMP $ 7
     $ 8
15BC AA12 ILD 12 (2)
15BE 01   XAE
15BF C408 LDI 08
15C1 CA0E ST 0E (2)
15C3 C2CC LD CC (2)
15C5 CA00 ST 00 (2)
15C7 C2CA LD CA (2)
15C9 9816 JZ $ 12
15CB C100 LD 00 (1)
15CD 9406 JP $ 9
15CF C402 LDI 02
15D1 CA0E ST 0E (2)
15D3 9004 JMP $ 10
     $ 9
15D5 C401 LDI 01
15D7 CA0E ST 0E (2)
     $ 10
15D9 C2CC LD CC (2)
15DB CA00 ST 00 (2)
15DD 9002 JMP $ 12
     $ 11
15DF 90AC JMP $ 6
     $ 12
15E1 C100 LD 00 (1)
15E3 CA0D ST 0D (2)
15E5 3F   XPPC 3
15E6 C20C LD 0C (2)
15E8 DACB OR CB (2)
15EA 01   XAE
15EB AA12 ILD 12 (2)
15ED 01   XAE
15EE CA00 ST 00 (2)
15F0 C208 LD 08 (2)
15F2 DACB OR CB (2)
15F4 01   XAE
15F5 AA12 ILD 12 (2)
15F7 01   XAE
15F8 CA00 ST 00 (2)
15FA AA12 ILD 12 (2)
15FC 01   XAE
15FD 31   XPAL 1
15FE E201 XOR 01 (2)
1600 9C0D JNZ $ 13
1602 E201 XOR 01 (2)
1604 31   XPAL 1
1605 35   XPAH 1
1606 E202 XOR 02 (2)
1608 981B JZ $ 15
160A E202 XOR 02 (2)
160C 35   XPAH 1
160D 9003 JMP $ 14
     $ 13
160F E201 XOR 01 (2)
1611 31   XPAL 1
     $ 14
1612 C501 LD @01 (1)
1614 BA0E DLD 0E (2)
1616 9CC1 JNZ $ 10
1618 C4FF LDI FF
161A CA00 ST 00 (2)
161C C400 LDI 00
161E 37   XPAH 3
161F C455 LDI 55
1621 33   XPAL 3
1622 3F   XPPC 3
1623 90BA JMP $ 11
     $ 15
1625 C4FF LDI FF
1627 CA00 ST 00 (2)
1629 C400 LDI 00
162B 37   XPAH 3
162C C455 LDI 55
162E 33   XPAL 3
162F 3F   XPPC 3
1630 00   HALT
```

```
     $ 16
1631  C416  LDI 16
1633  37    XPAH 3
1634  CA11  ST 11 (2)
1636  C45A  LDI 5A
1638  33    XPAL 3
1639  CA10  ST 10 (2)
163B  C20D  LD 0D (2)
163D  D4F0  ANI F0
163F  1E    RR
1640  1E    RR
1641  1E    RR
1642  1E    RR
1643  01    XAE
1644  C380  LD 80 (3)
1646  CA0C  ST 0C (2)
1648  C20D  LD 0D (2)
164A  D40F  ANI 0F
164C  01    XAE
164D  C380  LD 80 (3)
164F  CA0B  ST 0B (2)
1651  C211  LD 11 (2)
1653  37    XPAH 3
1654  C210  LD 10 (2)
1656  33    XPAL 3
1657  3F    XPPC 3
1658  90D7  JMP $ 16
     TAB:
165A 30 31 32 33 34 35 36 37
1662 38 39 01 02 03 04 05 06
```

6) ASCII-PRINTER
1670 - 16C5

```
1670 C4 F0 CA 1C C4 0E CA 1D
1678 C4 80 36 C4 00 32 C4 00
1680 37 C4 55 33 C4 0F 35 C4
1688 00 31 C2 00 94 FC D4 7F
1690 01 C2 00 94 02 90 FA 40
1698 E4 08 98 22 E4 05 98 17
16A0 40 D4 60 98 E5 E4 60 98
16A8 E1 40 D4 3F CD 01 31 01
16B0 40 31 C4 20 60 9C D3 C4
16B8 FF CD 01 3F 90 C9 31 98
16C0 C6 31 CD FF 90 C4
```

```
     $ 0
1670  C4F0  LDI F0
1672  CA1C  ST 1C (2)
1674  C40E  LDI 0E
1676  CA1D  ST 1D (2)
1678  C480  LDI 80
167A  36    XPAH 2
167B  C400  LDI 00
167D  32    XPAL 2
167E  C400  LDI 00
1680  37    XPAH 3
1681  C455  LDI 55
1683  33    XPAL 3
1684  C40F  LDI 0F
1686  35    XPAH 1
     $ 1
1687  C400  LDI 00
1689  31    XPAL 1
     $ 2
168A  C200  LD 00 (2)
168C  94FC  JP $ 2
168E  D47F  ANI 7F
1690  01    XAE
     $ 3
1691  C200  LD 00 (2)
1693  9402  JP $ 4
1695  90FA  JMP $ 3
     $ 4
1697  40    LDE
1698  E408  XRI 08
169A  9822  JZ $ 6
169C  E405  XRI 05
169E  9817  JZ $ 5
16A0  40    LDE
16A1  D460  ANI 60
16A3  98E5  JZ $ 2
16A5  E460  XRI 60
16A7  98E1  JZ $ 2
16A9  40    LDE
16AA  D43F  ANI 3F
16AC  CD01  ST @01 (1)
16AE  31    XPAL 1
16AF  01    XAE
16B0  40    LDE
16B1  31    XPAL 1
16B2  C420  LDI 20
16B4  60    XRE
16B5  9CD3  JNZ $ 2
     $ 5
16B7  C4FF  LDI FF
16B9  CD01  ST @01 (1)
16BB  3F    XPPC 3
16BC  90C9  JMP $ 1
     $ 6
16BE  31    XPAL 1
16BF  98C6  JZ $ 1
16C1  31    XPAL 1
16C2  CDFF  ST @FF (1)
16C4  90C4  JMP $ 2
```

7) ASCII-DISPLAY
16D0 - 170A

```
16D0 C4 00 31 C4 08 C8 34 C4
16D8 00 CD 01 B8 2E 9C F8 C4
16E0 80 36 C4 00 32 C4 1F 33
16E8 C2 00 94 FC 01 C2 00 94
16F0 02 90 FA 40 D4 0F C8 02
16F8 C3 00 C9 FD 40 D4 7F 1C
```

```
1700 1C 1C 1C 01 C3 80 C9 FE
1708 90 DE 55
```

```
     $ 0
16D0  C400  LDI 00
16D2  31    XPAL 1
16D3  C408  LDI 08
16D5  C834  ST 34 =170A
     $ 1
16D7  C400  LDI 00
16D9  CD01  ST @01 (1)
16DB  B82E  DLD 2E =170A
16DD  9CF8  JNZ $ 1
16DF  C480  LDI 80
16E1  36    XPAH 2
16E2  C400  LDI 00
16E4  32    XPAL 2
16E5  C41F  LDI 1F
16E7  33    XPAL 3
     $ 2
16E8  C200  LD 00 (2)
16EA  94FC  JP $ 2
16EC  01    XAE
     $ 3
16ED  C200  LD 00 (2)
16EF  9402  JP $ 4
16F1  90FA  JMP $ 3
     $ 4
16F3  40    LDE
16F4  D40F  ANI 0F
16F6  C802  ST 02 =16F9
16F8  C380  LD 80 (3)
16FA  C9FD  ST FD (1)
16FC  40    LDE
16FD  D47F  ANI 7F
16FF  1C    SR
1700  1C    SR
1701  1C    SR
1702  1C    SR
1703  01    XAE
1704  C380  LD 80 (3)
1706  C9FE  ST FE (1)
1708  90DE  JMP $ 2
170A  55    BYTE
```

8) ASCII-TV
1710 - 178F

```
1710 C4 80 35 C4 00 31 C4 FC
1718 36 C4 00 32 C4 20 CE 01
1720 37 36 01 40 36 C4 F0 60
1728 9C F2 32 C4 FC 36 C2 00
1730 37 C4 23 CA 00 C1 00 94
1738 FC D4 7F 01 C1 00 94 02
1740 90 FA 37 CA 00 40 E4 19
1748 98 CC 40 E4 05 98 DB 40
1750 E4 0D 98 1D 40 E4 08 98
1758 20 40 D4 FC E4 1C 9C 22
1760 40 D4 08 01 C0 80 C8 02
1768 C6 00 90 C2 00 01 40 FF
1770 C0 32 D4 C0 32 C6 40 90
```

```
1778 B5 C6 FF 37 C4 20 CA 01
1780 90 AC 40 D4 60 98 A7 E4
1788 60 98 A3 40 CE 01 90 9E

     $ 0
1710 C480  LDI 80
1712 35    XPAH 1
1713 C400  LDI 00
1715 31    XPAL 1
     $ 1
1716 C4FC  LDI FC
1718 36    XPAH 2
1719 C400  LDI 00
171B 32    XPAL 2
     $ 2
171C C420  LDI 20
171E CE01  ST @01 (2)
1720 37    XPAH 3
1721 36    XPAH 2
1722 01    XAE
1723 40    LDE
1724 36    XPAH 2
1725 C4F0  LDI F0
1727 60    XRE
1728 9CF2  JNZ $ 2
     $ 3
172A 32    XPAL 2
172B C4FC  LDI FC
172D 36    XPAH 2
     $ 4
172E C200  LD 00 (2)
1730 37    XPAH 3
1731 C423  LDI 23
1733 CA00  ST 00 (2)
     $ 5
1735 C100  LD 00 (1)
1737 94FC  JP $ 5
1739 D47F  ANI 7F
173B 01    XAE
     $ 6
173C C100  LD 00 (1)
173E 9402  JP $ 7
1740 98FA  JMP $ 6
     $ 7
1742 37    XPAH 3
1743 CA00  ST 00 (2)
1745 40    LDE
1746 E419  XRI 19
1748 98CC  JZ $ 1
174A 40    LDE
174B E405  XRI 05
174D 98DB  JZ $ 3
174F 40    LDE
1750 E40D  XRI 0D
1752 981D  JZ $ 8
1754 40    LDE
1755 E408  XRI 08
1757 9820  JZ $ 9
1759 40    LDE
175A D4FC  ANI FC
175C E41C  XRI 1C
175E 9C22  JNZ $ 10

1760 40    LDE
1761 D40B  ANI 0B
1763 01    XAE
1764 C080  LD 80 =16E5
1766 C802  ST 02 =1769
1768 C600  LD @00 (2)
176A 90C2  JMP $ 4
     TAB:
176C 00 01 40 FF C0
     $ 8
1771 32    XPAL 2
1772 D4C0  ANI C0
1774 32    XPAL 2
1775 C640  LD @40 (2)
1777 90B5  JMP $ 4
     $ 9
1779 C6FF  LD @FF (2)
177B 37    XPAH 3
177C C420  LDI 20
177E CA01  ST 01 (2)
1780 90AC  JMP $ 4
     $ 10
1782 40    LDE
1783 D460  ANI 60
1785 98A7  JZ $ 4
1787 E460  XRI 60
1789 98A3  JZ $ 4
178B 40    LDE
178C CE01  ST @01 (2)
178E 909E  JMP $ 4
```

9) SI-DRUCK AUF TV

1790 - 1826

```
1790 C4 00 35 C4 00 31 C4 04
1798 C8 6B C4 00 32 C4 FC 36
17A0 C4 18 37 C4 06 33 C4 10
17A8 C8 5A C4 20 CE 01 CE 01
17B0 35 01 40 35 3F 31 01 40
17B8 31 3F C4 20 CE 01 CE 01
17C0 C5 01 01 3F C4 20 CE 01
17C8 C1 FF 94 06 C5 01 01 3F
17D0 90 06 C4 20 CE 01 CE 01
17D8 C4 20 CE 01 CE 01 CE 31
17E0 B8 22 9C C6 CE 10 B8 1D
17E8 E4 03 98 0D C0 17 9C AD
17F0 00 C0 13 35 C0 11 31 90
17F8 9D 35 C8 0A 35 31 C8 07
1800 31 90 9A 55 55 55 55 40
1808 1C 1C 1C 1C 02 F4 F6 94
1810 02 F4 F9 F4 40 CE 01 06
1818 E4 01 07 D4 0F 98 05 40
1820 D4 0F 90 E8 3F 90 E0
```

```
     $ 0
1790 C400  LDI 00
1792 35    XPAH 1
1793 C400  LDI 00
1795 31    XPAL 1

     $ 1
1796 C404  LDI 04
1798 C86B  ST 6B =1804
179A C400  LDI 00
179C 32    XPAL 2
     $ 2
179D C4FC  LDI FC
179F 36    XPAH 2
17A0 C418  LDI 18
17A2 37    XPAH 3
17A3 C406  LDI 06
17A5 33    XPAL 3
17A6 C410  LDI 10
17A8 C85A  ST 5A =1803
     $ 3
17AA C420  LDI 20
17AC CE01  ST @01 (2)
17AE CE01  ST @01 (2)
17B0 35    XPAH 1
17B1 01    XAE
17B2 40    LDE
17B3 35    XPAH 1
17B4 3F    XPPC 3
17B5 31    XPAL 1
17B6 01    XAE
17B7 40    LDE
17B8 31    XPAL 1
17B9 3F    XPPC 3
17BA C420  LDI 20
17BC CE01  ST @01 (2)
17BE CE01  ST @01 (2)
17C0 C501  LD @01 (1)
17C2 01    XAE
17C3 3F    XPPC 3
17C4 C420  LDI 20
17C6 CE01  ST @01 (2)
17C8 C1FF  LD FF (1)
17CA 9406  JP $ 4
17CC C501  LD @01 (1)
17CE 01    XAE
17CF 3F    XPPC 3
17D0 9006  JMP $ 5
     $ 4
17D2 C420  LDI 20
17D4 CE01  ST @01 (2)
17D6 CE01  ST @01 (2)
     $ 5
17D8 C420  LDI 20
17DA CE01  ST @01 (2)
17DC CE01  ST @01 (2)
17DE CE31  ST @31 (2)
17E0 B822  DLD 22 =1803
17E2 9CC6  JNZ $ 3
17E4 CE10  ST @10 (2)
17E6 B81D  DLD 1D =1804
17E8 E403  XRI 03
17EA 980D  JZ $ 6
17EC C017  LD 17 =1804
17EE 9CAD  JNZ $ 2
17F0 00    HALT
17F1 C013  LD 13 =1805
17F3 35    XPAH 1
```

```
17F4  C011  LD 11 =1806
17F6  31    XPAL 1
17F7  909D  JMP $ 1
      $ 6
17F9  35    XPAH 1
17FA  C80A  ST 0A =1805
17FC  35    XPAH 1
17FD  31    XPAL 1
17FE  C807  ST 07 =1806
1800  31    XPAL 1
1801  909A  JMP $ 2
1803  55    BYTE
1804  55    BYTE
1805  55    BYTE
1806  55    BYTE
      $ 7
1807  40    LDE
1808  1C    SR
1809  1C    SR
180A  1C    SR
180B  1C    SR
      $ 8
180C  02    CCL
180D  F4F6  ADI F6
180F  9402  JP $ 9
1811  F4F9  ADI F9
      $ 9
1813  F440  ADI 40
1815  CE01  ST @01 (2)
1817  06    CSA
1818  E401  XRI 01
181A  07    CAS
181B  D40F  ANI 0F
181D  9805  JZ $ 10
181F  40    LDE
1820  D40F  ANI 0F
1822  90E8  JMP $ 8
      $ 10
1824  3F    XPPC 3
1825  90E0  JMP $ 7
```

10) TV LOESCHEN
1830 - 1843

```
1830 C4 FC 37 C4 00 33 C4 20
1838 CF 01 37 01 40 37 40 E4
1840 F0 9C F3 00
```

```
      $ 0
1830  C4FC  LDI FC
1832  37    XPAH 3
1833  C400  LDI 00
1835  33    XPAL 3
      $ 1
1836  C420  LDI 20
1838  CF01  ST @01 (3)
183A  37    XPAH 3
183B  01    XAE
183C  40    LDE
183D  37    XPAH 3
183E  40    LDE
183F  E4F0  XRI F0
1841  9CF3  JNZ $ 1
1843  00    HALT
```

11) TV-DEMO 1
1850 - 1866

```
1850 C4 FC 35 C4 00 CB 10 31
1858 A8 0D CD 01 35 01 40 35
1860 C4 F0 60 9C F3 00 55
```

```
      $ 0
1850  C4FC  LDI FC
1852  35    XPAH 1
1853  C400  LDI 00
1855  C810  ST 10 =1866
1857  31    XPAL 1
      $ 1
1858  A80D  ILD 0D =1866
185A  CD01  ST @01 (1)
185C  35    XPAH 1
185D  01    XAE
185E  40    LDE
185F  35    XPAH 1
1860  C4F0  LDI F0
1862  60    XRE
1863  9CF3  JNZ $ 1
1865  00    HALT
1866  55    BYTE
```

12) TV-DEMO 2
1870 - 188F

```
1870 C4 00 C8 1C 32 C4 FC 36
1878 36 01 40 36 C4 F0 60 9C
1880 08 B8 0D B8 0B 8F 40 90
1888 EC A8 05 CE 01 90 E9 55
```

```
      $ 0
1870  C400  LDI 00
1872  C81C  ST 1C =188F
1874  32    XPAL 2
      $ 1
1875  C4FC  LDI FC
1877  36    XPAH 2
      $ 2
1878  36    XPAH 2
1879  01    XAE
187A  40    LDE
187B  36    XPAH 2
187C  C4F0  LDI F0
187E  60    XRE
187F  9C08  JNZ $ 3
1881  B80D  DLD 0D =188F
1883  B80B  DLD 0B =188F
1885  8F40  DLY 40
1887  90EC  JMP $ 1
      $ 3
1889  A805  ILD 05 =188F
188B  CE01  ST @01 (2)
188D  90E9  JMP $ 2
188F  55    BYTE
```

13) TERMINAL-RAM-TERMINALL
1890 - 18CF

```
1890 C4 80 36 C4 00 32 C4 36
1898 CA 0B C4 14 CA 08 C4 00
18A0 CA 08 C4 FF CA 05 C4 05
18A8 CA A5 C4 0C 37 C4 00 33
18B0 C2 05 D4 02 98 FA C2 04
18B8 D4 7F CF 01 37 C8 51 37
18C0 33 C8 4E 33 C2 05 D4 01
18C8 98 FA C3 FF CA 04 90 E0
```

```
      $ 0
1890  C480  LDI 80
1892  36    XPAH 2
1893  C400  LDI 00
1895  32    XPAL 2
1896  C436  LDI 36
1898  CA0B  ST 0B (2)
189A  C414  LDI 14
189C  CA08  ST 08 (2)
189E  C400  LDI 00
18A0  CA08  ST 08 (2)

18A2  C4FF  LDI FF
18A4  CA05  ST 05 (2)
18A6  C405  LDI 05
18A8  CA05  ST 05 (2)
18AA  C40C  LDI 0C
18AC  37    XPAH 3
18AD  C400  LDI 00
18AF  33    XPAL 3
      $ 1
18B0  C205  LD 05 (2)
18B2  D402  ANI 02
18B4  98FA  JZ $ 1
18B6  C204  LD 04 (2)
18B8  D47F  ANI 7F
18BA  CF01  ST @01 (3)
18BC  37    XPAH 3
18BD  C851  ST 51 =190F
18BF  37    XPAH 3
18C0  33    XPAL 3
18C1  C84E  ST 4E =1910
18C3  33    XPAL 3
      $ 2
18C4  C205  LD 05 (2)
18C6  D401  ANI 01
18C8  98FA  JZ $ 2
18CA  C3FF  LD FF (3)
18CC  CA04  ST 04 (2)
18CE  90E0  JMP $ 1
```

```
14)  RAM-TERMINAL
18D0 - 1910

18D0 C4 80 36 C4 00 32 C4 36
18D8 CA 0B C4 14 CA 08 C4 00
18E0 CA 08 C4 FF CA 05 C4 01
18E8 CA 05 C4 0C 37 C4 00 33
18F0 C2 05 D4 01 98 FA C7 01
18F8 CA 04 37 01 40 37 C0 10
1900 60 9C ED 33 01 40 33 C0
1908 08 60 9C E4 08 90 DB 55
1910 55

      $  0
18D0  C480  LDI 80
18D2  36    XPAH 2
18D3  C400  LDI 00
18D5  32    XPAL 2
18D6  C436  LDI 36
18D8  CA0B  ST 0B (2)
18DA  C414  LDI 14
18DC  CA08  ST 08 (2)
18DE  C400  LDI 00
18E0  CA08  ST 08 (2)
18E2  C4FF  LDI FF
18E4  CA05  ST 05 (2)
18E6  C401  LDI 01
18E8  CA05  ST 05 (2)
      $  1
18EA  C40C  LDI 0C
18EC  37    XPAH 3
18ED  C400  LDI 00
18EF  33    XPAL 3
      $  2
18F0  C205  LD 05 (2)
18F2  D401  ANI 01
18F4  98FA  JZ $  2
18F6  C701  LD @01 (3)
18F8  CA04  ST 04 (2)
18FA  37    XPAH 3
18FB  01    XAE
18FC  40    LDE
18FD  37    XPAH 3
18FE  C010  LD 10 =190F
1900  60    XRE
1901  9CED  JNZ $  2
1903  33    XPAL 3
1904  01    XAE
1905  40    LDE
1906  33    XPAL 3
1907  C008  LD 08 =1910
1909  60    XRE
190A  9CE4  JNZ $  2
190C  08    NOP
190D  90DB  JMP $  1
190F  55    BYTE
1910  55    BYTE
```

```
1)  A5-DIAGNOSE
1000 - 1228  ( 0C00 - 0E28 )

0C00 C4 5E C9 06 C4 3D C9 05
0C08 C4 00 C9 FF C9 00 C4 80
0C10 C9 04 C9 03 C9 02 C9 01
0C18 C4 3E CA 1D C4 00 37 C4
0C20 55 33 3F C2 01 CA 0A CA
0C28 10 C2 02 CA 0B CA 11 3F
0C30 C2 01 CA 13 C2 02 CA 14
0C38 C4 FF CA 12 C2 10 CA 0E
0C40 C2 11 CA 0F C4 0D 37 C4
0C48 B4 33 3F C4 55 CA 12 C2
0C50 0A CA 0E C2 0B CA 0F C2
0C58 01 CA 13 C2 0B CA 14 C4
0C60 0D 37 C4 B4 33 3F C2 0E
0C68 E2 01 9C 08 C2 0F E2 02
0C70 9C 02 90 22 C2 01 CA 13
0C78 C2 02 CA 14 02 C2 0E F4
0C80 01 CA 0A CA 0C C2 0F F4
0C88 00 CA 0B CA 0D C4 0D 37
0C90 C4 DC 33 3F 98 A2 C4 00
0C98 CA 12 C2 10 CA 0E C2 11
0CA0 CA 0F C4 0D 37 C4 B4 33
0CA8 3F C2 10 CA 0C C2 11 CA
0CB0 0D C4 0D 37 C4 DC 33 3F
0CB8 C4 FF CA 12 C2 10 CA 0E
0CC0 C2 11 CA 0F C4 0D 37 C4
0CC8 B4 33 3F C2 10 CA 0C C2
0CD0 11 CA 0D C4 0D 37 C4 DC
0CD8 33 3F C2 01 CA 13 C2 02
0CE0 CA 14 C4 AA CA 12 90 1B
0CE8 C2 0C CA 10 C2 0D CA 11
0CF0 C2 12 E4 55 9C 0A C2 0A
0CF8 CA 13 C2 0F CA 14 90 03
0D00 40 CA 14 C4 01 31 C4 07
0D08 35 C2 12 01 40 E4 AA 9C
0D10 0A C4 54 C9 00 C4 5C C9
0D18 FF 90 24 40 E4 55 9C 0A
0D20 C4 86 C9 00 C4 00 C9 FF
0D28 90 15 40 98 0A C4 4F C9
0D30 00 C4 00 C9 FF 90 08 C4
0D38 5B C9 00 C4 00 C9 FF C4
0D40 79 C9 06 C4 50 C9 05 C9
0D48 04 C9 02 C4 5C C9 03 C4
0D50 00 C9 01 C4 0E 37 C4 1C
0D58 33 3F C4 A0 CA 1D C2 10
0D60 CA 01 C2 11 CA 02 C4 00
0D68 37 C4 55 33 3F C4 00 C9
0D70 FF C9 00 C9 05 C9 06 C4
0D78 0E 37 C4 1C 33 3F C2 13
0D80 CA 01 C2 14 CA 02 C4 00
0D88 37 C4 55 33 3F C4 00 C9
0D90 FF C9 00 40 E4 AA 98 0B
0D98 40 E4 55 98 06 C4 00 C9
0DA0 01 C9 02 8F FF 8F FF C4
0DA8 0E 37 C4 1C 33 3F C4 0D
0DB0 37 C4 02 33 3F C2 0E 31
0DB8 C2 0F 35 C2 12 C9 00 C2
0DC0 0E E2 13 9C 06 C2 0F E2
0DC8 14 98 0F 02 C2 0E F4 01
0DD0 CA 0E C2 0F F4 00 CA 0F
0DD8 90 DB 3F 90 D8 C2 0C 31
0DE0 C2 0D 35 C1 00 E2 12 C9
0DE8 00 01 C2 12 E4 55 98 05
0DF0 40 98 0C 90 03 40 9C 07
0DF8 C4 0C 37 C4 E7 33 3F C2
0E00 0C E2 01 9C 09 C2 0D E2
0E08 02 9C 03 3F 90 CF 02 C2
0E10 0C F4 01 CA 0C C2 0D F4
0E18 00 CA 0D 90 C0 C4 08 C8
0E20 08 8F FF B8 04 9C FA 3F
0E28 55

      $  0
0C00  C45E  LDI 5E
0C02  C906  ST 06 (1)
0C04  C43D  LDI 3D
0C06  C905  ST 05 (1)
0C08  C400  LDI 00
0C0A  C9FF  ST FF (1)
0C0C  C900  ST 00 (1)
0C0E  C480  LDI 80
0C10  C904  ST 04 (1)
0C12  C903  ST 03 (1)
0C14  C902  ST 02 (1)
0C16  C901  ST 01 (1)
0C18  C43E  LDI 3E
0C1A  CA1D  ST 1D (2)
0C1C  C400  LDI 00
0C1E  37    XPAH 3
0C1F  C455  LDI 55
0C21  33    XPAL 3
0C22  3F    XPPC 3
0C23  C201  LD 01 (2)
0C25  CA0A  ST 0A (2)
0C27  CA10  ST 10 (2)
0C29  C202  LD 02 (2)
0C2B  CA0B  ST 0B (2)
0C2D  CA11  ST 11 (2)
0C2F  3F    XPPC 3
0C30  C201  LD 01 (2)
0C32  CA13  ST 13 (2)
0C34  C202  LD 02 (2)
0C36  CA14  ST 14 (2)
      $  1
0C38  C4FF  LDI FF
0C3A  CA12  ST 12 (2)
0C3C  C210  LD 10 (2)
0C3E  CA0E  ST 0E (2)
0C40  C211  LD 11 (2)
0C42  CA0F  ST 0F (2)
0C44  C40D  LDI 0D
0C46  37    XPAH 3
0C47  C4B4  LDI B4
0C49  33    XPAL 3
0C4A  3F    XPPC 3
0C4B  C455  LDI 55
0C4D  CA12  ST 12 (2)
0C4F  C20A  LD 0A (2)
0C51  CA0E  ST 0E (2)
0C53  C20B  LD 0B (2)
0C55  CA0F  ST 0F (2)
0C57  C201  LD 01 (2)
```

```
0C59  CA13  ST 13 (2)
0C5B  C20B  LD 0B (2)
0C5D  CA14  ST 14 (2)
0C5F  C40D  LDI 0D
0C61  37    XPAH 3
0C62  C4B4  LDI B4
0C64  33    XPAL 3
0C65  3F    XPPC 3
0C66  C20E  LD 0E (2)
0C68  E201  XOR 01 (2).
0C6A  9C08  JNZ $ 2
0C6C  C20F  LD 0F (2)
0C6E  E202  XOR 02 (2)
0C70  9C02  JNZ $ 2
0C72  9022  JMP $ 3
      $ 2
0C74  C201  LD 01 (2)
0C76  CA13  ST 13 (2)
0C78  C202  LD 02 (2)
0C7A  CA14  ST 14 (2)
0C7C  02    CCL
0C7D  C20E  LD 0E (2)
0C7F  F401  ADI 01
0C81  CA0A  ST 0A (2)
0C83  CA0C  ST 0C (2)
0C85  C20F  LD 0F (2)
0C87  F400  ADI 00
0C89  CA0B  ST 0B (2)
0C8B  CA0D  ST 0D (2)
0C8D  C40D  LDI 0D
0C8F  37    XPAH 3
0C90  C4DC  LDI DC
0C92  33    XPAL 3
0C93  3F    XPPC 3
0C94  90A2  JMP $ 1
      $ 3
0C96  C400  LDI 00
0C98  CA12  ST 12 (2)
0C9A  C210  LD 10 (2)
0C9C  CA0E  ST 0E (2)
0C9E  C211  LD 11 (2)
0CA0  CA0F  ST 0F (2)
0CA2  C40D  LDI 0D
0CA4  37    XPAH 3
0CA5  C4B4  LDI B4
0CA7  33    XPAL 3
0CA8  3F    XPPC 3
0CA9  C210  LD 10 (2)
0CAB  CA0C  ST 0C (2)
0CAD  C211  LD 11 (2)
0CAF  CA0D  ST 0D (2)
0CB1  C40D  LDI 0D
0CB3  37    XPAH 3
0CB4  C4DC  LDI DC
0CB6  33    XPAL 3
0CB7  3F    XPPC 3
0CB8  C4FF  LDI FF
0CBA  CA12  ST 12 (2)
0CBC  C210  LD 10 (2)
0CBE  CA0E  ST 0E (2)
0CC0  C211  LD 11 (2)
0CC2  CA0F  ST 0F (2)
0CC4  C40D  LDI 0D
0CC6  37    XPAH 3
0CC7  C4B4  LDI B4
0CC9  33    XPAL 3
0CCA  3F    XPPC 3
0CCB  C210  LD 10 (2)
0CCD  CA0C  ST 0C (2)
0CCF  C211  LD 11 (2)
0CD1  CA0D  ST 0D (2)
0CD3  C40D  LDI 0D
0CD5  37    XPAH 3
0CD6  C4DC  LDI DC
0CD8  33    XPAL 3
0CD9  3F    XPPC 3
0CDA  C201  LD 01 (2)
0CDC  CA13  ST 13 (2)
0CDE  C202  LD 02 (2)
0CE0  CA14  ST 14 (2)
0CE2  C4AA  LDI AA
0CE4  CA12  ST 12 (2)
0CE6  901B  JMP $ 5
0CE8  C20C  LD 0C (2)
0CEA  CA10  ST 10 (2)
0CEC  C20D  LD 0D (2)
0CEE  CA11  ST 11 (2)
0CF0  C212  LD 12 (2)
0CF2  E455  XRI 55
0CF4  9C0A  JNZ $ 4
0CF6  C20A  LD 0A (2)
0CF8  CA13  ST 13 (2)
0CFA  C20F  LD 0F (2)
0CFC  CA14  ST 14 (2)
0CFE  9003  JMP $ 5
      $ 4
0D00  40    LDE
0D01  CA14  ST 14 (2)
      $ 5
0D03  C401  LDI 01
0D05  31    XPAL 1
0D06  C407  LDI 07
0D08  35    XPAH 1
0D09  C212  LD 12 (2)
0D0B  01    XAE
0D0C  40    LDE
0D0D  E4AA  XRI AA
0D0F  9C0A  JNZ $ 6
0D11  C454  LDI 54
0D13  C900  ST 00 (1)
0D15  C45C  LDI 5C
0D17  C9FF  ST FF (1)
0D19  9024  JMP $ 9
      $ 6
0D1B  40    LDE
0D1C  E455  XRI 55
0D1E  9C0A  JNZ $ 7
0D20  C406  LDI 06
0D22  C900  ST 00 (1)
0D24  C400  LDI 00
0D26  C9FF  ST FF (1)
0D28  9015  JMP $ 9
      $ 7
0D2A  40    LDE
0D2B  980A  JZ $ 8
0D2D  C44F  LDI 4F
0D2F  C900  ST 00 (1)
0D31  C400  LDI 00
0D33  C9FF  ST FF (1)
0D35  9008  JMP $ 9
      $ 8
0D37  C45B  LDI 5B
0D39  C900  ST 00 (1)
0D3B  C400  LDI 00
0D3D  C9FF  ST FF (1)
      $ 9
0D3F  C479  LDI 79
0D41  C906  ST 06 (1)
0D43  C450  LDI 50
0D45  C905  ST 05 (1)
0D47  C904  ST 04 (1)
0D49  C902  ST 02 (1)
0D4B  C45C  LDI 5C
0D4D  C903  ST 03 (1)
0D4F  C400  LDI 00
0D51  C901  ST 01 (1)
0D53  C40E  LDI 0E
0D55  37    XPAH 3
0D56  C41C  LDI 1C
0D58  33    XPAL 3
0D59  3F    XPPC 3
0D5A  C4A8  LDI A8
0D5C  CA1D  ST 1D (2)
0D5E  C210  LD 10 (2)
0D60  CA01  ST 01 (2)
0D62  C211  LD 11 (2)
0D64  CA02  ST 02 (2)
0D66  C400  LDI 00
0D68  37    XPAH 3
0D69  C455  LDI 55
0D6B  33    XPAL 3
0D6C  3F    XPPC 3
0D6D  C400  LDI 00
0D6F  C9FF  ST FF (1)
0D71  C900  ST 00 (1)
0D73  C905  ST 05 (1)
0D75  C906  ST 06 (1)
0D77  C40E  LDI 0E
0D79  37    XPAH 3
0D7A  C41C  LDI 1C
0D7C  33    XPAL 3
0D7D  3F    XPPC 3
0D7E  C213  LD 13 (2)
0D80  CA01  ST 01 (2)
0D82  C214  LD 14 (2)
0D84  CA02  ST 02 (2)
0D86  C400  LDI 00
0D88  37    XPAH 3
0D89  C455  LDI 55
0D8B  33    XPAL 3
0D8C  3F    XPPC 3
0D8D  C400  LDI 00
0D8F  C9FF  ST FF (1)
0D91  C900  ST 00 (1)
0D93  40    LDE
0D94  E4AA  XRI AA
```

```
0D96 980B JZ $ 10
0D98 40   LDE
0D99 E455 XRI 55
0D9B 9806 JZ $ 10
0D9D C400 LDI 00
0D9F C901 ST 01 (1)
0DA1 C902 ST 02 (1)
     $ 10
0DA3 8FFF DLY FF
0DA5 8FFF DLY FF
0DA7 C40E LDI 0E
0DA9 37   XPAH 3
0DAA C41C LDI 1C
0DAC 33   XPAL 3
0DAD 3F   XPPC 3
0DAE C40D LDI 0D
0DB0 37   XPAH 3
0DB1 C402 LDI 02
0DB3 33   XPAL 3
0DB4 3F   XPPC 3
     $ 11
0DB5 C20E LD 0E (2)
0DB7 31   XPAL 1
0DB8 C20F LD 0F (2)
0DBA 35   XPAH 1
0DBB C212 LD 12 (2)
0DBD C900 ST 00 (1)
0DBF C20E LD 0E (2)
0DC1 E213 XOR 13 (2)
0DC3 9C06 JNZ $ 12
0DC5 C20F LD 0F (2)
0DC7 E214 XOR 14 (2)
0DC9 980F JZ $ 13
     $ 12
0DCB 02   CCL
0DCC C20E LD 0E (2)
0DCE F401 ADI 01
0DD0 CA0E ST 0E (2)
0DD2 C20F LD 0F (2)
0DD4 F400 ADI 00
0DD6 CA0F ST 0F (2)
0DD8 900B JMP $ 11
     $ 13
0DDA 3F   XPPC 3
0DDB 9008 JMP $ 11
     $ 14
0DDD C20C LD 0C (2)
0DDF 31   XPAL 1
0DE0 C20D LD 0D (2)
0DE2 35   XPAH 1
0DE3 C100 LD 00 (1)
0DE5 E212 XOR 12 (2)
0DE7 C900 ST 00 (1)
0DE9 01   XPE
0DEA C212 LD 12 (2)
0DEC E455 XRI 55
0DEE 9805 JZ $ 15
0DF0 40   LDE
0DF1 9800 JZ $ 17
0DF3 9003 JMP $ 16
     $ 15
0DF5 40   LDE
0DF6 9C07 JNZ $ 17
     $ 16
0DF8 C40C LDI 0C
0DFA 37   XPAH 3
0DFB C4E7 LDI E7
0DFD 33   XPAL 3
0DFE 3F   XPPC 3
     $ 17
0DFF C20C LD 0C (2)
0E01 E201 XOR 01 (2)
0E03 9C09 JNZ $ 18
0E05 C20D LD 0D (2)
0E07 E202 XOR 02 (2)
0E09 9C03 JNZ $ 18
0E0B 3F   XPPC 3
0E0C 98CF JMP $ 14
     $ 18
0E0E 02   CCL
0E0F C20C LD 0C (2)
0E11 F401 ADI 01
0E13 CA0C ST 0C (2)
0E15 C20D LD 0D (2)
0E17 F400 ADI 00
0E19 CA0D ST 0D (2)
0E1B 98C0 JMP $ 14
0E1D C408 LDI 08
0E1F C808 ST 08 =0E28
     $ 19
0E21 8FFF DLY FF
0E23 B804 DLD 04 =0E28
0E25 9CFA JNZ $ 19
0E27 3F   XPPC 3
0E28 55   BYTE
```

2) COMPARE
1230 - 12E3

```
1230 C4 08 CA 00 31 C4 00 CD
1238 FF BA 00 9C F8 C4 58 C9
1240 07 C4 5C C9 06 C4 3E CA
1248 1D C4 00 37 C4 55 33 3F
1250 C2 01 CA 10 C2 02 CA 11
1258 3F C2 01 CA 12 C2 02 CA
1260 13 3F C2 01 CA 14 C2 02
1268 CA 15 C2 10 31 C2 11 35
1270 C2 14 33 C2 15 37 C1 00
1278 E3 00 9C 23 C2 10 E2 12
1280 9C 06 C2 11 E2 13 98 47
1288 C5 01 31 CA 10 35 CA 11
1290 02 C2 14 F4 01 CA 14 C2
1298 15 F4 00 CA 15 90 CB C1
12A0 00 CA 00 C2 10 CA 01 C2
12A8 11 CA 02 C4 A0 CA 1D C4
12B0 00 37 C4 55 33 3F 00 C2
12B8 14 CA 01 33 C2 15 CA 02
12C0 37 C3 00 CA 00 C4 00 37
12C8 C4 55 33 3F 00 90 AD C4
12D0 01 31 C4 07 35 C4 5C C9
12D8 05 C4 54 C9 06 C4 03 37
```

```
12E0 C4 D4 33 3F
     $ 0
1230 C408 LDI 08
1232 CA00 ST 00 (2)
1234 31   XPAL 1
     $ 1
1235 C400 LDI 00
1237 CDFF ST @FF (1)
1239 BA00 DLD 00 (2)
123B 9CF8 JNZ $ 1
123D C458 LDI 58
123F C907 ST 07 (1)
1241 C45C LDI 5C
1243 C906 ST 06 (1)
1245 C43E LDI 3E
1247 CA1D ST 1D (2)
1249 C400 LDI 00
124B 37   XPAH 3
124C C455 LDI 55
124E 33   XPAL 3
124F 3F   XPPC 3
1250 C201 LD 01 (2)
1252 CA10 ST 10 (2)
1254 C202 LD 02 (2)
1256 CA11 ST 11 (2)
1258 3F   XPPC 3
1259 C201 LD 01 (2)
125B CA12 ST 12 (2)
125D C202 LD 02 (2)
125F CA13 ST 13 (2)
1261 3F   XPPC 3
1262 C201 LD 01 (2)
1264 CA14 ST 14 (2)
1266 C202 LD 02 (2)
1268 CA15 ST 15 (2)
     $ 2
126A C210 LD 10 (2)
126C 31   XPAL 1
126D C211 LD 11 (2)
126F 35   XPAH 1
1270 C214 LD 14 (2)
1272 33   XPAL 3
1273 C215 LD 15 (2)
1275 37   XPAH 3
1276 C100 LD 00 (1)
1278 E300 XOR 00 (3)
127A 9C23 JNZ $ 5
     $ 3
127C C210 LD 10 (2)
127E E212 XOR 12 (2)
1280 9C06 JNZ $ 4
1282 C211 LD 11 (2)
1284 E213 XOR 13 (2)
1286 9847 JZ $ 6
     $ 4
1288 C501 LD @01 (1)
128A 31   XPAL 1
128B CA10 ST 10 (2)
128D 35   XPAH 1
128E CA11 ST 11 (2)
1290 02   CCL
```

```
1291  C214  LD 14 (2)
1293  F401  ADI 01
1295  CA14  ST 14 (2)
1297  C215  LD 15 (2)
1299  F400  ADI 00
129B  CA15  ST 15 (2)
129D  90CB  JMP $ 2
      $ 5
129F  C100  LD 00 (1)
12A1  CA00  ST 00 (2)
12A3  C210  LD 10 (2)
12A5  CA01  ST 01 (2)
12A7  C211  LD 11 (2)
12A9  CA02  ST 02 (2)
12AB  C4A0  LDI A0
12AD  CA1D  ST 1D (2)
12AF  C400  LDI 00
12B1  37    XPAH 3
12B2  C455  LDI 55
12B4  33    XPAL 3
12B5  3F    XPPC 3
12B6  00    HALT
12B7  C214  LD 14 (2)
12B9  CA01  ST 01 (2)
12BB  33    XPAL 3
12BC  C215  LD 15 (2)
12BE  CA02  ST 02 (2)
12C0  37    XPAH 3
12C1  C300  LD 00 (3)
12C3  CA00  ST 00 (2)
12C5  C400  LDI 00
12C7  37    XPAH 3
12C8  C455  LDI 55
12CA  33    XPAL 3
12CB  3F    XPPC 3
12CC  00    HALT
12CD  90AD  JMP $ 3
      $ 6
12CF  C401  LDI 01
12D1  31    XPAL 1
12D2  C407  LDI 07
12D4  35    XPAH 1
12D5  C45C  LDI 5C
12D7  C905  ST 05 (1)
12D9  C454  LDI 54
12DB  C906  ST 06 (1)
12DD  C403  LDI 03
12DF  37    XPAH 3
12E0  C4D4  LDI D4
12E2  33    XPAL 3
12E3  3F    XPPC 3
```

3) SUB-HEX
1300 - 136E

```
1300 C4 08 31 C4 40 C9 FD C4
1308 40 C9 FA C4 80 C9 FF C9
1310 FE C9 FC C9 FB C9 F9 C9
1318 F8 C4 1F 33 C4 13 36 C4
1320 6E 32 C4 04 C8 49 C1 07
1328 94 FC 01 8F 0A C1 07 E4
1330 FF 94 FA 8F 0A 40 E4 D0
1338 98 C6 B8 33 E4 01 9C 02
1340 C5 FF 40 D4 0F CE FF 01
1348 C3 80 CD FF C0 21 9C D6
1350 C5 FD 03 C2 02 FE 01 D4
1358 0F 01 C3 80 CD 01 A8 0F
1360 E4 02 9C EF C1 08 94 FC
1368 90 96 55 55 55 55 55
```

```
      $ 0
1300  C408  LDI 08
1302  31    XPAL 1
1303  C440  LDI 40
1305  C9FD  ST FD (1)
1307  C448  LDI 48
1309  C9FA  ST FA (1)
130B  C480  LDI 80
130D  C9FF  ST FF (1)
130F  C9FE  ST FE (1)
1311  C9FC  ST FC (1)
1313  C9FB  ST FB (1)
1315  C9F9  ST F9 (1)
1317  C9F8  ST F8 (1)
1319  C41F  LDI 1F
131B  33    XPAL 3
131C  C413  LDI 13
131E  36    XPAH 2
131F  C46E  LDI 6E
1321  32    XPAL 2
1322  C404  LDI 04
1324  C849  ST 49 =136E
      $ 1
1326  C107  LD 07 (1)
1328  94FC  JP $ 1
132A  01    XAE
132B  8F0A  DLY 0A
      $ 2
132D  C107  LD 07 (1)
132F  E4FF  XRI FF
1331  94FA  JP $ 2
1333  8F0A  DLY 0A
1335  40    LDE
1336  E4D0  XRI D0
1338  98C6  JZ $ 0
133A  B833  DLD 33 =136E
133C  E401  XRI 01
133E  9C02  JNZ $ 3
1340  C5FF  LD @FF (1)
      $ 3
1342  40    LDE
1343  D40F  ANI 0F
1345  CEFF  ST @FF (2)
1347  01    XAE
1348  C380  LD 80 (3)
134A  CDFF  ST @FF (1)
134C  C021  LD 21 =136E
134E  9CD6  JNZ $ 1
1350  C5FD  LD @FD (1)
1352  03    SCL
      $ 4
1353  C202  LD 02 (2)
1355  FE01  CAD @01 (2)
1357  D40F  ANI 0F
1359  01    XAE
135A  C380  LD 80 (3)
135C  CD01  ST @01 (1)
135E  A80F  ILD 0F =136E
1360  E402  XRI 02
1362  9CEF  JNZ $ 4
      $ 5
1364  C108  LD 08 (1)
1366  94FC  JP $ 5
1368  9096  JMP $ 0
136A  55    BYTE
136B  55    BYTE
136C  55    BYTE
136D  55    BYTE
136E  55    BYTE
```

4) HEX-MATRIX
1370 - 1407

```
1370 C4 73 C9 06 C4 50 C9 05
1378 C4 80 C9 04 C9 03 C9 02
1380 C9 01 C9 00 C9 FF C4 00
1388 CA 0F 37 C4 55 33 3F C2
1390 08 E4 B0 9C 43 C4 78 C9
1398 00 C4 10 C9 FF CA 0F C4
13A0 3E CA 1D C4 04 CA 0E C4
13A8 D3 CA 12 3F C2 02 D4 3F
13B0 DC 80 01 AA 12 01 CA 80
13B8 C2 01 D4 3F DC 80 01 AA
13C0 12 01 CA 90 BA 0E 9C E3
13C8 AA 12 01 C4 FF CA 80 90
13D0 02 CA 0F C4 0A CA 1D 3F
13D8 C4 5F C9 00 C4 5E C9 FF
13E0 C4 3E CA 1D 3F C2 02 CA
13E8 11 C2 01 CA 10 3F C4 FF
13F0 CA 1D C4 FF CA 1C C4 B4
13F8 CA 13 C4 0F CA 14 C2 0F
1400 90 01 3F C2 10 31 C2 11
1408 35 C4 D3 CA 12 35 CA 0D
1410 35 C4 02 CA 0E C4 14 37
1418 C4 8E 33 3F AA 12 01 C2
1420 0C DC 40 CA 80 AA 12 01
1428 C2 0B DC 40 CA 80 BA 0E
1430 90 86 31 CA 0D 31 90 DD
1438 AA 12 01 C4 08 CA 0E C4
1440 20 CA 80 C1 00 CA 0D 3F
1448 C2 0C 01 AA 12 01 CA 80
1450 C2 0B 01 AA 12 01 CA 80
1458 AA 12 01 31 E2 01 9C 0D
1460 E2 01 31 35 E2 02 98 1B
1468 E2 02 35 90 03 E2 01 31
1470 C5 01 BA 0E 9C C9 C4 FF
1478 CA 80 C4 00 37 C4 55 33
1480 3F 90 86 C4 FF CA 80 C4
1488 00 37 C4 55 33 3F 00 C4
1490 14 37 CA 11 C4 B8 33 CA
1498 10 C2 0D D4 F0 1E 1E 1E
14A0 1E 01 C3 80 CA 0C C2 0D
14A8 D4 0F 01 C3 80 CA 0B C2
14B0 11 37 C2 10 33 3F 90 D7
```

```
14B8 30 31 32 33 34 35 36 37
14C0 38 39 01 02 03 04 05 06

     $ 0
1370 C473 LDI 73
1372 C906 ST 06 (1)
1374 C450 LDI 50
1376 C905 ST 05 (1)
1378 C480 LDI 80
137A C904 ST 04 (1)
137C C903 ST 03 (1)
137E C902 ST 02 (1)
1380 C901 ST 01 (1)
1382 C900 ST 00 (1)
1384 C9FF ST FF (1)
1386 C400 LDI 00
1388 CA0F ST 0F (2)
138A 37   XPAH 3
138B C455 LDI 55
138D 33   XPAL 3
138E 3F   XPPC 3
138F C208 LD 08 (2)
1391 E4B0 XRI B0
1393 9C43 JNZ $ 3
1395 C478 LDI 78
1397 C900 ST 00 (1)
1399 C410 LDI 10
139B C9FF ST FF (1)
139D CA0F ST 0F (2)
139F C43E LDI 3E
13A1 CA1D ST 1D (2)
13A3 C404 LDI 04
13A5 CA0E ST 0E (2)
13A7 C4D3 LDI D3
13A9 CA12 ST 12 (2)
     $ 1
13AB 3F   XPPC 3
13AC C202 LD 02 (2)
13AE D43F ANI 3F
13B0 DC80 ORI 80
13B2 01   XAE
13B3 AA12 ILD 12 (2)
13B5 01   XAE
13B6 CA80 ST 80 (2)
13B8 C201 LD 01 (2)
13BA D43F ANI 3F

13BC DC90 ORI 90
13BE 01   XAE
13BF AA12 ILD 12 (2)
13C1 01   XAE
13C2 CA80 ST 80 (2)
13C4 BA0E DLD 0E (2)
13C6 9CE3 JNZ $ 1
13C8 AA12 ILD 12 (2)
13CA 01   XAE
13CB C4FF LDI FF
13CD CA80 ST 80 (2)
13CF 9002 JMP $ 2
13D1 CA0F ST 0F (2)
     $ 2
13D3 C40A LDI 0A
13D5 CA1D ST 1D (2)
13D7 3F   XPPC 3
     $ 3
13D8 C45F LDI 5F
13DA C980 ST 80 (1)
13DC C45E LDI 5E
13DE C9FF ST FF (1)
13E0 C43E LDI 3E
13E2 CA1D ST 1D (2)
13E4 3F   XPPC 3
13E5 C202 LD 02 (2)
13E7 CA11 ST 11 (2)
13E9 C201 LD 01 (2)
13EB CA10 ST 10 (2)
13ED 3F   XPPC 3
13EE C4FF LDI FF
13F0 CA1D ST 1D (2)
13F2 C4FF LDI FF
13F4 CA1C ST 1C (2)
13F6 C4B4 LDI B4
13F8 CA13 ST 13 (2)
13FA C40F LDI 0F
13FC CA14 ST 14 (2)
13FE C20F LD 0F (2)
1400 9801 JZ $ 4
1402 3F   XPPC 3
     $ 4
1403 C210 LD 10 (2)
1405 31   XPAL 1
1406 C211 LD 11 (2)
1408 35   XPAH 1
     $ 5
1409 C4D3 LDI D3
140B CA12 ST 12 (2)
140D 35   XPAH 1
140E CA0D ST 0D (2)
1410 35   XPAH 1
1411 C402 LDI 02
1413 CA0E ST 0E (2)
     $ 6
1415 C414 LDI 14
1417 37   XPAH 3
1418 C48E LDI 8E
141A 33   XPAL 3
141B 3F   XPPC 3
141C AA12 ILD 12 (2)
141E 01   XAE
141F C20C LD 0C (2)
1421 DC40 ORI 40
1423 CA80 ST 80 (2)
1425 AA12 ILD 12 (2)
1427 01   XAE
1428 C20B LD 0B (2)
142A DC40 ORI 40
142C CA80 ST 80 (2)
142E BA0E DLD 0E (2)
1430 9806 JZ $ 7
1432 31   XPAL 1
1433 CA0D ST 0D (2)
1435 31   XPAL 1
1436 98DD JMP $ 6
     $ 7
1438 AA12 ILD 12 (2)
143A 01   XAE
143B C408 LDI 08
143D CA0E ST 0E (2)
     $ 8
143F C420 LDI 20
1441 CA80 ST 80 (2)
1443 C100 LD 00 (1)
1445 CA0D ST 0D (2)
1447 3F   XPPC 3
1448 C20C LD 0C (2)
144A 01   XAE
144B AA12 ILD 12 (2)
144D 01   XAE
144E CA30 ST 30 (2)
1450 C20E LD 0E (2)
1452 01   XAE
1453 AA12 ILD 12 (2)
1455 01   XAE
1456 CA90 ST 90 (2)
1458 AA12 ILD 12 (2)
145A 01   XAE
145B 31   XPAL 1
145C E201 XOR 01 (2)
145E 9C0D JNZ $ 9
1460 E201 XOR 01 (2)
1462 31   XPAL 1
1463 35   XPAH 1
1464 E202 XOR 02 (2)
1466 981B JZ $ 11
1468 E202 XOR 02 (2)
146A 35   XPAH 1
146B 9803 JMP $ 10
     $ 9
146D E201 XOR 01 (2)
146F 31   XPAL 1
     $ 10
1470 C501 LD @01 (1)
1472 BA0E DLD 0E (2)
1474 9CC9 JNZ $ 8
1476 C4FF LDI FF
1478 CA80 ST 80 (2)
147A C400 LDI 00
147C 37   XPAH 3
147D C455 LDI 55
147F 33   XPAL 3
1480 3F   XPPC 3
1481 9806 JMP $ 5
     $ 11
1483 C4FF LDI FF
1485 CA80 ST 80 (2)
1487 C400 LDI 00
1489 37   XPAH 3
148A C455 LDI 55
148C 33   XPAL 3
148D 3F   XPPC 3
148E 00   HALT
     $ 12
148F C414 LDI 14
1491 37   XPAH 3
1492 CA11 ST 11 (2)
1494 C4B8 LDI B8
```

```
1496  33    XPAL 3
1497  CA10  ST 10 (2)
1499  C20D  LD 0D (2)
149B  D4F0  ANI F0
149D  1E    RR
149E  1E    RR
149F  1E    RR
14A0  1E    RR
14A1  01    XAE
14A2  C380  LD 80 (3)
14A4  CA0C  ST 0C (2)
14A6  C20D  LD 0D (2)
14A8  D40F  ANI 0F
14AA  01    XAE
14AB  C380  LD 80 (3)
14AD  CA0B  ST 0B (2)
14AF  C211  LD 11 (2)
14B1  37    XPAH 3
14B2  C210  LD 10 (2)
14B4  33    XPAL 3
14B5  3F    XPPC 3
14B6  90D7  JMP $ 12
      TAB:
14B8  30 31 32 33 34 35 36 37
14C0  38 39 01 02 03 04 05 06
```

```
18E7 C401 LDI 01
18E9 E8F2 DAD F2 =18DC
18EB C8F0 ST F0 =18DC
18ED C400 LDI 00
18EF E8ED DAD ED =18DD
18F1 C8EB ST EB =18DD
18F3 06   CSA
18F4 9402 JP $ 7
18F6 90E6 JMP $ 4
     $ 7
18F8 C4A6 LDI A6
18FA 8F00 DLY 00
18FC C108 LD 08 (1)
18FE 94E7 JP $ 6
1900 C40A LDI 0A
1902 CA1D ST 1D (2)
1904 3F   XPPC 3
1905 C201 LD 01 (2)
1907 D40F ANI 0F
1909 E209 XOR 09 (2)
190B 9CD1 JNZ $ 4
     $ 8
190D C0CE LD CE =18DC
190F CA01 ST 01 (2)
1911 C0CB LD CB =18DC
1913 CA02 ST 02 (2)
1915 C4A8 LDI A8
1917 CA1D ST 1D (2)
1919 3F   XPPC 3
191A C400 LDI 00
191C C9FF ST FF (1)
191E C900 ST 00 (1)
1920 C408 LDI 08
1922 C8B6 ST B6 =18D9
     $ 9
1924 8FFF DLY FF
1926 B8B2 DLD B2 =18D9
1928 9CFA JNZ $ 9
192A C40A LDI 0A
192C CA1D ST 1D (2)
192E 03   SCL
192F C0AA LD AA =18DA
1931 F8AA CAD AA =18DC
1933 C0A7 LD A7 =18DB
1935 F8A7 CAD A7 =18DD
1937 9403 JP $ 11
1939 3F   XPPC 3
     $ 10
193A 98A2 JMP $ 4
     $ 11
193C C09F LD 9F =18DC
193E C09B ST 9B =18DA
1940 C09C LD 9C =18DD
1942 C098 ST 98 =18DB
1944 C46D LDI 6D
1946 C906 ST 06 (1)
1948 C473 LDI 73
194A C905 ST 05 (1)
194C C430 LDI 30
194E C904 ST 04 (1)
1950 C478 LDI 78
1952 C903 ST 03 (1)
1954 C45B LDI 5B
1956 C902 ST 02 (1)
1958 C479 LDI 79
195A C901 ST 01 (1)
195C C406 LDI 06
195E C820 ST 20 =197F
     $ 12
1960 C4B0 LDI B0
1962 C9FF ST FF (1)
1964 8FFF DLY FF
1966 C400 LDI 00
1968 C9FF ST FF (1)
196A 8FFF DLY FF
196C C108 LD 08 (1)
196E 9403 JP $ 13
1970 3F   XPPC 3
1971 90C7 JMP $ 10
     $ 13
1973 B80B DLD 0B =197F
1975 9CE9 JNZ $ 12
1977 C400 LDI 00
1979 C905 ST 05 (1)
197B C906 ST 06 (1)
197D 908E JMP $ 8
197F 55   BYTE
```

Hasenjagd

```
1980 C440 LDI 40
1982 CDFF ST @FF (1)
1984 C907 ST 07 (1)
1986 C476 LDI 76
1988 C906 ST 06 (1)
198A C477 LDI 77
198C C905 ST 05 (1)
198E C907 ST 07 (1)
1990 C438 LDI 38
1992 C904 ST 04 (1)
1994 C902 ST 02 (1)
1996 C430 LDI 30
1998 C901 ST 01 (1)
199A C400 LDI 00
199C CA13 ST 13 (2)
199E CA00 ST 00 (2)
     $ 1
19A0 AA13 ILD 13 (2)
19A2 C44R LDI 48
19A4 33   XPAL 3
19A5 C40D LDI 0D
19A7 37   XPAH 3
19A8 C407 LDI 07
19AA CF01 ST@01 (3)
19AC C406 LDI 06
19AE CF01 ST @01 (5)
19B0 CA0F ST OF (2)
     $ 2
19B2 AA1. ILD 1. (2)
19B4 C106 LD 06 (1)
19B6 94FA JP $ 2
19B8 C211 LD 11 (2)
19BA 01   XAE
     $ 3
19BB 40   LDE
19BC 1F   RRR
19BD 01   XAE
19BE D403 ANI 03
19C0 980C JZ $ 4
19C2 E401 XRI 01
19C4 980C JZ $ 5
19C6 E401 XRI 01
19C8 E402 XRI 02
19CA 980A JZ $ 6
19CC 90ED JMP $ 3
     $ 4
19CE C401 LDI 01
19D0 9006 JMP $ 7
     $ 5
19D2 C402 LDI 02
19D4 9002 JMP $ 7
     $ 6
19D6 C4FF LDI FF
     $ 7
19D8 CA10 ST 10 (2)
19DA 03   SCL
19DB C20F LD 0F (2)
19DD FA10 CAD 10 (2)
19DF CA0F ST 0F (2)
19E1 CF01 ST @01 (3)
19E3 E4FF XRI FF
19E5 9808 JZ $ 8
19E7 E4FF XRI FF
19E9 E4FE XRI FE
19EB 9802 JZ $ 8
19ED 90CC JMP $ 3
     $ 8
19EF C44A LDI 4A
19F1 33   XPAL 3
19F2 C40D LDI 0D
19F4 37   XPAH 3
19F5 C400 LDI 00
19F7 CA14 ST 14 (2)
     $ 9
19F9 C408 LDI 08
19FB CA12 ST 12 (2)
     $ 10
19FD C400 LDI 00
19FF CD01 ST @01 (1)
1A01 BA12 DLD 12 (2)
1A03 9CF8 JNZ $ 10
1A05 C5F8 LD @F8 (1)
1A07 C701 LD @01 (3)
1A09 01   XAE
1A0A 40   LDE
1A0B E4FF XRI FF
1A0D 9805 JZ $ 11
1A0F 40   LDE
1A10 E4FE XRI FE
1A12 9C08 JNZ $ 12
     $ 11
1A14 C213 LD 13 (2)
1A16 E409 XRI 09
1A18 9C06 JNZ $ 1
1A1A 907D JMP $ 22
     $ 12
1A1C C45C LDI 5C
1A1E C980 ST 80 (1)
1A20 C408 LDI 08
```

```
1A22 CA12 ST 12 (2)
     $ 13
1A24 0F40 DLY 40
1A26 BA12 DLD 12 (2)
1A28 9CFA JNZ $ 13
1A2A C214 LD 14 (2)
1A2C 9CCB JNZ $ 9
1A2E C211 LD 11 (2)
1A30 D407 ANI 07
1A32 CA11 ST 11 (2)
1A34 60   XRE
1A35 9810 JZ $ 14
1A37 02   CCL
1A38 C401 LDI 01
1A2A F20B ADD 0B (2)
1A3C 60   XRE
1A3D 9808 JZ $ 14
1A3F 03   SCL
1A40 C211 LD 11 (2)
1A42 F001 CRI 01
1A44 60   YRE

1A45 9CB2 JNZ $ 9
     $ 14
1A47 C4FF LDI FF
1A49 CA14 ST 14 (2)
1A4B C108 LD 08 (1)
1A4D 9402 JP $ 15
1A4F 90A8 JMP $ 9
     $ 15
1A51 C47E LDI 7E
1A53 C980 ST 80 (1)
1A55 C430 LDI 30
1A57 CA12 ST 12 (2)
     $ 16
1A59 8F20 DLY 20
1A5B C108 LD 08 (1)
1A5D 9402 JP $ 17
1A5F 9006 JMP $ 18
     $ 17
1A61 BA12 DLD 12 (2)
1A63 9894 JZ $ 9
1A65 90F2 JMP $ 16
     $ 18
1A67 E4FF XRI FF
1A69 D407 ANI 07
1A6B E3FF XOR FF (3)
1A6D 981A JZ $ 21
     $ 19
1A6F C400 LDI 00
1A71 C980 ST 80 (1)
1A73 03   SCL
1A74 C300 LD 00 (3)
1A76 FC01 CAI 01
1A78 CB00 ST 00 (3)
1A7A E4FF XRI FF
     $ 20
1A7C 9896 JZ $ 11
1A7E E4FF XRI FF
1A80 01   XAE
1A81 C45C LDI 5C
1A83 C980 ST 80 (1)
1A85 8F90 DLY 90

1A87 90E6 JMP $ 19
     $ 21
1A89 C463 LDI 63
1A8B C980 ST 80 (1)
1A8D 8FFF DLY FF
1A8F C408 LDI 08
1A91 C980 ST 80 (1)
1A93 AA00 ILD 00 (2)
1A95 C400 LDI 00
1A97 90E3 JMP $ 20
     $ 22
1A99 C4A0 LDI A0
1A9B CA1D ST 1D (2)
1A9D C402 LDI 02
1A9F CA1C ST 1C (2)
1AA1 C400 LDI 00
1AA3 37   XPAH 3
1AA4 C455 LDI 55
1AA6 33   XPAL 3
1AA7 3F   XPPC 3
1AA8 C501 LD @01 (1)
1AAA C400 LDI 00
1AAC C906 ST 06 (1)
1AAE C900 ST 00 (1)
1AB0 C47C LDI 7C
1AB2 C905 ST 05 (1)
1AB4 C47B LDI 7B
1AB6 C901 ST 01 (1)
1AB8 C904 ST 04 (1)
1ABA C478 LDI 78
1ABC C902 ST 02 (1)
1ABE C41C LDI 1C
1AC0 C903 ST 03 (1)
1AC2 00   HALT
1AC3 C419 LDI 19
1AC5 37   XPAH 3
1AC6 C47F LDI 7F
1AC8 33   XPAL 3
1AC9 3F   XPPC 3
```

Master Mind

```
1B00 08   NOP
1B01 C45E LDI 5E
1B03 C980 ST 80 (1)
1B05 C454 LDI 54
1B07 C901 ST 01 (1)
1B09 C404 LDI 04
1B0B C902 ST 02 (1)
1B0D C437 LDI 37
1B0F C903 ST 03 (1)
1B11 C905 ST 05 (1)
1B13 C4C0 LDI C0
1B15 C904 ST 04 (1)
1B17 C400 LDI 00
1B19 C9FF ST FF (1)
1B1B C906 ST 06 (1)
1B1D C827 ST 27 =1B45
1B1F C826 ST 26 =1B46
1B21 C825 ST 25 =1B47
1B23 C824 ST 24 =1B48
1B25 C41B LDI 1B
1B27 36   XPAH 2
     $ 1
1B28 C445 LDI 45
1B2A 32   XPAL 2
1B2B C404 LDI 04
1B2D C86E ST 6E =1B9C
1B2F C108 LD 08 (1)
1B31 01   XAE
1B32 40   LDE
1B33 E490 XRI 90
1B35 983D JZ $ 5
     $ 2
1B37 AA09 ILD 00 (2)
1B39 E496 XRI 96
1B3B 9CEB JNZ $ 1
1B3D CE01 ST @01 (2)
1B3F B35C DLD 5C =1B9C
1B41 9CF4 JNZ $ 2
1B43 98E3 JZ $ 1
1B45 55   BYTE
1B46 55   BYTE
1B47 55   BYTE
1B48 55   BYTE
1B49 55   BYTE
1B4A 55   BYTE
1B4B 55   BYTE
1B4C 55   BYTE
     $ 3
1B4D C404 LDI 04
1B4F C84B ST 4B =1B9B
1B51 31   XPAL 1
1B52 C440 LDI 40
1B54 C902 ST 02 (1)
1B56 C903 ST 03 (1)
1B58 C460 LDI 60
1B5A C9FC ST FC (1)
1B5C C9FD ST FD (1)
1B5E C9FE ST FE (1)
1B60 C9FF ST FF (1)
1B62 C400 LDI 00
1B64 C901 ST 01 (1)
1B66 C900 ST 00 (1)
1B68 C834 ST 34 =1B9D
1B6A C833 ST 33 =1B9E
1B6C C44D LDI 4D
1B6E 32   XPAL 2
     $ 4
1B6F C108 LD 08 (1)
1B71 94FC JP $ 4
1B73 01   XAE
     $ 5
1B74 8F0A DLY 0A
     $ 6
1B76 C108 LD 08 (1)
1B78 E4FF XRI FF
1B7A 94FA JP $ 6
1B7C 8F0A DLY 0A
1B7E 40   LDE
1B7F E490 XRI 90
1B81 980A JZ $ 3
1B83 C817 ST 17 =1B9B
1B85 9818 JZ $ 8
1B87 B817 DLD 17 =1B9B
1B89 40   LDE
```

1. Das NIM-Spiel

Das NIM-Spiel liegt auf den Adressen 1000 - 1867. Es wird nach dem Einlesen der Cassette gestartet mit R 1000 R. Auf dem Display erscheint NIM, der Drucker beginnt automatisch mit dem Druckvorgang. Er druckt den Text: Das NIM-Spiel kennen Sie die Regeln des NIM-Spiels schon? (ja = 1, nein = 0). Bei drücken der Taste 0 druckt er weiter: nein, das macht ja nichts. Hier die Reglen: Vor Ihnen liegen 8 Streichholzhäufchen. Abwechselnd nehmen Sie und ich aus einem der Haufen eine beliebige Anzahl von Hölzchen weg, jedoch mindestens 1. Doch dürfen bei jedem Zug nur von einem der Haufen Streichölzer genommen werden. Gewonnen hat derjenige, der das allerletzte Hölzchen wegnimmt. Zum Würfeln viermal eine Taste drücken!

Jetzt drücken Sie vier beliebige Tasten, und es wird ausgedruckt: der Anfangsbestand ist: und hierunter die acht Haufen mit ihren Inhalten. Außerdem fragt der Computer: Wollen Sie anfangen? (ja = 1, nein = 0). Nach dem Drücken der 1 erscheint auf dem Display: Haufen? und Sie tippen eine Zahl von 1 - 8. Hierauf folgt auf dem Display: Zahl? und Sie geben die Anzahl der von Ihnen wegzunehmenden Hölzchen in einer dreistelligen Zahl, z.B.: 008 ein. Der Computer erwidert seine Züge automatisch bis zum (meistens für Sie bitteren) Ende des Spiels.

Wie Sie gewinnen können:
Man schreibt die Anzahl der Streichhölzer je Haufen als 8-stellige Dualzahl auf, bildet die 8-Spalten-Summen, übergibt dem Microprocessor eine solche Streichhölzerzahl, in der die Spaltensumme gerade ist. Beim Spielbeginn überprüft man, ob der gewürfelte Anfangsbestand zufällig 8 gerade Spaltensummen ergibt. In diesem Falle beantwortet man die Frage : "Wollen Sie anfangen?" mit nein (0) und veranlaßt den Microprocessor zu einem Verlegenheitszug.

Beispiel: Der Anfangsbestand lautet

1	2	3	4	5	6	7	8
184	96	145	102	19	197	222	150

Diese Zahlen werden als Dualzahlen untereinander geschrieben:

Haufen								
1	1	0	1	1	1	0	0	0
2	0	1	1	0	0	0	0	0
3	1	0	0	1	0	0	0	1
4	0	1	1	0	0	1	1	0
5	0	0	0	1	0	0	1	1
6	1	1	0	0	0	1	0	1
7	1	1	0	1	1	1	1	0
8	1	0	0	1	0	1	1	0
	5	4	3	5	2	4	4	3

Da eine ungerade Spaltensumme vorkommt, beginnen wir.
Z.B.: Der Zug:

Haufen	Anzahl
7	111

führt z.B. zu geraden Spaltensummen.

Auf der Adresse 1880 - 197F liegt der vielen schon bekannte Reaktionszeittester.

Sie starten unter R 1880 R. Auf dem Display erscheint "Start". Nach Drücken einer beliebigen Taste erscheint nach dem Verschwinden des Wortes Start nach einiger Zeit eine Hex-Zahl zwischen 0 und F. Diese Zahl können Sie durch Drücken der zu ihr gehörenden Taste löschen. Die hierfür aufgewandte Zeit erscheint auf dem Display in Millisekunden. Liegt die Zeit unter 1000, also unter 1 Sekunde, werden Sie mit der Schrift "Spitze" mit blinkendem Ausrufezeichen belohnt. Nach Drücken einer beliebigen Taste (unter Umständen etwas länger festhalten) beginnt das Spiel von neuem. Die Belohnung "Spitze" gibt es allerdings nur, wenn Sie die vorherige Bestleistung unterschreiten.

Auf 1980 - 1 AC9 finden Sie die Hasenjagd.

Nach dem Start R 1980 R erscheint auf dem Display "HALALI". Sie drücken eine beliebige Taste und der Hase (ein kleines Quadrat) läuft los, um ein einem Display anzuhalten und die Ohren aufzustellen. Dies ist der Zeitpunkt, an dem Sie ihn mit einer Taste zwischen 0 und 7 - jedem der acht Displays ist eine dieser 8 Tasten zugeordnet - ab-

schießen können. Getroffen springt er hoch und bleibt liegen. Das Drücken einer beliebigen Taste startet den nächsten Hasen. Sie haben 9 Chancen. Nach dem 9. Versuch erscheint auf dem Display: Beute x.

Adresse 1 B 00 - 1 B F A Master Mind

Nach dem Start unter R 1 B 00 erscheint auf dem Display "M - Mind". Sie drücken die Taste down und sehen auf dem Display -- .... Der Prozessor hat in der Zwischenzeit durch einen Zufallsgenerator 4 Zahlen, bestehend aus den Zffern 0 - 5 (jede Ziffer kann auch mehrfach erscheinen), erzeugt, die aber nicht sichtbar gemacht werden. Sie schreiben nun durch Tastendruck vier von Ihnen gewählte Zahlen in das Display. Nach Druck der Taste up (mit dem Druck der Taste down können Sie sich korrigieren) erscheinen an den ersten beiden Displaystellen zwei Zahlen von 1 - 4, deren erste die Anzahl der von Ihnen geratenen Ziffern angibt, die sowohl vom Wert als auch von der Stelle her richtig sind. Die zweite Zahl gibt die Anzahl der Ziffern an, die zwar von der Zahl, aber nicht von der Stelle her richtig sind. Durch Raten und Kombinieren können Sie nun in weiteren Durchläufen - starten mit down - die richtige Zahl auf das Display bringen. Wer dies mit den wenigsten Durchläufen schafft, hat gewonnen.

Unter 1 C 00 - 1 D E 5 finden Sie ein Puzzle.

Nach dem Start erscheint auf dem Display - PUZZLE -. Nach Drücken einer beliebigen Taste werden 8 kleine Quadrate abwechselnd oben und unten dargestellt. Durch Betätigung der Tasten 0 - 7 können einzelne oder mehrere Quadrate hoch oder heruntergesetzt werden. Ziel des Spieles ist es, alle Quadrate entweder in die obere oder in die untere Reihe zu bekommen.

Des Rätsels Lösung ist - oben - Taste 4 und 1 - unten - Tasten 7, 3, 6, 3, 3, 2, 1, 6, 7 und 1.

Den Monopoly-Würfel finden Sie bei 1 E 00 - 1 F0C

Nach dem Start sehen Sie eine flimmernde 8. Durch Drücken einer beliebigen Taste würfeln Sie eine Zahl zwischen 1 und 6. Jetzt starten Sie wieder durch Drücken einer beliebigen Taste und es erscheint ein zweiter Würfel. Bei Zahlengleichheit (Pasch) blinken beiden Zahlen.

Das Laufschriftprogramm. Es befindet sich unter 1 F 20 - 1 F 63

Der Speicherbereich, auf den zurückgegriffen wird, beginnt bei 0C00. In diesen Speicherbereich von 0C00 - 0FE0 können Sie beliebig viele Zeichen einschreiben. Am Ende Ihrer Laufschriftreihe schreiben Sie ein FF ein. Hier macht der Prozessor einen Rücksprung an die Adresse 0C00. Jetzt beginnt ein neuer Zyclus. Das Einschreiben der Buchstaben geschieht im Siebensegment-Code lt. untenstehender Tabelle.

Die Geschwindigkeit der Laufschrift beeinflussen Sie unter der Adresse 1F61. Hier finden Sie die Date 20. Die langsamste Geschwindigkeit ist FE und die schnellste FF. Alle anderen Kombinationen sind möglich. Unter der Date 1F62 können Sie unabhängig von der Laufgeschwindigkeit die Zeit des Wiederstartes beeinflussen. Start bei 1F20.

Siebensegment-Code für die Laufschrift

A	77	K	75	V	3E	7	07
B	7C	L	38	W	7E	8	7F
C	58	M	37	X	76	9	6F
D	5E	N	54	Y	6E	0	3F
E	79	O	5C	Z	5A	+	46
F	71	P	73	1	06	-	40
G	3D	Q	67	2	5B	=	48
H	74	R	50	3	4F	'	08
I	0D	S	6C	4	66	,	04
J	1E	T	78	5	6D	!	0E
		U	1C	6	7D	?	53

```
1E66 C406 LDI 06
     $ 6
1E67 06   DB6
1E68 C902 ST 02 (1)
1E6A 8F0F DLY 0F
1E6C C108 LD 08 (1)
1E6E 9409 JP $ 9
1E70 C406 LDI 06
1E72 60   XRE
1E73 9861 JZ $ 18
1E75 00   HALT
1E76 90EC JMP $ 6
     $ 9
1E78 C45B LDI 5B
1E7A C902 ST 02 (1)
1E7C 8F0F DLY 0F
1E7E C108 LD 08 (1)
1E80 9408 JP $ 10
1E82 C45B LDI 5B
1E84 60   XRE
1E85 9854 JZ $ 19
1E87 00   HALT
1E88 9010 JMP $ 11
     $ 10
1E8A C44F LDI 4F
1E8C C902 ST 02 (1)
1E8E 8F0F DLY 0F
1E90 C108 LD 08 (1)
1E92 940A JP $ 13
1E94 C44F LDI 4F
1E96 60   XRE
1E97 9847 JZ $ 20
1E99 00   HALT
     $ 11
1E9A 90C8 JMP $ 6
     $ 12
1E9C 90C8 JMP $ 7
     $ 13
1E9E C466 LDI 66
1EA0 C902 ST 02 (1)
1EA2 8F0F DLY 0F
1EA4 C108 LD 08 (1)
1EA6 9408 JP $ 14
1EA8 C466 LDI 66
1EAA 60   XRE
1EAB 9838 JZ $ 21
1EAD 00   HALT
1EAE 90EA JMP $ 11
     $ 14
1EB0 C46D LDI 6D
1EB2 C902 ST 02 (1)
1EB4 8F0F DLY 0F
1EB6 C108 LD 08 (1)
1EB8 9408 JP $ 15
1EBA C46D LDI 6D
1EBC 60   XRE
1EBD 982B JZ $ 22
1EBF 00   HALT
1EC0 90D8 JMP $ 11
     $ 15
1EC2 C47D LDI 7D
1EC4 C902 ST 02 (1)
1EC6 8F0F DLY 0F
1EC8 C108 LD 08 (1)
1ECA 9408 JP $ 17
1ECC C47D LDI 7D
1ECE 60   XRE
1ECF 981E JZ $ 23
1ED1 00   HALT
     $ 16
1ED2 90C6 JMP $ 11
     $ 17
1ED4 90C6 JMP $ 12
     $ 18
1ED6 C406 LDI 06
1ED8 01   XAE
1ED9 9017 JMP $ 24
     $ 19
1EDB C45B LDI 5B
1EDD 01   XAE
1EDE 9012 JMP $ 24
     $ 20
1EE0 C44F LDI 4F
1EE2 01   XAE
1EE3 900D JMP $ 24
     $ 21
1EE5 C466 LDI 66
1EE7 01   XAE
1EE8 9008 JMP $ 24
     $ 22

1EEA C46D LDI 6D
1EEC 01   XAE
1EED 9003 JMP $ 24
     $ 23
1EEF C47D LDI 7D
1EF1 01   XAE
     $ 24
1EF2 40   CLE
1EF3 C902 ST 02 (1)
1EF5 C903 ST 03 (1)
1EF7 8F6F DLY 6F
1EF9 C408 LDI 08
1EFB C902 ST 02 (1)
1EFD C903 ST 03 (1)
1EFF 8F5F DLY 5F
1F01 C108 LD 08 (1)
1F03 945D JP $ 24
     $ 25
1F05 C108 LD 08 (1)
1F07 9402 JP $ 26
1F09 90FA JMP $ 25
     $ 26
1F0B 90C5 JMP $ 16
```

Laufschrift

```
1F20 C400 LDI 00
1F22 31   XPAL 1
1F23 C40C LDI 0C
1F25 35   XPAH 1
1F26 C408 LDI 08
1F28 32   XPAL 2
1F29 C407 LDI 07
1F2B 36   XPAH 2
     $ 1
1F2C C501 LD @01 (1)
1F2E CEFF ST @FF (2)
1F30 C100 LD 00 (1)
1F32 E4FF XRI FF
1F34 9C0C JNZ $ 3
1F36 C02B LD 2B =1F62
1F38 C82A ST 2A =1F63
     $ 2
1F3A 8F0A DLY 0A
1F3C B826 DLD 26 =1F63
1F3E 9CFA JNZ $ 2
1F40 90DE JMP $ 0
     $ 3
1F42 32   XPAL 2
1F43 9C0E JNZ $ 5
1F45 C01B LD 1B =1F61
1F47 C81B ST 1B =1F63
     $ 4
1F49 8F0A DLY 0A
1F4B B817 DLD 17 =1F63
1F4D 9CFA JNZ $ 4
1F4F C5F9 LD @F9 (1)
1F51 C408 LDI 08
     $ 5
1F53 01   XAE
1F54 C408 LDI 08
1F56 32   XPAL 2
1F57 C200 LD 00 (2)
1F59 01   XAE
1F5A 32   XPAL 2
1F5B 01   XAE
1F5C 94CE JP $ 1
1F5E 3F   XPPC 3
1F5F 9020 JMP $ 6
1F61 55   BYTE
1F62 55   BYTE
1F63 55   BYTE
```

```
    N I M
1000 C4 07 35 C4 09 51 C4 00
1008 C9 00 C9 01 C9 02 C9 46
1010 C9 07 C4 37 C9 03 C4 05
1018 C9 04 C4 54 C9 05 C4 14
1020 36 C4 2D 32 3E 44 41 53
1028 6A 4E 49 4D 6D 53 5B 49
1030 45 4C FF 3E FF 3E 05 05
1038 0E 0E 05 0E 20 15 05 05
1040 10 04 29 05 20 12 05 07
1048 05 0C 0E 20 04 05 15 FF
1050 3E 0E 05 0C 2D 13 10 09
1058 05 0C 13 20 13 05 06 0F
1060 0E 20 3F FF C4 14 35 C4
1068 51 31 3D 40 98 7A 3E 17
1070 09 05 20 09 03 08 20 13
1078 05 08 05 2C 08 01 02 05
1080 20 09 03 08 20 05 13 20
1088 02 05 09 FF 3E 09 08 0E
1090 05 0E 20 0C 09 14 20 05
1098 09 0E 05 0D 20 12 0F 15
10A0 14 09 0E 09 05 12 14 05
10A8 0E FF 3E 13 10 09 05 0C
10B0 05 12 20 1A 15 20 14 15
10B8 0E 2E 04 01 13 20 0D 01
10C0 03 08 14 20 04 01 13 FF
10C8 3E 13 10 09 05 0C 20 0E
10D0 01 14 15 05 12 0C 09 03
10D8 08 20 13 10 01 0E 0E 05
10E0 0E 04 05 12 2E FF 90 6C
10E8 3E 04 01 13 20 0D 01 03
10F0 08 14 20 0A 01 20 0E 09
10F8 03 08 14 13 2C 08 09 05
1100 12 FF 3E 04 09 05 20 12
1108 05 07 05 0C 0E 3A FF 3E
1110 16 0F 12 20 09 08 0E 05
1118 0E 20 0C 09 05 07 05 0E
1120 20 38 20 13 14 12 05 09
1128 03 08 08 0F 0C 1A 2D FF
1130 3E 08 01 05 15 06 03 08
1138 05 0E 2E 01 02 17 05 03
1140 08 13 05 0C 0E 04 20 0E
1148 05 08 0D 05 0E 20 13 09
1150 05 FF 90 02 90 64 3E 15
1158 0E 04 20 09 03 08 20 01
1160 15 13 20 05 09 0E 05 0D
1168 20 04 05 12 20 08 01 15
1170 06 05 0E FF 3E 05 09 0E
1178 05 20 02 05 0C 09 05 02
1180 09 07 05 20 01 0E 1A 01
1188 08 0C 20 16 0F 0E 20 08
1190 0F 05 0C 1A 2D FF 3E 03
1198 08 05 0E 20 17 05 07 2C
11A0 0A 05 04 0F 03 08 20 0D
11A8 09 0E 04 05 13 14 05 0E
11B0 13 20 05 09 0E 13 2E FF
11B8 90 02 90 62 3E 04 0F 03
11C0 08 20 04 01 12 06 20 0D
11C8 01 0E 20 02 05 09 20 0A
11D0 05 04 05 0D 20 1A 15 07
11D8 20 0E 15 12 FF 3E 16 0F
11E0 0E 20 05 09 0E 05 0D 20
11E8 04 05 12 20 09 01 15 06
11F0 05 0E 20 13 14 12 05 09
11F8 03 08 2D FF 3E 08 0F 05
1200 0C 1A 05 12 20 17 05 07
1208 0E 05 08 0D 05 0E 2E 07
1210 05 17 0F 0E 0E 05 0E 20
1218 08 01 14 FF 90 02 90 34
1220 3E 04 05 12 0A 05 0E 09
1228 07 05 2C 04 05 12 20 04
1230 01 13 20 01 0C 0C 05 12
1238 0C 05 14 1A 14 05 FF 3E
1240 08 0F 05 0C 1A 03 08 05
1248 0E 20 17 05 07 0E 09 0D
1250 0D 14 2E FF 3E 1A 15 0D
1258 20 17 15 05 12 06 05 0C
1260 0E 20 34 20 0D 01 0C 20
1268 05 09 0E 05 20 14 01 13
1270 14 05 FF 3E 04 12 15 05
1278 03 08 05 0E 20 21 FF 3E
1280 FF 3E FF C4 07 36 C4 08
1288 32 C4 0C 35 C4 00 31 A9
1290 00 C2 00 94 FA 8F 1E A9
1298 01 C2 00 94 02.90 F8 8F
12A0 1E A9 02 C2 00 94 FA 8F
12A8 1E A9 03 C2 00 94 02 90
12B0 F8 8F 1E A9 04 C2 00 94
12B8 FA 8F 1E A9 05 C2 00 94
12C0 02 90 F8 8F 1E A9 06 C2
12C8 00 94 FA 8F 1E A9 07 C2
12D0 00 94 02 90 F8 8F 1E C4
12D8 14 36 C4 2D 32 3E 04 05
12E0 12 20 01 0E 06 01 0E 07
12E8 12 02 05 13 14 01 0E 04
12F0 20 09 13 14 20 3A FF C4
12F8 08 37 C4 55 33 C4 14 C6
1300 A7 C4 A8 CB A8 3F 3E FF
1308 3E 17 0F 0C 0C 05 0E 20
1310 13 09 05 20 01 0E 06 01
1318 2E 07 05 0E 20 3F FF C4
1320 14 35 C4 51 31 3D 40 98
1328 07 C4 16 37 C4 91 33 3F
1330 C4 0C 35 C4 00 31 C4 0C
1338 36 C4 08 32 C4 07 C9 10
1340 C4 13 37 C4 03 33 3F C1
1348 10 01 C2 00 9C 87 40 98
1350 63 89 19 90 F1 C4 14 36
1358 C4 0C 32 C1 10 01 C2 00
1360 C9 15 C4 0C 36 C4 08 32
1368 C4 07 C9 11 01 C1 00 D1
1370 15 9C 04 B9 11 90 F5 C1
1378 00 C9 13 C4 00 C9 00 C4
1380 07 C9 10 3F C1 10 98 04
1388 B9 10 90 F7 C4 07 C9 10
1390 C4 00 C9 12 C1 10 01 C2
1398 00 D9 12 C9 12 40 98 04
13A0 B9 10 90 F0 C1 11 01 C1
13A8 12 C9 00 03 C1 13 F9 12
13B0 C9 12 90 19 C4 00 C9 10
13B8 01 C1 00 9C 04 A9 10 90
13C0 F7 03 FC 01 C9 00 40 C9
13C8 11 C4 01 C9 12 C4 16 37
13D0 C4 13 33 3F C4 07 C9 14
13D8 C1 10 01 C4 00 CA 00 C4
13E0 14 36 C4 0C 32 C2 00 C9
13E8 15 C4 0C 36 C4 08 32 C1
13F0 14 01 C1 00 D1 15 C9 16
13F8 C1 10 01 C2 00 E1 16 CA
1400 00 C1 14 98 04 B9 14 90
1408 E6 3F 90 C8 01 02 04 08
1410 10 20 40 80 01 02 04 08
1418 16 32 64 28 00 00 00 00
1420 00 00 00 01 30 31 32 33
1428 34 35 36 37 38 39 C4 0F
1430 37 C4 00 33 C6 01 C6 01
1438 CF 01 E4 FF 9C F8 C6 FF
1440 C4 00 37 C4 55 33 C4 F0
1448 CB A7 C4 0E CB A8 3F 3E
1450 90 DC 3E 28 0A 01 3D 31
1458 20 2C 20 0E 05 09 0E 3D
1460 30 29 FF 3E FF 3E FF C4
1468 00 37 C4 55 33 C4 02 CB
1470 A7 C4 0A CB A8 3F C4 0F
1478 37 C4 E0 33 C3 09 01 C4
1480 0C 37 C4 10 33 40 CB 00
1488 C4 14 36 C4 2D 32 40 E4
1490 00 98 0D 40 E4 01 98 02
1498 90 CD 3E 00 01 FF 90 06
14A0 3E 0E 05 09 0E FF 3D 90
14A8 A9 C4 14 36 C4 2D 32 3E
14B0 2D 2D 31 2D 2D 2D 32 2D
14B8 2D 2D 33 2D 2D 2D 34 2D
14C0 2D 2D 35 2D 2D 2D 36 2D
14C8 2D 2D 37 2D 2D 2D 38 FF
14D0 C4 0F 36 C4 00 32 C4 0C
14D8 35 C4 10 31 C4 00 C9 00
14E0 C1 00 31 C4 15 37 C4 13
14E8 33 3F C4 0C 35 C4 10 31
14F0 A9 00 E4 00 98 04 C6 04
14F8 90 E6 C4 00 37 C4 55 33
1500 C4 F0 CB A7 C4 0E CB A8
1508 C4 FF CA 04 3F C4 00 37
1510 C4 14 33 3F C4 00 CA 01
1518 CA 02 C1 00 CA 00 C4 07
1520 CA 03 C4 14 35 C4 0C 31
1528 C2 03 01 C1 00 D2 00 98
1530 13 C4 14 31 C1 00 02 EA
1538 02 CA 02 C4 1C 31 C1 00
1540 EA 01 CA 01 C2 03 98 04
1548 BA 03 90 D9 C4 20 CA 03
1550 C4 24 31 C2 01 98 05 01
1558 C1 00 90 02 C4 20 CA 00
1560 C2 02 D4 F0 1E 1E 1E 1E
1568 98 05 01 C1 00 90 09 01
1570 C2 00 E4 20 9C F5 C4 20
1578 CA 01 C2 02 D4 0F 01 C1
1580 00 CA 02 3F 90 8E C4 14
1588 36 C4 2D 32 3E 05 01 15
1590 06 05 0E 20 20 01 0E 1A
1598 01 08 0C FF C4 0C 35 C4
15A0 11 31 A9 00 C4 0F 36 C4
15A8 00 32 C4 15 37 C4 13 33
15B0 3F C6 04 C4 0C 35 C4 12
15B8 31 C4 20 CE 01 CE 01 CE
15C0 01 CE 01 3F C6 04 C4 FF
15C8 CA 00 C4 00 37 C4 55 33
15D0 C4 F0 CB A7 C4 0E CB A8
15D8 3F C4 14 37 C4 A8 33 3F
15E0 C4 0C 35 C4 00 31 C4 00
15E8 C9 10 C9 11 C9 12 C1 10
15F0 01 02 C1 12 F1 98 C9 12
15F8 C4 00 F1 .. C9 11 B9 10
1600 E4 F5 9C 2A C1 11 9C 07
1608 C1 12 9C 02 C9 10 3F C4
1610 C9 12 3F C4 14 36 C4
1618 2D 32 3E 00 05 09 0E 20
1620 1A 15 07 20 2A FF C4 00
1628 37 C4 55 33 C4 15 CB A7
```

```
1630 C4 85 CB A6 3F C4 15 37
1638 C4 0F CC 3F C1 10 98 07
1640 C4 15 37 C4 91 33 3F 3E
1648 1F FF FF 20 08 01 02 05
1650 0E 20 0C 05 09 04 05 12
1658 20 16 05 12 0C 0F 12 05
1660 05 20 21 FF 3E FF 3E FF
1668 3E 0E 0F 03 08 20 05 09
1670 0E 20 13 10 09 05 0C 20
1678 3F FF C4 14 35 C4 51 31
1680 3D 40 98 07 C4 1F 37 C4
1688 FF 33 3F C4 00 37 C4 00
1690 33 3F 3E 09 00 12 20 1A
1698 15 07 20 3A FF C4 07 35
16A0 C4 00 31 C4 74 C9 07 C4
16A8 77 C9 06 C4 1C C9 05 C4
16B0 71 C9 04 C4 79 C9 03 C4
16B8 54 C9 02 C4 00 C9 01 C4
16C0 DB C9 00 C4 02 37 C4 10
16C8 33 3F 40 98 F6 E4 08 98
16D0 05 40 D4 08 9C ED C4 0B
16D8 36 C4 FF 32 C2 00 98 E3
16E0 40 CA 12 C4 07 35 C4 00
16E8 31 C4 5B C9 07 C4 77 C9
16F0 06 C4 74 C9 05 C4 38 C9
16F8 04 C4 00 C9 03 C9 01 C9
1700 00 C4 08 C9 02 C4 00 37
1708 C4 55 33 C4 02 CB A7 C4
1710 0A CB A8 3F C4 0F 36 C4
1718 E0 32 C2 07 C9 02 C2 09
1720 01 40 98 09 E4 01 98 05
1728 40 E4 02 9C B6 C4 08 C9
1730 01 3F C2 07 C9 01 40 E4
1738 02 98 14 C2 09 E4 09 98
1740 1E C2 09 E4 08 98 18 C2
1748 09 D4 08 9C E0 98 10 C2
1750 09 D4 08 9C D8 C2 09 E4
1758 07 98 D2 E4 01 98 CC C4
1760 0C 35 C4 00 31 C4 00 C9
1768 12 40 C9 10 40 98 0E 02
1770 C1 12 F4 64 C9 12 02 40
1778 F4 FF 01 90 EF C2 09 01
1780 40 98 0E 02 C1 12 F4 0A
1788 C9 12 02 40 F4 FF 01 90
1790 EF C1 10 E4 02 9C 0C C2
1798 09 E4 05 9C 06 C4 01 C9
17A0 10 90 04 C4 00 C9 10 C4
17A8 07 35 C4 08 C9 00 3F C2
17B0 07 C9 00 C4 0C 35 C1 10
17B8 98 12 C2 09 D4 08 9C E7
17C0 C2 09 E4 07 98 E1 E4 01
17C8 98 DD 90 12 C2 09 E4 09
17D0 98 0C C2 09 E4 08 98 06
17D8 C2 09 D4 08 9C C9 C2 09
17E0 01 40 98 0E 02 C1 12 F4
17E8 01 C9 12 02 40 F4 FF 01
17F0 90 EF 02 C1 11 F4 FF 01
17F8 EF C1 00 F9 12 C9 10 06
1800 D4 80 98 06 C1 10 C9 80
1808 90 07 C4 16 37 C4 9C 33
1810 3F 02 C1 11 F4 FF C9 11
1818 C4 00 37 C4 55 33 C4 15
1820 CB A7 C4 85 CB A8 3F C4
1828 15 37 C4 DF 33 3F C1 10
1830 98 07 C4 13 37 C4 2F 33
1838 3F C4 14 36 C4 2D 32 3E
1840 07 12 01 14 15 0C 09 05
1848 12 05 2C 20 13 09 05 20
1850 08 01 02 05 0E 20 07 05
1858 17 0F 0E 0E 05 0E 20 21
1860 FF C4 16 37 C4 63 33 3F
```

Reaktionszeittester

TAB:

```
1880 C4 00 C8 57 C4 10 C8 54
1888 C4 6D C9 04 C4 78 C9 00
1890 C9 03 C4 5F C9 02 C4 50
1898 C9 01 C4 00 C9 05 C9 06
18A0 C9 FF A8 36 C1 08 94 F8
18A8 8F FF C4 00 C9 00 C9 01
18B0 C9 02 C9 03 C9 04 C0 22
18B8 CA 01 C4 A0 CA 1D 8F 20
18C0 B8 18 9C FA C4 00 37 C4
18C8 55 33 3F C4 00 C9 FF C9
18D0 00 C9 02 C9 03 C9 04 90
18D8 07 00 00 10 01 00 90 A8
18E0 C4 00 C8 F9 C8 F8 02 C4
18E8 01 E8 F2 C8 F0 C4 00 E8
18F0 ED C8 EB 06 94 02 90 E6
18F8 C4 A6 8F 00 C1 08 94 E7
1900 C4 0A CA 1D 3F C2 01 D4
1908 0F E2 09 9C D1 C0 CE CA
1910 01 C0 CB CA 02 C4 A0 CA
1918 1D 3F C4 00 C9 FF C9 00
1920 C4 08 C8 B6 8F FF B8 B2
1928 9C FA C4 0A CA 1D 03 C0
1930 AA F8 AA C0 A7 F8 A7 94
1938 03 3F 90 A2 C0 9F C8 9B
1940 C0 9C C8 98 C4 6D C9 06
1948 C4 73 C9 05 C4 30 C9 04
1950 C4 78 C9 03 C4 5B C9 02
1958 C4 79 C9 01 C4 08 C8 20
1960 C4 B0 C9 FF 8F FF C4 00
1968 C9 FF 8F FF C1 08 94 03
1970 3F 90 C7 B8 0B 9C E9 C4
1978 00 C9 05 C9 06 90 8E 07
```

Hasenjagd

TAB:

```
1980 C4 40 CD FF C9 07 C4 76
1988 C9 06 C4 77 C9 05 C9 03
1990 C4 38 C9 04 D9 02 C4 30
1998 C9 01 C4 00 CA 13 CA 00
19A0 AA 13 C4 4A 33 C4 0D 37
19A8 C4 07 CF 01 C4 06 CF 01
19B0 CA 0F AA 11 C1 08 94 FA
19B8 C2 11 01 40 1F 01 D4 03
19C0 98 0C E4 01 98 0C E4 01
19C8 E4 02 98 0A 90 ED C4 01
19D0 90 06 C4 02 90 02 C4 FF
19D8 CA 10 03 C2 0F FA 10 CA
19E0 0F CF 01 E4 FF 98 08 E4
19E8 FF E4 FE 98 02 90 CC C4
19F0 4A 33 C4 0D 37 C4 00 CA
19F8 14 C4 08 CA 12 C4 00 CD
1A00 01 BA 12 9C F8 C5 F8 C7
1A08 01 01 40 E4 FF 98 05 40
1A10 E4 FE 9C 08 C2 13 E4 09
1A18 9C 86 90 7D C4 5C C9 80
1A20 C4 08 CA 12 8F 40 BA 12
1A28 9C FA C2 14 9C CB C2 11
1A30 D4 07 CA 11 60 98 10 02
1A38 C4 01 F2 0B 60 98 08 03
1A40 C2 11 FC 01 60 9C B2 C4
1A48 FF CA 14 C1 08 94 02 90
1A50 A8 C4 7E C9 00 C4 30 CA
1A58 12 8F 20 C1 08 94 02 90
1A60 06 BA 12 98 94 90 F2 E4
1A68 FF D4 07 E3 FF 98 1A C4
1A70 00 C9 00 03 C3 00 FC 01
1A78 CB 00 E4 FF 98 96 E4 FF
1A80 01 C4 5C C9 00 8F 90 90
1A88 E6 C4 63 C9 00 8F FF C4
1A90 08 C9 00 AA 00 C4 00 90
1A98 E3 C4 A0 CA 1D C4 02 CA
1AA0 1C C4 00 37 C4 55 33 3F
1AA8 C5 01 C4 00 C9 06 C9 00
1AB0 C4 7C C9 05 C4 7B C9 01
1AB8 C9 04 C4 7B C9 02 C4 1C
1AC0 C9 03 00 C4 19 37 C4 7F
1AC8 33 3F
```

Master Mind

TAB:

```
1B00 00 C4 5E C9 00 C4 54 C9
1B08 01 C4 04 C9 02 C4 37 C9
1B10 03 C9 05 C4 C0 C9 04 C4
1B18 00 C9 FF C9 06 C8 27 C8
1B20 26 C8 25 C8 24 C4 1B 36
1B28 C4 45 32 C4 04 C8 6E C1
1B30 08 01 40 E4 90 98 3D AA
1B38 00 E4 06 9C EB CE 01 B8
1B40 5C 9C F4 98 E3 55 55 55
1B48 55 55 55 55 C4 04 C8
1B50 4B 31 C4 40 C9 02 C9 03
1B58 C4 00 C9 FC C9 FD C9 FE
1B60 C9 FF C4 00 C9 01 C9 00
1B68 C8 34 C8 33 C4 4D 32 C1
1B70 08 94 FC 01 8F 0A C1 08
1B78 E4 FF 94 FA 8F 0A 40 E4
1B80 90 98 CA C0 17 98 18 B8
1B88 13 40 D4 0F CE FF 01 C4
1B90 01 37 C4 1F 33 C3 80 CD
1B98 FF 90 D4 55 55 55 55 40
1BA0 E4 80 9C CB C4 07 31 C4
1BA8 05 C8 F1 B8 EF 98 18 C6
1BB0 FF D4 7F CA 00 E2 04 9C
1BB8 F2 C4 80 DA 00 CA 00 C4
1BC0 F7 CA 04 A6 D9 90 E4 C0
1BC8 D5 01 C3 80 C9 00 C4 05
1BD0 C8 CB C4 1B 37 C4 49 33
1BD8 C4 05 C8 C0 B8 BF 98 14
1BE0 C6 01 B8 B8 98 EF C7 01
1BE8 E2 FF 9C F6 C4 F6 CB FF
1BF0 A8 AD 90 E1 C8 A6 C0 A7
1BF8 90 94
```

Puzzle

```
      TAB:
1C00 C4 40 C9 06 C9 FF C4 73
1C08 C9 05 C4 3E C9 04 C4 5B
1C10 C9 03 C9 62 C4 32 C9 01
1C18 C4 79 C9 00 C4 00 C9 06
1C20 C9 FF 8F 8F C4 46 C9 06
1C28 C9 FF 8F 5F C1 08 94 EC
1C30 C1 08 94 02 90 FA 9F FF
1C38 C4 00 01 C4 63 C9 06 C4
1C40 5C C9 05 C4 63 C9 04 C4
1C48 5C C9 03 C4 63 C9 02 C4
1C50 5C C9 01 C4 63 C9 00 C4
1C58 5C C9 FF C1 08 E4 F0 9C
1C60 0F C4 63 C9 05 01 C1 08
1C68 94 02 90 FA 8F FF 90 56
1C70 C1 08 E4 F1 9C 0F C4 63
1C78 C9 FF 01 5C 08 94 02 90
1C80 FA 8F FF 90 41 C1 08 E4
1C88 F2 9C 0F C4 5C C9 00 01
1C90 C1 08 94 02 90 FA 9F FF
1C98 90 2C C1 08 E4 F3 9C 0F
1CA0 C4 5C C9 04 01 C1 08 94
1CA8 02 90 FA 8F FF 90 5A C1
1CB0 08 E4 F4 9C 13 C4 63 C9
1CB8 03 01 C1 08 94 02 90 FA
1CC0 8F FF 90 02 90 95 90 41
1CC8 C1 08 E4 F5 9C 0F C4 63
1CD0 C9 01 01 C1 08 94 02 90
1CD8 FA 8F FF 90 2C C1 08 E4
1CE0 F6 9C 0F C4 5C C9 02 01
1CE8 C1 08 94 02 90 FA 8F FF
1CF0 90 17 C1 08 E4 F7 9C 0F
1CF8 C4 5C C9 06 01 C1 08 94
1D00 02 90 FA 8F FF 90 02 90
1D08 BA C1 08 E4 F0 9C 13 C4
1D10 63 60 98 74 C4 5C C9 01
1D18 C1 08 94 02 90 FA 8F FF
1D20 90 E5 C1 08 E4 F1 9C 13
1D28 C4 63 60 98 5B C4 5C C9
1D30 00 C1 08 94 02 90 FA 8F
1D38 FF 90 CC C1 08 E4 F2 9C
1D40 13 C4 63 60 98 42 C4 63
1D48 C9 06 C1 08 94 02 90 FA
1D50 8F FF 90 B3 C1 08 E4 F3
1D58 9C 15 C4 63 60 98 29 C4
1D60 5C C9 05 C1 08 94 02 90
1D68 FA 8F FF 90 9A 90 9A C1
1D70 08 E4 F4 9C 17 C4 5C 60
1D78 98 0E C4 63 C9 03 C1 08
1D80 94 02 90 FA 8F FF 90 02
1D88 90 4F 90 DF C1 08 E4 F5
1D90 9C 13 C4 5C 60 98 42 C4
1D98 5C C9 02 C1 08 94 02 90
1DA0 FA 8F FF 90 E5 C1 08 E4
1DA8 F6 9C 13 C4 5C 60 98 29
1DB0 C4 5C C9 FF C1 08 94 02
1DB8 90 FA 8F FF 90 CC C1 08
1DC0 E4 F7 9C 13 C4 5C 60 98
1DC8 10 C4 63 C9 04 C1 08 94
1DD0 02 90 FA 8F FF 90 B3 90
1DD8 94 C4 5C C9 FF C4 63 C9
1DE0 01 40 C9 05 90 EF
```

Monopoly Würfel

```
      TAB:
1E00 C4 00 C9 06 C9 05 C9 04
1E08 C9 02 C9 01 C9 00 C9 FF
1E10 C4 06 39 05 01 8F 0F C1
1E18 08 94 03 00 50 48 C4 5B
1E20 C9 03 01 8F 0F C1 08 94
1E28 03 00 90 3B C4 4F C9 03
1E30 01 8F 0F C1 08 94 03 00
1E38 90 2C C4 66 C9 03 01 8F
1E40 0F C1 08 94 03 00 90 1E
1E48 C4 6D C9 03 01 8F 0F C1
1E50 08 94 03 00 90 10 C4 7D
1E58 C9 03 01 8F 0F C1 08 94
1E60 03 00 90 02 90 9A C4 06
1E68 C9 02 8F 0F C1 08 94 08
1E70 C4 06 60 98 61 00 90 EC
1E78 C4 5B C9 02 8F 0F C1 08
1E80 94 08 C4 5B 60 98 54 00
1E88 90 10 C4 4F C9 02 8F 0F
1E90 C1 08 94 0A C4 4F 60 98
1E98 47 00 90 C8 90 C8 C4 66
1EA0 C9 02 8F 0F C1 08 94 08
1EA8 C4 66 60 98 38 00 90 EA
1EB0 C4 6D C9 02 8F 0F C1 08
1EB8 94 08 C4 6D 60 98 2B 00
1EC0 90 D8 C4 7D C9 02 8F 0F
1EC8 C1 08 94 08 C4 7D 60 98
1ED0 1E 00 90 C6 90 C6 C4 06
1ED8 01 90 17 C4 5B 01 90 12
1EE0 C4 4F 01 90 0D C4 66 01
1EE8 90 08 C4 6D 01 90 03 C4
1EF0 7D 01 40 C9 02 C9 03 8F
1EF8 8F C4 00 C9 02 C9 03 8F
1F00 5F C1 08 94 ED C1 08 94
1F08 02 90 FA 90 C5
```

Laufschrift

```
      TAB:
1F20 C4 00 31 C4 0C 35 C4 08
1F28 32 C4 07 36 C5 01 CE FF
1F30 C1 00 E4 FF 9C 0C C8 2B
1F38 C8 2A 8F 0A B8 26 9C FA
1F40 90 DE 32 9C 0E C0 1B C8
1F48 1B 8F 0A B8 17 9C FA C5
1F50 F9 C4 08 01 C4 08 32 C2
1F58 00 01 32 01 94 CE 3F 90
1F60 20 20 20 00
```

Reaktionszeittester

```
1880  C400  LDI 00
1882  C857  ST 57 =18DA
1884  C410  LDI 10
1886  C854  ST 54 =18DB
      $  1
1888  C46D  LDI 6D
188A  C904  ST 04 (1)
188C  C478  LDI 78
188E  C900  ST 00 (1)
1890  C903  ST 03 (1)
1892  C45F  LDI 5F
1894  C902  ST 02 (1)
1896  C450  LDI 50
1898  C901  ST 01 (1)
189A  C400  LDI 00
189C  C905  ST 05 (1)
189E  C906  ST 06 (1)
18A0  C9FF  ST FF (1)
      $  2
18A2  A836  ILD 36 =18D9
18A4  C108  LD 08 (1)
18A6  94FA  JP $  2
18A8  8FFF  DLY FF
18AA  C400  LDI 00
18AC  C900  ST 00 (1)
18AE  C901  ST 01 (1)
18B0  C902  ST 02 (1)
18B2  C903  ST 03 (1)
18B4  C904  ST 04 (1)
18B6  C822  LD 22 =18D9
18B8  CA01  ST 01 (2)
18BA  C4A0  LDI A0
18BC  CA1D  ST 1D (2)
      $  7
18BE  8F20  DLY 20
18C0  B818  DLD 18 =18D9
18C2  9CFA  JNZ $  7
18C4  C400  LDI 00
18C6  37    XPAH 3
18C7  C455  LDI 55
18C9  33    XPAL 3
18CA  3F    XPPC 3
18CB  C400  LDI 00
18CD  C9FF  ST FF (1)
18CF  C900  ST 00 (1)
18D1  C902  ST 02 (1)
18D3  C903  ST 03 (1)
18D5  C904  ST 04 (1)
18D7  9007  JMP $  5
18D9  55    BYTE
18DA  55    BYTE
18DB  55    BYTE
18DC  55    BYTE
18DD  55    BYTE
      $  4
18DE  9008  JMP $  1
      $  5
18E0  C400  LDI 00
18E2  C8F9  ST F9 =18DC
18E4  C8F8  ST F8 =18DD
18E6  02    CCL
      $  6
```

```
1B8A  340F  ANI 0F
1B8C  CEFF  ST @FF (2)
      $ 7
1B8E  01    XAE
1B8F  C401  LDI 01
1B91  37    XPAH 3
1B92  C41F  LDI 1F
1B94  33    XPAL 3
1B95  C380  LD 80 (3)
1B97  CDFF  ST @FF (1)
1B99  90D4  JMP $ 4
1B9B  55    BYTE
1B9C  55    BYTE
1B9D  55    BYTE
1B9E  55    BYTE
      $ 8
1B9F  40    LDE
1BA0  E480  XRI 80
1BA2  9CCB  JNZ $ 4
1BA4  C407  LDI 07
1BA6  31    XPAL 1
1BA7  C405  LDI 05
1BA9  C8F1  ST F1 =1B9B
      $ 9
1BAB  B8EF  DLD EF =1B9B
1BAD  9818  JZ $ 10
1BAF  C6FF  LD @FF (2)
1BB1  D47F  ANI 7F
1BB3  CA00  ST 00 (2)
1BB5  E204  XOR 04 (2)
1BB7  9CF2  JNZ $ 9
1BB9  C480  LDI 80
1BBB  DA00  OR 00 (2)
1BBD  CA00  ST 00 (2)
1BBF  C4F7  LDI F7
1BC1  CA04  ST 04 (2)
1BC3  A8D9  ILD D9 =1B9D
1BC5  90E4  JMP $ 9
      $ 10
1BC7  C0D5  LD D5 =1B9D
1BC9  01    XAE
1BCA  C380  LD 80 (3)
1BCC  C900  ST 00 (1)
1BCE  C405  LDI 05
1BD0  C8CB  ST CB =1B9C
1BD2  C41B  LDI 1B
1BD4  37    XPAH 3
      $ 11
1BD5  C449  LDI 49
1BD7  33    XPAL 3
1BD8  C405  LDI 05
1BDA  C8C0  ST C0 =1B9B
1BDC  B8BF  DLD BF =1B9C
1BDE  9814  JZ $ 13
1BE0  C601  LD @01 (2)
      $ 12
1BE2  B8B8  DLD B8 =1B9B
1BE4  98EF  JZ $ 11
1BE6  C701  LD @01 (3)
1BE8  E2FF  XOR FF (2)
1BEA  9CF6  JNZ $ 12
1BEC  C4F6  LDI F6
1BEE  C8FF  ST FF (3)
1BF0  A8AD  ILD AD =1B9E
1BF2  90E1  JMP $ 11
      $ 13
1BF4  C8A6  ST A6 =1B9B
1BF6  C0A7  LD A7 =1B9E
1BF8  9094  JMP $ 7
```

Puzzle

```
1C00  C440  LDI 40
1C02  C906  ST 06 (1)
1C04  C9FF  ST FF (1)
1C06  C473  LDI 73
1C08  C905  ST 05 (1)
1C0A  C43E  LDI 3E
1C0C  C904  ST 04 (1)
1C0E  C45B  LDI 5B
1C10  C903  ST 03 (1)
1C12  C902  ST 02 (1)
1C14  C438  LDI 38
1C16  C901  ST 01 (1)
1C18  C479  LDI 79
1C1A  C900  ST 00 (1)
      $ 1
1C1C  C400  LDI 00
1C1E  C906  ST 06 (1)
1C20  C9FF  ST FF (1)
1C22  8F9F  DLY 9F
1C24  C440  LDI 40
1C26  C906  ST 06 (1)
1C28  C9FF  ST FF (1)
1C2A  8F5F  DLY 5F
1C2C  C108  LD 08 (1)
1C2E  94EC  JP $ 1
      $ 2
1C30  C108  LD 08 (1)
1C32  9402  JP $ 3
1C34  90FA  JMP $ 2
      $ 3
1C36  8FFF  DLY FF
1C38  C400  LDI 00
1C3A  01    XAE
1C3B  C463  LDI 63
1C3D  C906  ST 06 (1)
1C3F  C45C  LDI 5C
1C41  C905  ST 05 (1)
1C43  C463  LDI 63
1C45  C904  ST 04 (1)
1C47  C45C  LDI 5C
1C49  C903  ST 03 (1)
1C4B  C463  LDI 63
1C4D  C902  ST 02 (1)
1C4F  C45C  LDI 5C
1C51  C901  ST 01 (1)
1C53  C463  LDI 63
1C55  C900  ST 00 (1)
1C57  C45C  LDI 5C
1C59  C9FF  ST FF (1)
      $ 4
1C5B  C108  LD 08 (1)
1C5D  E4F0  XRI F0
1C5F  9C0F  JNZ $ 7
1C61  C463  LDI 63
1C63  C905  ST 05 (1)
1C65  01    XAE
      $ 5
1C66  C108  LD 08 (1)
1C68  9402  JP $ 6
1C6A  90FA  JMP $ 5
      $ 6
1C6C  8FFF  DLY FF
1C6E  9056  JMP $ 20
      $ 7
1C70  C108  LD 08 (1)
1C72  E4F1  XRI F1
1C74  9C0F  JNZ $ 10
1C76  C463  LDI 63
1C78  C9FF  ST FF (1)
1C7A  01    XAE
      $ 8
1C7B  5C    ??
1C7C  08    NOP
1C7D  9402  JP $ 9
1C7F  90FA  JMP $ 8
      $ 9
1C81  8FFF  DLY FF
1C83  9041  JMP $ 20
      $ 10
1C85  C108  LD 08 (1)
1C87  E4F2  XRI F2
1C89  9C0F  JNZ $ 13
1C8B  C45C  LDI 5C
1C8D  C900  ST 00 (1)
1C8F  01    XAE
      $ 11
1C90  C108  LD 08 (1)
1C92  9402  JP $ 12
1C94  90FA  JMP $ 11
      $ 12
1C96  8FFF  DLY FF
1C98  902C  JMP $ 20
      $ 13
1C9A  C108  LD 08 (1)
1C9C  E4F3  XRI F3
1C9E  9C0F  JNZ $ 16
1CA0  C45C  LDI 5C
1CA2  C904  ST 04 (1)
1CA4  01    XAE
      $ 14
1CA5  C108  LD 08 (1)
1CA7  9402  JP $ 15
1CA9  90FA  JMP $ 14
      $ 15
1CAB  8FFF  DLY FF
1CAD  9058  JMP $ 31
      $ 16
1CAF  C108  LD 08 (1)
1CB1  E4F4  XRI F4
1CB3  9C13  JNZ $ 21
1CB5  C463  LDI 63
1CB7  C903  ST 03 (1)
1CB9  01    XAE
      $ 17
```

```
1CBA C108 LD 08 (1)
1CBC 9402 JP $ 18
1CBE 90FA JMP $ 17
     $ 18
1CC0 8FFF DLY FF
1CC2 9002 JMP $ 20
     $ 19
1CC3 02   CCL
1CC4 9095 JMP $  4
     $ 20
1CC6 9041 JMP $ 31
     $ 21
1CC8 C108 LD 08 (1)
1CCA E4F5 XRI F5
1CCC 9C0F JNZ $ 24
1CCE C463 LDI 63
1CD0 C901 ST 01 (1)
1CD2 01   XAE
     $ 22
1CD3 C108 LD 08 (1)
1CD5 9402 JP $ 23
1CD7 90FA JMP $ 22
     $ 23
1CD9 8FFF DLY FF
1CDB 902C JMP $ 31
     $ 24
1CDD C108 LD 08 (1)
1CDF E4F6 XRI F6
1CE1 9C0F JNZ $ 27
1CE3 C45C LDI 5C
1CE5 C902 ST 02 (1)
1CE7 01   XAE
     $ 25
1CE8 C108 LD 08 (1)
1CEA 9402 JP $ 26
1CEC 90FA JMP $ 25
     $ 26
1CEE 8FFF DLY FF
1CF0 9017 JMP $ 31
     $ 27
1CF2 C108 LD 08 (1)
1CF4 E4F7 XRI F7
1CF6 9C0F JNZ $ 30
1CF8 C45C LDI 5C
1CFA C906 ST 06 (1)
1CFC 01   XAE
     $ 28
1CFD C108 LD 08 (1)
1CFF 9402 JP $ 29
1D01 90FA JMP $ 28
     $ 29
1D03 8FFF DLY FF
1D05 9002 JMP $ 31
     $ 30
1D07 90BA JMP $ 19
     $ 31
1D09 C108 LD 08 (1)
1D0B E4F0 XRI F0
1D0D 9C13 JNZ $ 34
1D0F C463 LDI 63
1D11 60   XRE
1D12 9874 JZ $ 48
1D14 C45C LDI 5C
1D16 C901 ST 01 (1)
     $ 32
1D18 C108 LD 08 (1)
1D1A 9402 JP $ 33
1D1C 90FA JMP $ 32
     $ 33
1D1E 8FFF DLY FF
1D20 90E5 JMP $ 30
     $ 34
1D22 C108 LD 08 (1)
1D24 E4F1 XRI F1
1D26 9C13 JNZ $ 37
1D28 C463 LDI 63
1D2A 60   XRE
1D2B 985B JZ $ 49
1D2D C45C LDI 5C
1D2F C900 ST 00 (1)
     $ 35
1D31 C108 LD 08 (1)
1D33 9402 JP $ 36
1D35 90FA JMP $ 35
     $ 36
1D37 8FFF DLY FF
1D39 90C0 JMP $ 34
     $ 37
1D3B C108 LD 08 (1)
1D3D E4F2 XRI F2
1D3F 9C13 JNZ $ 40
1D41 C463 LDI 63
1D43 60   XRE
1D44 9842 JZ $ 48
1D46 C463 LDI 63
1D48 C906 ST 06 (1)
     $ 38
1D4A C108 LD 08 (1)
1D4C 9402 JP $ 39
1D4E 90FA JMP $ 38
     $ 39
1D50 8FFF DLY FF
1D52 90B3 JMP $ 30
     $ 40
1D54 C108 LD 08 (1)
1D56 E4F3 XRI F3
1D58 9C15 JNZ $ 45
1D5A C463 LDI 63
1D5C 60   XRE
1D5D 9829 JZ $ 48
1D5F C45C LDI 5C
1D61 C905 ST 05 (1)
     $ 41
1D63 C108 LD 08 (1)
1D65 9402 JP $ 42
1D67 90FA JMP $ 41
     $ 42
1D69 8FFF DLY FF
     $ 43
1D6B 909A JMP $ 30
     $ 44
1D6D 909A JMP $ 31
     $ 45
1D6F C108 LD 08 (1)
1D71 E4F4 XRI F4
1D73 9C17 JNZ $ 50
1D75 C45C LDI 5C
1D77 60   XRE
1D78 980E JZ $ 48
1D7A C463 LDI 63
1D7C C903 ST 03 (1)
     $ 46
1D7E C108 LD 08 (1)
1D80 9402 JP $ 47
1D82 90FA JMP $ 46
     $ 47
1D84 8FFF DLY FF
1D86 9002 JMP $ 49
     $ 48
1D88 904F JMP $ 61
     $ 49
1D8A 90DF JMP $ 43
     $ 50
1D8C C108 LD 08 (1)
1D8E E4F5 XRI F5
1D90 9C13 JNZ $ 53
1D92 C45C LDI 5C
1D94 60   XRE
1D95 9842 JZ $ 61
1D97 C45C LDI 5C
1D99 C902 ST 02 (1)
     $ 51
1D9B C108 LD 08 (1)
1D9D 9402 JP $ 52
1D9F 90FA JMP $ 51
     $ 52
1DA1 8FFF DLY FF
1DA3 90E5 JMP $ 49
     $ 53
1DA5 C108 LD 08 (1)
1DA7 E4F6 XRI F6
1DA9 9C13 JNZ $ 56
1DAB C45C LDI 5C
1DAD 60   XRE
1DAE 9829 JZ $ 61
1DB0 C45C LDI 5C
1DB2 C9FF ST FF (1)
     $ 54
1DB4 C108 LD 08 (1)
1DB6 9402 JP $ 55
1DB8 90FA JMP $ 54
     $ 55
1DBA 8FFF DLY FF
1DBC 90CC JMP $ 49
     $ 56
1DBE C108 LD 08 (1)
1DC0 E4F7 XRI F7
1DC2 9C13 JNZ $ 59
1DC4 C45C LDI 5C
1DC6 60   XRE
1DC7 9810 JZ $ 61
1DC9 C463 LDI 63
1DCB C904 ST 04 (1)
     $ 57
1DCD C108 LD 08 (1)
1DCF 9402 JP $ 58
1DD1 90FA JMP $ 57
     $ 58
```

```
1DD3  8FFF  DLY FF
      $ 59
1DD5  3063  JMP $ 49
      $ 30
1DD7  3044  JMP $ 44
      $ 44
1DD9  C45C  LDI 5C
1DDB  C9FF  ST FF (1)
1DDD  C463  LDI 63
1DDF  C901  ST 01 (1)
1DE1  40    LDE
1DE2  C905  ST 05 (1)
1DE4  90EF  JMP $ 59
```

Monopoly Würfel

```
1E00  C400  LDI 00
1E02  C906  ST 06 (1)
1E04  C905  ST 05 (1)
1E06  C904  ST 04 (1)
1E08  C902  ST 02 (1)
1E0A  C901  ST 01 (1)
1E0C  C900  ST 00 (1)
1E0E  C9FF  ST FF (1)
1E10  C406  LDI 06
1E12  C903  ST 03 (1)
1E14  01    XAE
1E15  8F0F  DLY 0F
1E17  C108  LD 08 (1)
1E19  9403  JP $ 1
1E1B  00    HALT
1E1C  9048  JMP $ 7
      $ 1
1E1E  C45B  LDI 5B
1E20  C903  ST 03 (1)
1E22  01    XAE
1E23  8F0F  DLY 0F
1E25  C108  LD 08 (1)
1E27  9403  JP $ 2
1E29  00    HALT
1E2A  903B  JMP $ 8
      $ 2
1E2C  C44F  LDI 4F
1E2E  C903  ST 03 (1)
1E30  01    XAE
1E31  8F0F  DLY 0F
1E33  C108  LD 08 (1)
1E35  9403  JP $ 3
1E37  00    HALT
1E38  902C  JMP $ 7
      $ 3
1E3A  C466  LDI 66
1E3C  C903  ST 03 (1)
1E3E  01    XAE
1E3F  8F0F  DLY 0F
1E41  C108  LD 08 (1)
1E43  9403  JP $ 4
1E45  00    HALT
1E46  901E  JMP $ 7
      $ 4
1E48  C46D  LDI 6D
1E4A  C903  ST 03 (1)
1E4C  01    XAE
1E4D  8F0F  DLY 0F
1E4F  C108  LD 08 (1)
1E51  9403  JP $ 5
1E53  00    HALT
1E54  9010  JMP $ 7
      $ 5
1E56  C47D  LDI 7D
1E58  C903  ST 03 (1)
1E5A  01    XAE
1E5B  8F0F  DLY 0F
1E5D  C108  LD 08 (1)
1E5F  9403  JP $ 6
1E61  00    HALT
1E62  9002  JMP $ 7
      $ 6
1E64  909A  JMP $ 0
      $ 7
```

SC/MP-1-Platinen-Mikrocomputer für Automations-, Regel- und Steuer-Anwendungen

Auf einer durchkontaktierten Europa-Platine sind folgende Komponenten untergebracht:

- CPU INS 8060
- 3 K EPROM MM 5204Q
- 1/2 K RAM
- 32 programmierbare I/O-Lines
- Adressdecodierung und 64-poliger DIN-Stecker.

Die Platine kann in folgenden 2 Arten betrieben werden:

1. Ohne eigene CPU kann sie mit dem DIN-Stecker oder über ein Adapter-Kabel an alle gängigen SC/MP-Entwicklungs-Systeme angeschlossen werden. Sie ist dann Bestandteil des Entwicklungs-Systems und kann über dessen Betriebssystem angesprochen werden.
   Diese Betriebsart ist in der Programmentwicklungszeit sehr nützlich. Das zu testende Programm kann in beliebige RAM-Bereiche des Systems geladen werden.

   Die Karte kann aber auch als 3 K EPROM- und/oder 32 I/O-Lines-Karte auf dem Entwicklungssystem verbleiben.

2. Nachdem das Programm befriedigend läuft wird eine entsprechende Anzahl EPROM's programmiert, die Karte vom Entwicklungssystem getrennt, mit diesen EPROM's und einer CPU bestückt und funktioniert dann als selbständiger 1-Karten-Mikrocomputer. Eine beliebige externe Erweiterung ist jederzeit möglich, da der gesamte Daten-, Adress- und Steuer-Bus an den DIN-Stecker herausgeführt ist. Im Fall einer Störung oder später notwendigen Änderung des Programms kann die Platine wieder mit dem Entwicklungssystem verbunden werden und so die Fehlersuche wesentlich erleichtert werden.

Informationen bei: Dr. Huschitt,
Bibliser Weg 29-31
6840 Lampertheim 5

SC/MP-1-Karten- Mikrocomputer

Diese Europakarte ist ein selbstständiger Mikrocomputer mit dem SC/MP II (INS 8060) als CPU, 3 K EPROM, 1/2 K RAM und 39 I/O-Lines.

Der gesamte Daten-, 12-Bit-Adress- und Steuer-Bus ist an einen 64-poligen DIN-Stecker (41612/C 64) herausgeführt, so daß eine externe Erweiterung jederzeit möglich ist.

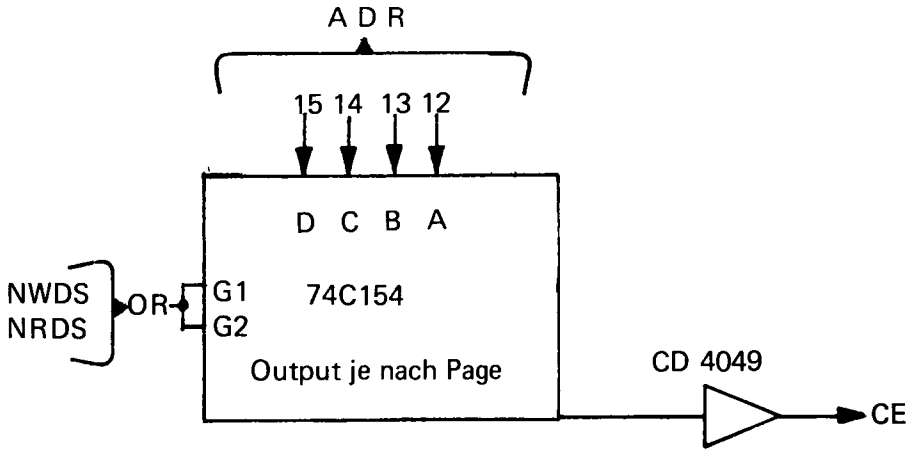
Programmentwicklung

Letztlich soll diese Karte mit eigener CPU und lauffähigem Programm in PROM's in einer Anwendung ihr Eigenleben führen. Bis es aber soweit ist muß das Programm entwickelt und getestet werden. Dies geschieht mit einem möglichst komfortablen Entwicklungssystem.

Ohne eigenen CPU kann diese Karte an jedes beliebige SC/MP-Entwicklungssystem angeschlossen werden. Sie ist dann funktionsmäßig Bestandteil des betreffenden Systems.

Auf das Original-ELEKTOR-SC/MP-System kann sie ohne jede Änderung oder zusätzlichen Aufwand direkt aufgesteckt werden, sie ist Elektor-Bus-kompatibel.

Zum Anschluß an andere SC/MP-Systeme ist folgendes zu beachten: Um adressierbar zu sein benötigt die Karte ein 4-K-CARD-ENABLE. Das Elektor-System besitzt dieses Signal für Page 1 (17a Elektor-Bus). Bei anderen Systemen lässt sich dieses Signal, soweit nicht vorhanden nach folgendem Vorschlag erzeugen:



Die Europakarte kann auf jede beliebige Page gelegt werden, sie ignoriert die Page-Adresse (Adr.-Bit 12 bis 15) sowieso. Folgende Leitungen müßen noch vom Entwicklungssystem an den SC/MP-1-Karten-Mikrocomputer geführt werden:

NRST
DATA 0 bis 7
ADR. 0 bis 11
NWDS
NRDS
V_{cc}
GND

am einfachsten über ein Flachkabel an einen 40-pol-DIL-Stecker der in den (im Entwicklungsstadium ohnehin leeren) CPU-Sockel gesteckt wird

- 12 V an die - 12 V Leitung der Eurokarte

Das 4-K-CE-Siganl wird an Lötpunkt 10 der Europa-Karte geführt.

Im Entwicklungsstadium wird das Programm zunächst im RAM laufen. Es ist am günstigsten eine 4-K-RAM-Karte auf die gleiche Page (!) mit dem SC/MP-1-Karten-Mikrocomputer zu legen. Lediglich die RAM's von Adr. XC00 bis XFFF müssen von dieser RAM-Karte entfernt werden. Auf diesen Adressen befinden sich die beiden RAM-I/O und die beiden MM2112, die im Entwicklungsstadium natürlich auf der Europakarte bleiben.

Das Programm soll zweckmäßigerweise bei X000 mit NOP beginnen. Es kann dann ohne jede Änderung in die PROM's übernommen werden.

Hinweise für den Aufbau

Die Lötseite der Platine ist an der grünen Lötstopmaske und an der Aufschrift "LÖTSEITE" erkennbar.
Die üblichen Belehrungen über Lötkolben, Lot, Umgang mit IC's usw. sollen hier, da als bekannt vorausgesetzt, nicht wiederholt werden.

Es empfiehlt sich folgendes Vorgehen:
- sämtliche passive Bauteile (Widerstände, Kondensatoren, IC-Sockel und Steckerleiste) einlöten. Verbindung mit dem Entwicklungs-System herstellen (im einfachsten Fall=Elektor, auf den Bus stecken), das Entwicklungssystem muß noch arbeiten. Wenn nicht, Lötfehler suchen.
- Lötpunkte 9 und 10 verbinden (nur falls Elektor), IC 2 und 3 montieren und ein programmiertes PROM in einen PROM-Sockel stecken. Es muß mit Modify auf den Adressen 1000 bis 1A00 (bei anderen Systemen als Elektor auf X000 bis XA00, je nach verwendeter Page) je nach Sockel zu finden sein.
- RAM und RAM - I/O montieren. Sie müssen auch auf den entsprechenden Adressen zu finden sein, (s. Adressplan).

Vor der Inbetriebnahme der Karte als selbstständiger Mikrocomputer, welche das Vorhandensein eines lauffähigen Programms in zumindest einem 5204 voraussetzt, muß noch die Funktion der diversen Lötbrücken erläutert werden.
Sie sind auf dem Bestückungsplan und auf dem Schaltplan gleich numeriert.

Mit den Verbindungen 1 - 2 , 3 - 4 und 5 - 6 werden NHOLD und CONT an V_{CC} und NENIN an GND gelegt. Diese Brücken brauchen nur im selbständigen Stadium eingesetzt zu werden. Wenn in einem größerem System von diesen SC/MP-Eingängen Gebrauch gemacht werden soll, kann man sie durch 4k7-Widerstände ersetzen.
Mit den Lötpunkten 14 bis 17 kann wahlweise ein RAM-I/O oder das

RAM (MM2112) an die obere Page-Grenze gelegt werden. Die Verbindungen 14-15 und 16-17 ergeben: 1E00=RAM, 1F00=RAM-I/O. Durch die Verbindungen 14 - 16 und 15 - 17 wird die Adresse dieser beiden Bauteile vertauscht.
Die Lötpunkte 11, 12 und 13 sind zu verwenden wenn der 2 MHZ-Takt von der Karte in einer externen Erweiterung benötigt wird (11 - 12) oder wenn ein Takt vorhanden ist und der Quarz eingespart werden kann (12 - 13).
Lötpunkt 10 führt zum CE des Adressdekoders. Hier wird in der Betriebsart "ohne eigene CPU" ein 4--K-CE benötigt. Auf dem Elektor-System liegt ein solches für Page 1 vor: 9 mit 10 verbinden. An Punkt 8 steht das selbst erzeugte NRDS-NWDS-OR-Signal zur Verfügung. Es kann benutzt werden wenn die Karte als selbständiger Mikrocomputer arbeitet: 8 mit 10 verbinden.
Bei größeren Konfigurationen, welche auf dieser Karte basierend extern, durch das Heranziehen der 4 höchstwertigsten Adressbits vollständige Adressdekodierung erlauben, (64 K) kann dieses NRDS-NWDS-OR-Signal herausgeführt und ein 4-K-CE hereingeführt werden: 7 mit 8 und 9 mit 10 verbinden.
Lötpunkt 18 geht an SENSE A. Hier kann wahlweise der Interrupt-Ausgang von IC 13 (Lötpunkt 19) oder von IC 14 (Lötpunkt 20) angeschlossen werden.

Die Reihe von Lötpunkten unterhalb von IC 2 und 3 sind CE-Leitungen, welche zur Adressierung von externen Komponenten herangezogen werden können. (XC00, XD00 usw.)

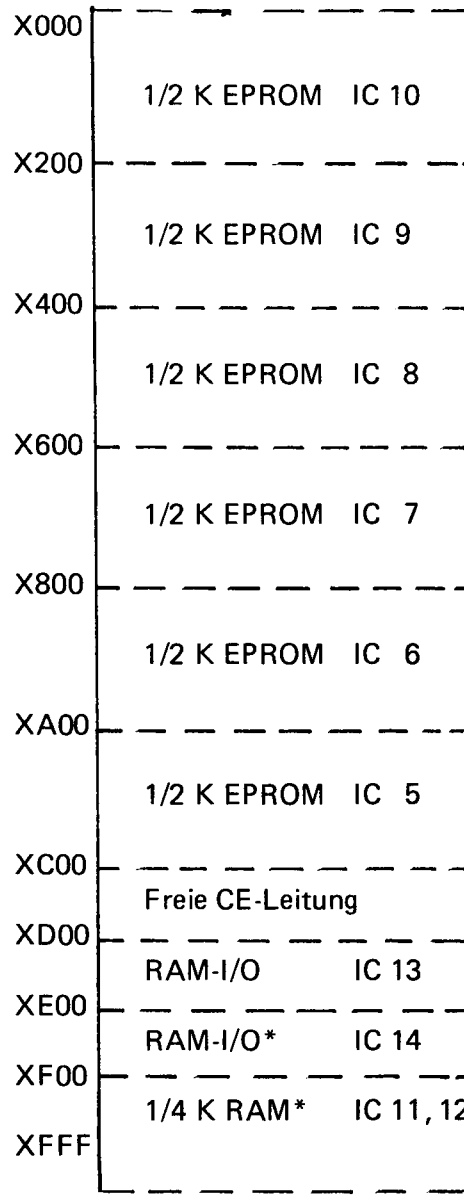
Stückliste (bei vollem Ausbau)

IC 1 SC/MP II
IC 2, 3 74LS155
IC 4 74LS00
IC 5 bis 10 MM 5204 Q
IC 11, 12 MM2112
IC 13,14 RAM-I/O
R 1 bis 4 4k7
R 5 100 k
R 6 1 k
Quarz 2 MHz
C 1 bis 6 1uF 10V
C 7 20 pF

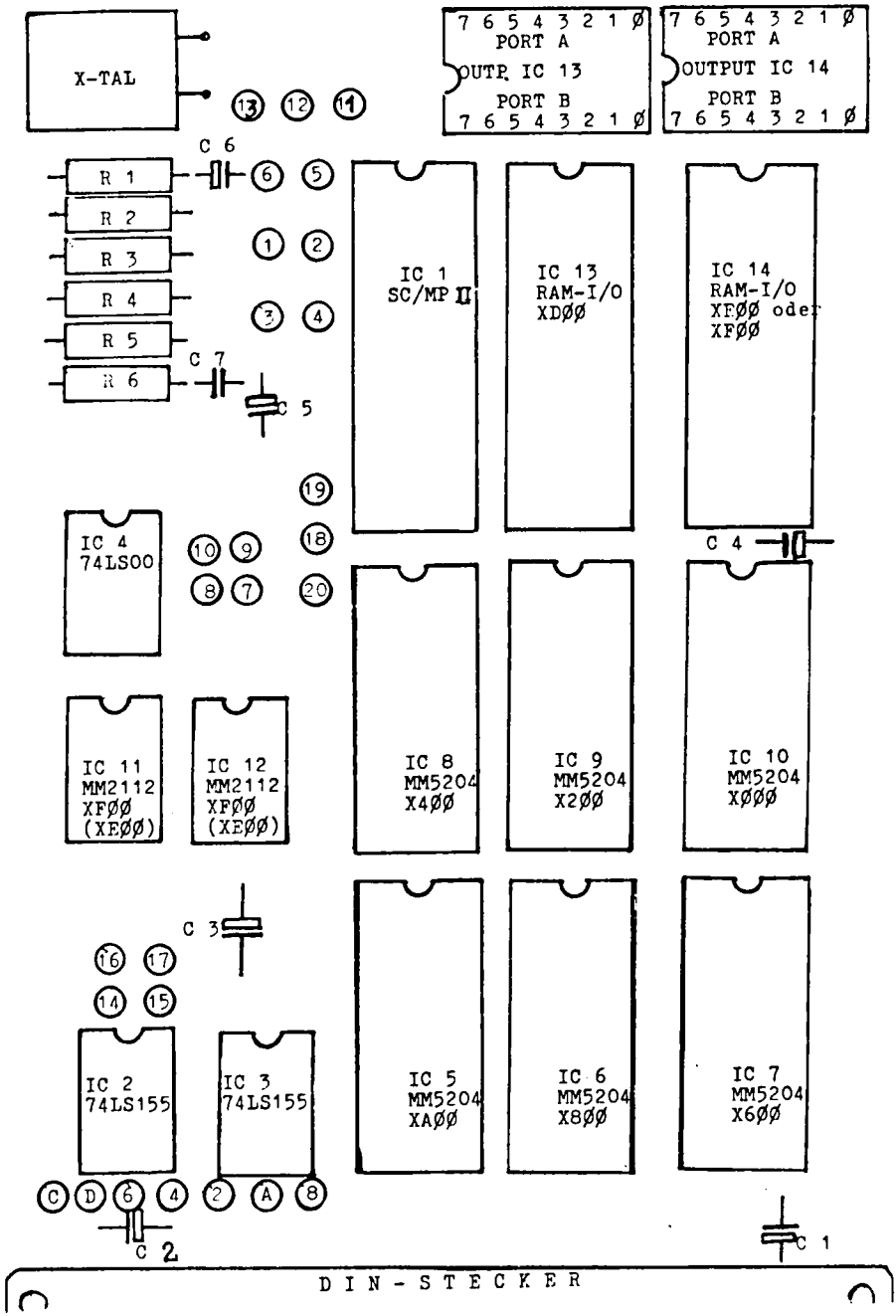
Stecker DIN 41612/C 64

	c	a
1	+5 V	+5 V
2		
3	-12 V	-12 V
4	GND	GND
5	NHOLD	NRST
6		NBREQ
7	DB 0	DB 1
8	DB 2	DB 3
9	DB 4	DB 5
10	DB 6	DB 7
11	CONT	NENIN
12	SENSE A	SENSE B
13	SIN	SOUT
14	FLAG 0	FLAG 1
15	FLAG 2	
16	GND	GND
17		Page-Enable
18	NENOUT	
19		
20		
21	AD 10	AD 11
22	AD 8	AD 9
23	AD 6	AD 7
24	AD 4	AD 5
25	AD 2	AD 3
26	AD 0	AD 1
27	NRDS+NWDS	X-Tal
28		
29		
30		NADS
31	NRDS	NWDS
32	GND	GND

Adressplan



B E S T Ü C K U N G S P L A N



RAM-I/O (INS 8154)

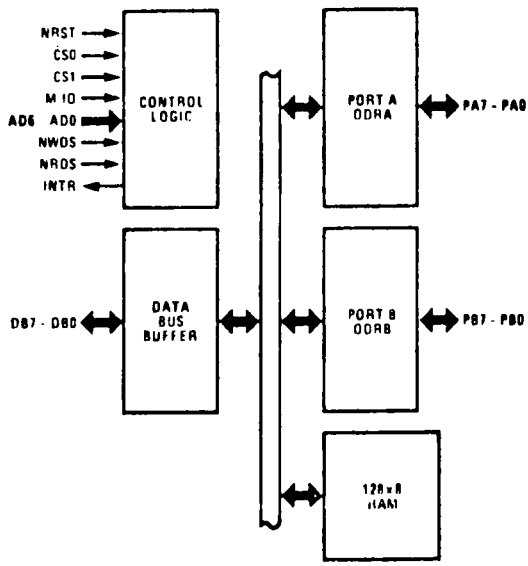
Dieses 40-Pin-DIL-IC beinhaltet 128 x 8 Bit statisches RAM und 2 programmierbare I/O-Ports zu je 8 Bit. Alle Eingänge und Ausgänge sind TTL-kompatibel. Der Baustein hat 8 Daten- und 7 Adress-Anschlüsse. Zum Anwählen stehen 2 Chip-Select-Eingänge zur Verfügung (High- und low-aktiv). M/IO (Pin 33) dient zum Selektieren des RAM (high) oder des I/O-Anteils (low). Dieser Anschluß kann als 8. Adresseingang (ADDR.-Bit 7) benutzt werden. Es ergibt sich dann folgende Adressorganisation, die auch für den SC/MP-1-Karten-Microcomputer

XX00	Bit 0 Port A löschen	
XX01	1	
XX02	2	
XX03	3	
XX04	4	
XX05	5	
XX06	6	
XX07	7	
XX08	Bit 0 Port B löschen	
XX09	1	
XX0A	2	
XX0B	3	
XX0C	4	
XX0D	5	
XX0E	6	
XX0F	7	Einzelbitoperationen
XX10	Bit 0 Port A setzen	
XX11	1	
XX12	2	
XX13	3	
XX14	4	
XX15	5	
XX16	6	
XX17	7	
XX18	Bit 0 Port B setzen	
XX19	1	
XX1A	2	
XX1B	3	

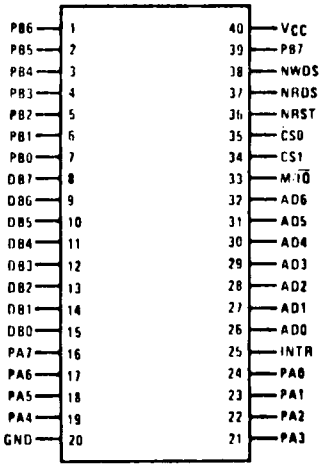
XX1C	4
XX1D	5
XX1E	6
XX1F	7
XX20	Port A
XX21	Port B
XX22	ODRA (= Output Definition Register A)
XX23	ODRB (= Output Definition Register B)
XX24	MDR (= Mode Definition Register)
.	
.	unbenutzte Adressen
.	
XX80	
.	
.	1/8 K RAM
XXFF	

Die 3 Definiton Register sind "Nur-Schreib-Register" d h. man kann sie mit einem ST-Befehl mit einem gewünschten Inhalt laden nicht aber mit einem LD-Befehl ihren Inhalt in den AC holen. Alle 3 Register sind nach einem Low-Signal an dem NRST-Eingang gelöscht.

Basic Block Diagram



Pin Configuration



Pin Names

DB7 - DB0	DATA BUS
AD6 - AD0	ADDRESS INPUT
NRST	RESET INPUT
M/IO	MEMORY IO SELECT
CS0, CS1	CHIP SELECTS
NWDS	WRITE STROBE
NRDS	READ STROBE
PA7 - PA0	PORT A
PB7 - PB0	PORT B
INTR	INTERRUPT REQUEST
VCC	+5 VOLTS
GND	0 VOLTS

Absolute Maximum Ratings*

Voltage at any Pin	- 0,5 V to +7,0 V
Operating Temperature Range	0° C to +70° C
Storage Temperature Range	- 65° C to +150° C
Lead Temperature (Soldering, 10 seconds)	300° C

* Absolute Maximum Ratings are those values beyond which the safety of the device cannot be guaranteed. Continuous operation at these limits is not intended; operation should be limited to those conditions specifed under Electrical Characteristics.

INTR (Pin 25) ist der Interrupt-Ausgang des IC. Wenn Port A als mehrfacher Interrupt-Eingang benutzt wird, erscheint an diesem Ausgang ein High-aktives Signal bei jeder Interrupt-Anforderung. Nach einem Reset ist dieser Ausgang low.
Mit dem MDR kann man Port A für 4 verschiedene Betriebsarten programmieren, Port B ist immer in dem Modus 1.

DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Bit Location
TS	OUT	M	—	—	—	—	—	MDR Bit Name
X	X	0	X	X	X	X	X	Basic I/O
X	0	1	X	X	X	X	X	Strobed Input
0	1	1	X	X	X	X	X	Strobed Output
1	1	1	X	X	X	X	X	Strobed Output with TRI-STATE Control

MODUS 1 = BASIC I/O

Port B arbeitet immer in diesem Modus, Port A nach einem low-Signal an NRST oder nachdem 0C in das MDR geschrieben wurde. Über die Output Definition Register A und B wird festgelegt,welches Bit,welcher Port Eingang und welches Ausgang ist. Eine "0" im ODR bedeutet,daß das betreffende Bit der betreffende Port als Eingang arbeitet und eine "1",daß es als Ausgang arbeitet. Z.B. LDI 03

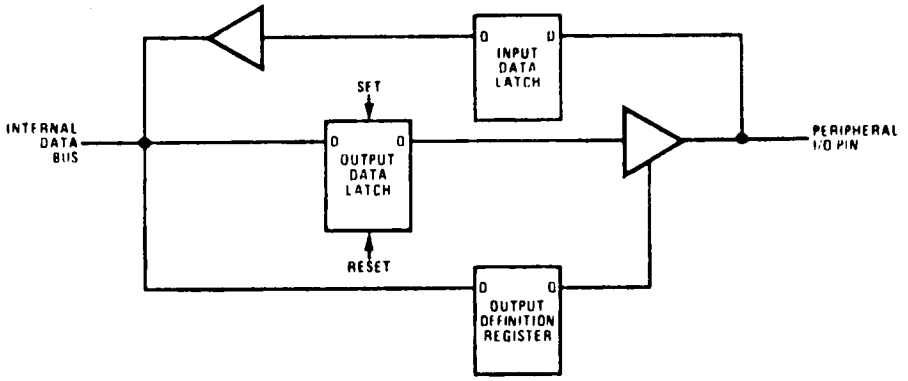
ST ODRA(2)

bewirkt,daß Bit 1 und 0 der Port A als Ausgang und Bit 2 bis 7 als Eingang arbeiten.

Mit einem ST-Befehl kann ein beliebiges Byte in die Port geschrieben werden und, sofern als Ausgang definiert, werden die zur betreffende Port gehörigen Pins den entsprechenden TTL-Pegel annehmen. Dieser bleibt bis zum Überschreiben mit einem anderen Byte, dem Umprogrammieren als Eingang oder bis zu einem NRST erhalten. Als Eingang definierte Pins können mit einem LD-Befehl abgefragt werden. Der gesamte Inhalt der Port erscheint als ein Byte im Ac.

Außer diesem parallelen Zugriff auf die Ports gibt es noch die Möglichkeit der Einzel-Bit-Operationen. Ein ST-Befehl, gleichgültig mit welchem AC-Inhalt (!) auf eine der Adressen XX00 bis XX1F bewirkt ein selektives Löschen oder Setzen eines einzelnen Bit ohne die anderen zu verändern. Voraussetzung ist natürlich, daß das betreffende Bit als Ausgang arbeitet.

Mit einem LD-Befehl bezogen auf eine der o.a. Adressen kann das Bit alleine in AC Bit 7 geholt werden. Die restlichen AC-Bits sind nach einer solchen Leseoperation immer 0. Der (AC) wird also entweder 80 oder 00 sein.



Internal Logic of One Bit of an I/O Port with ODR

MODUS 2 = Strobed Input (Port A)

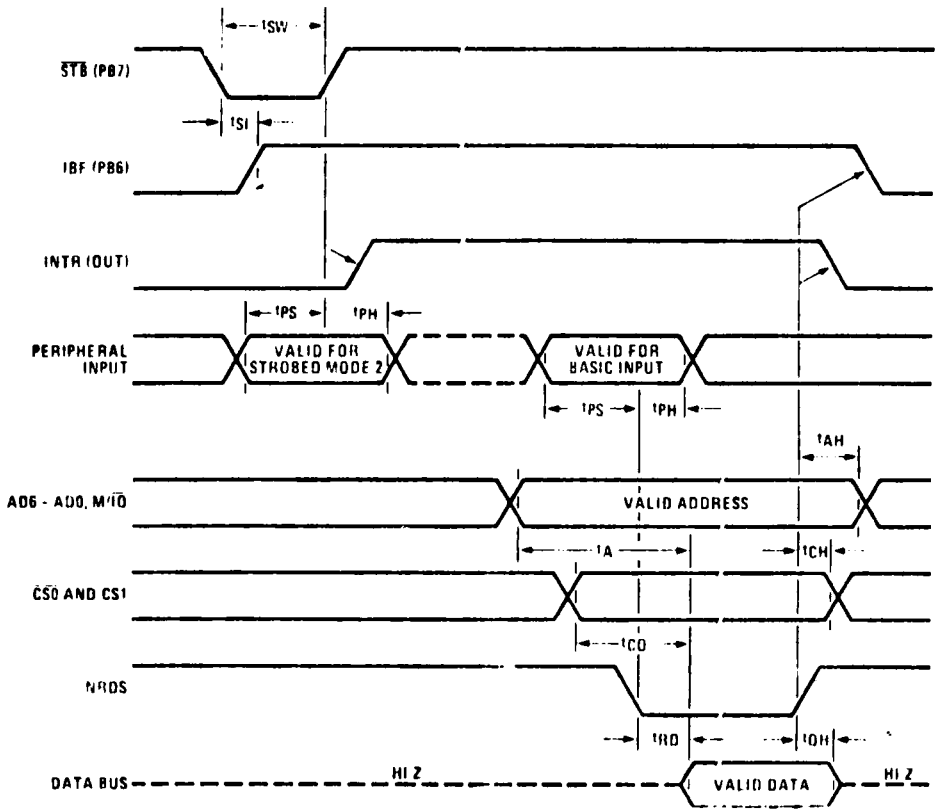
Dieser Modus erlaubt einen Datentransfer in 2 Etappen. Ein kurzfristig am Ausgang einer Peripherie vorhandener Zustand wird mittels eines von dieser Peripherie zu generierenden Strobe-Impulses in Port A geschreiben und dort zwischengespeichert. Die CPU kann zu einem beliebigen späteren Zeitpunkt die Daten aus Port A abholen und verarbeiten.

Der Strobe-Impuls von der Peripherie wird an Bit 7 Port B geführt. Das RAM-I/O generiert ein "Input-Buffer-Full"-Signal (Bit 6 Port B als Ausgang), welches der betreffende Peripherie anzeigt, daß die Daten aus Port A noch nicht abgeholt sind. IBF (Input Buffer Full) wird von der abfallenden Flanke des STB (Strobe-Impuls) gesetzt und von der ansteigenden Flanke des NRDS der nächsten Leseoperation der Port A (NRDS mit anderen Zieladressen nicht !) rückgesetzt.
Port B hat in diesem Modus nur noch 6 Bit zur freien Verfügung. Bit 7

Port B hat eine doppelte Funktion. Als Eingang für den erwähnten Strobe und als IE (Interrupt Enable Flag) wird das Output-FF dieses Bit über eine UND-Verknüpfung an den INT-Ausgang des RAM-I/O geführt. Mit einer Einzelbit-Operation wie in Modus 1 beschrieben, kann dieses FF gesetzt (=Interrupt erlaubt) oder gelöscht (= Interrupt gesperrt) werden. Falls gesetzt bewirkt die ansteigende Flanke des STB einen High-Zustand des INT, falls das Output-FF Bit 7 Port B 0 ist bleibt INT low.

Der parallele Zugriff auf Port B ist in diesem Modus erlaubt, Bit 6 und 7 werden von Schreibe und Lese-Operationen nicht verändert.

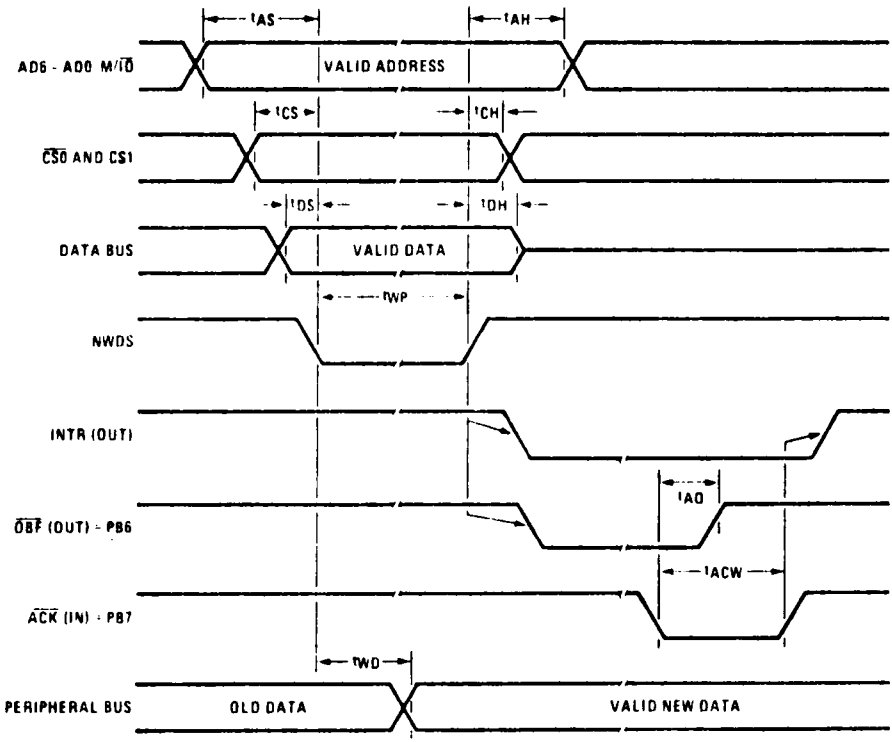
Strobed Input, Mode 2 Timing



MODUS 3 = Strobed Output (Port A)

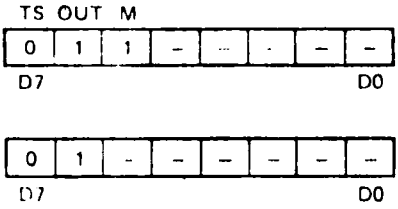
Diese Betriebsart erlaubt die Ausgabe von Daten an eine asynchrone Peripherie.

Die CPU schreibt die Daten in Port A. Das RAM-I/O generiert daraufhin ein OBF-Signal, (=Output-Buffer-Full - Bit 6 Port B) welches der Peripherie anzeigt, daß neue Daten zur Verfügung stehen. Die Peripherie kann die Daten zu einem beliebigen späteren Zeitpunkt abholen. Dabei soll sie ein ACK-Signal (=Acknowledge) aktivieren, welches an Port B Bit 7 geführt und dort vom Programm abgefragt, der CPU mitteilt, daß die Daten von der Peripherie abgeholt wurden. Diese ACK setzt auch das OBF zurück. Wie in Modus 2 steht auch hier Port B mit 6 Bit, auch für paralellen Zugriff zur freien Verfügung.

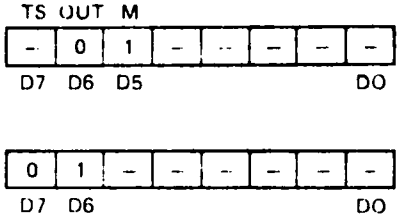


Strobed Output, Mode 3 Timing

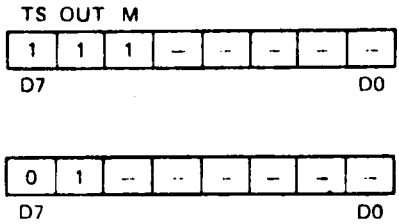
Wichtig
Voreinstellen der MDR und
ODRB für die diversen Modi



ODRA = "1s" at mode 3 pins.



ODRA = "0s" at mode 2 pins.

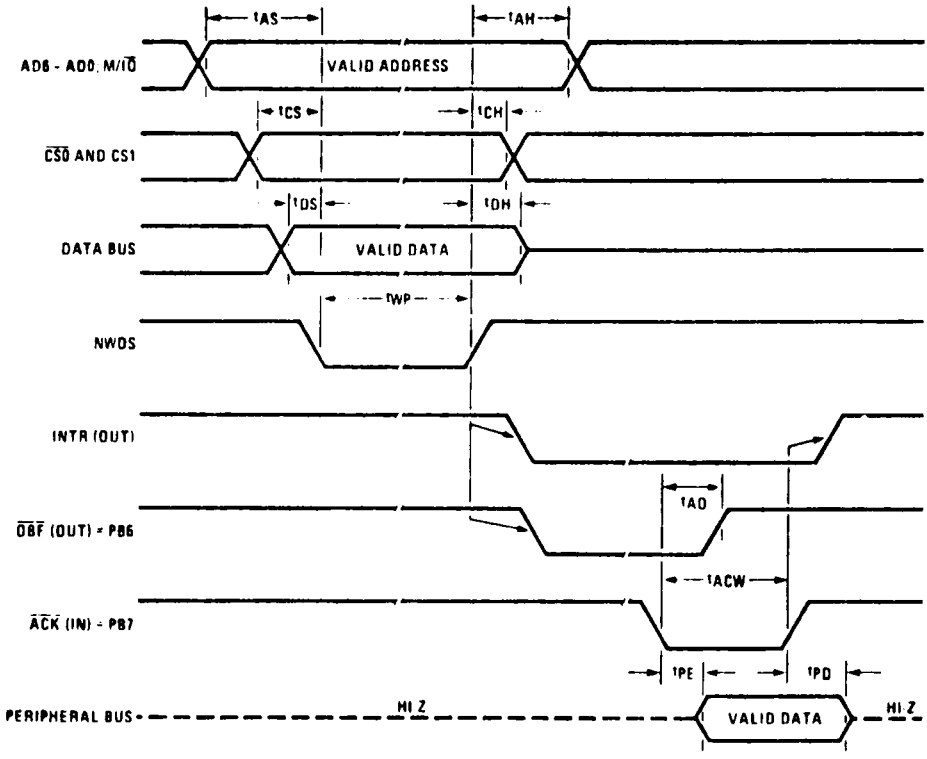


ODRA = "1s" at mode 4 pins.

MODUS 4 = Strobed Output mit Tri-State

Unterscheidet sich von Modus 3 nur dadurch, daß die Ausgänge der Port A in hochohmigem Zustand sind, außer wenn das von der Peripherie generierte ACK aktiv ist.

Dies ermöglicht den Aufbau eines peripheren Datenbusses.



Strobed Output with TRI-STATE Mode 4 Timing

Programm zum Abfragen einer 3 x 8 Tastenmatrix und zum Auffrischen einer 8-stelligen 7-Segment-Anzeige (gem. Schaltung)

Adressplan:

Ptr 2 muß vor dem Auffruf der Subroutine als "RAM-I/O"-Pointer mit der Adr. des ersten RAM-Byte eines RAM-I/O (z.B. 1F80) geladen sein.

```
          + A0 = Port A
          + A1 = Port B
          + A2 = ODRA
          + A3 = ODRB
          + A4 = MDR
            .
            .
(PTR 2) + 00 = Vielzweckzähler
            .
            .     freier RAM-Bereich
            .
          + 13 = Digit 0 ┐
          + 14 = Digit 1 │  RAM-Bereich in
            .            ▶ welchem die Display-
            .            │  Anzeigen als 7-Segm-Code
          + 1A = Digit 7 ┘  hinterlegt sein muß.
          + 1B = "Modus"-Flag; s.Text.
          + 1C = Ergebnis des Tastendrucks
          + 1D = Zwischenspeicher
                 freies RAM
```

Die Subroutine hat zwei Ansprungslabel: 1100 und 1106. Je nach Ansprung ergibt sich ein anderer Arbeits-Modus. Beim Ansprung des Labels 1100 wird die Subroutine erst verlassen, wenn eine Taste gedrückt und wieder losgelassen wurde. Im anderen Fall (1106) erfolgt ein Display-Refresh und ein Scannen der Tastenmatrix mit sofortigem Tücksprung in das Hauptprogramm (über PTR 3). Wenn während dieser Zeit eine Taste gedrückt war steht ihr Ergebnis in der RAM-Zelle (PTR 2) + 1C, wenn nicht steht 00 darin.
Vorherige Inhalte von AC, E, SR und PTR 1 werden von diesem Programm zerstört.

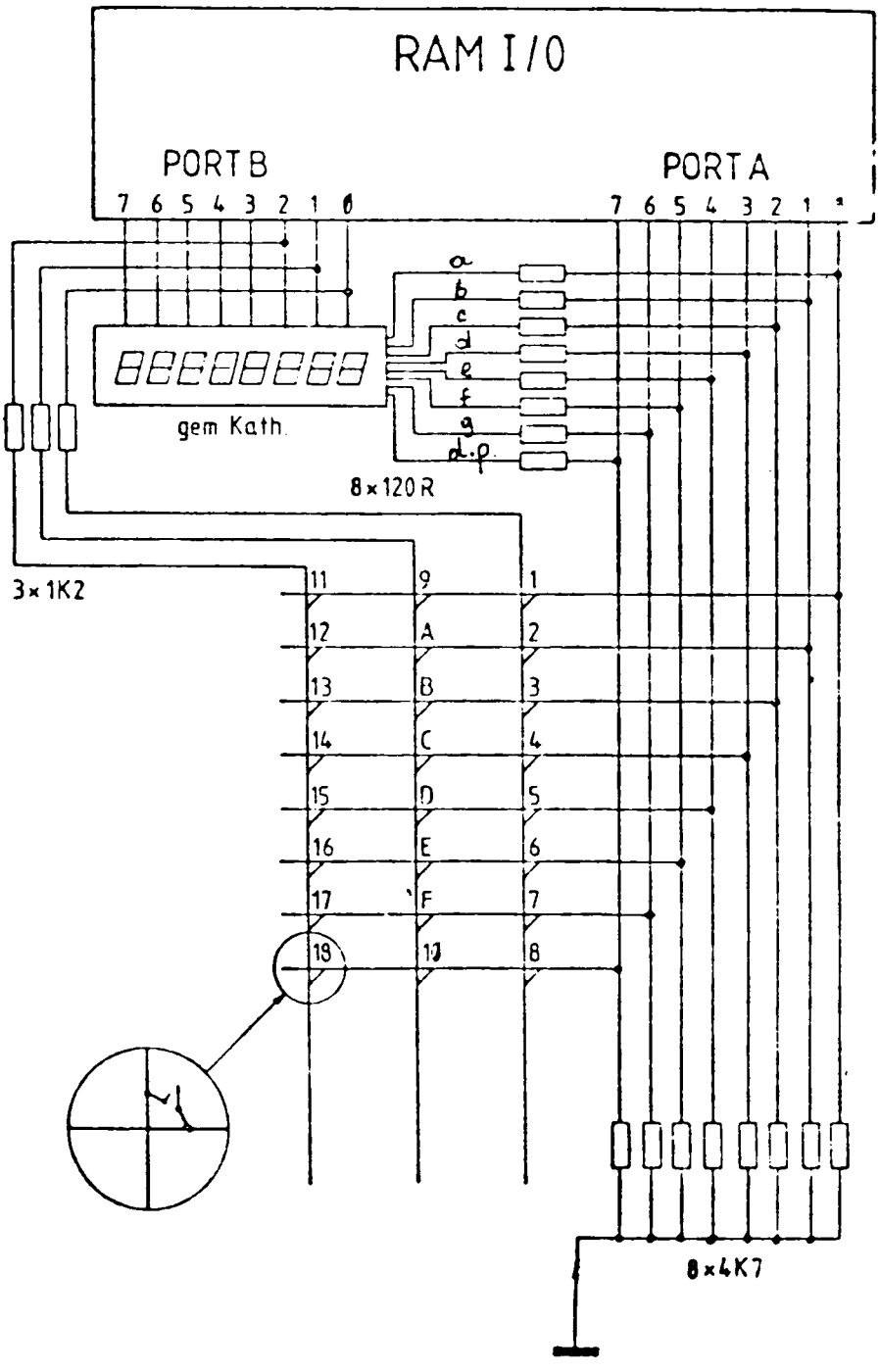
Displayroutine

```
1100  C400    LDI 00
1102  CA1B    ST 1B (2)       Modus-Flag 00
1104  9004    JMP $ 1
    $   0
1106  C4FF    LDI FF          Modus-Flag FF
1108  CA1B    ST 1B (2)
    $   1
110A  C400    LDI 00          Tastenspeicher
110C  CA10    ST 1C (2)       löschen
    $   2
110E  C492    LDI 92
1110  31      XPAL 1          Ptr 1 auf
1111  041F    LDI 1F          "Display-RAM"
1113  35      XPAH 1
1114  C400    LDI 00          MDR = 00
1116  CAA4    ST A4 (2)       (falls vorher
1118  C4FF    LDI FF          anderer Modus)
111A  CAA3    ST A3 (2)       Beide Ports =
111C  CAA2    ST A2 (2)       Ausgänge
111E  C4FE    LDI FE
1120  CAA1    ST A1 (2)       Port B initia-
1122  C409    LDI 09          lisieren
1124  CA00    ST 00 (2)       Zähler laden
    $   3
1126  C2A1    LD A1 (2)
1128  1E      RR              (Port B)
1129  CAA1    ST A1 (2)       rotieren
112B  BA00    DLD 00 (2)      Zähler - 1
112D  980D    JZ S 4          Über E 7-Segm-
112F  01      XAE             Code holen u.
1130  C180    LD 80 (1)       in Port A
1132  CAA0    ST A0 (2)       schreiben
1134  8F01    DLY 01          1 ms/Digit
1136  C400    LDI 00
1138  CAA0    ST A0 (2)       Port A = 00
113A  90EA    JMP $ 3
    $   4                     Tastensubroutine
113C  C400    LDI 00          Zwischen-
```

```
113E  CA1D    ST 1D (2)      speicher = 00
1140  CAA2    ST A2 (2)
1142  C404    LDI 04         Port A = Eing.
1144  CAA1    ST A1 (2)      Port B ini-
    $   5                    tialisieren
1146  C2A0    LD A0 (2)
1148  9C09    JNZ $ 6        Taste ge-
114A  C2A1    LD A1 (2)      drückt?
114C  9824    JZ $ 10        Port B fertig
114E  1C      SR             geshiftet?
114F  CAA1    ST A1 (2)      (Port B) SR
1151  90F3    JMP $ 5
    $   6
1153  01      XAE            (Port A) in E
    $   7
1154  AA1D    ILD 1D (2)     Tastenspei-
1156  40      LDE            cher + 1
1157  D401    ANI 01
1159  9C05    JNZ $ 8        (Port A) fer-
115B  40      LDE            tig rotiert?
115C  1E      RR
115D  01      XAE            (Port A) RR
115E  9CF4    JNZ $ 7
    $   8
1160  C408    LDI 08         Tastenwert
1162  01      XAE            errechnen.
    $   9
1163  02      CCL
1164  C2A1    LD A1 (2)
1166  1C      SR
1167  CAA!    ST A1 (2)
1169  9807    JZ $ 10
116B  C21D    LD 1D (2)
116D  70      ADE
116E  CA1D    ST 1D (2)
1170  90F1    JMP $ 9
    $   10
1172  C21B    LD 1B (2)
1174  9C10    JNZ $ 12
1176  C21C    LD 1C (2)
```

```
1178  9C06     JNZ S  11
117A  C21D     LD 1D (2)
117C  CA1C     ST 1C (2)
117E  908E     JMP S   2
    S  11
1180  C21D     LD 1D (2)
1182  9806     JZ S   13
1184  9088     JMP S  2
    S  12
1186  C21D     LD 1D (2)      Modus-Flag?
1188  CA1C     ST 1C (2)
    S  13
118A  3F       XPPC 3         Rücksprung
```

Das Programm ist relokierbar, d.h. es läuft ohne Änderung auf anderen Adressen.


RAM I/0
PORT B
7 6 5 4 3 2 1 0
PORT A
7 6 5 4 3 2 1 0
a
b
c
d
e
f
g
d.p
gem Kath.
8×120R
3×1K2
11
12
13
14
15
16
17
18
9
A
B
C
D
E
F
10
1
2
3
4
5
6
7
8
8×4K7

TINY BASIC für SC/MP - Systeme

Der hier beschriebene Interpreter belegt exakt 4K an Speicherplatz. Er eignet sich für alle SC/MP - Systeme, die außerdem mindestens 1K freies RAM (ab 0C00, z.B. ELEKTOR-System) besitzen. Weitere 4 - K - RAM-Karten sind erst für spätere Erweiterungen nötig.

Allgemeines

Auch dieser BASIC - Interpreter basiert, wie alle derzeit angebotenen SC/MP-BASIC-Versionen, auf NIBL. (Fehlerfreie Dex.-Version) NIBL ist die Abkürzung für National Industrial BASIC Language, eine Ende 1976 vom SC/MP - Hersteller National Semiconductor herausgebrachte TINY-BASIC-Version, die über den üblichen Befehlssatz hinaus zahlreiche und z. T. recht raffinierte Zusätze (z.B. indirect Operator, DO/UNTIL usw.) hat und sogar einfache Stringverarbeitung gestattet.

Es ist schon einige Programmierkunst nötig, um bei einem vergleichweise "langweiligen" Prozessor wie dem SC/MP (dessen Vorzüge unbestritten sind!...) derart viele Funktionen in 4K unterzubringen. So weicht dann auch NIBL "innerlich" stark von dem Aufbau anderer entsprechender Interpreter ab: Es wird sozusagen softwaremäßig ein "neuer Rechner" erzeugt, mit eigenen Registern, einem eigenen Programmzähler usw. Für deren Inhalte sind bestimmte RAM-Bytes reserviert. Dieser Rechner kennt nur 4 Befehlstypen: DO, TEST, CALL, JUMP; was das bedeutet, kann man sich ungefähr denken. Die eigentliche Interpretation übernimmt ein nur etwa 3/4 K langes "Programm", welches ausschließlich aus diesen Befehlen zusammengesetzt ist. Fast der ganze Rest besteht aus einzelnen Unterprogrammen, mit deren Hilfe diese "Befehle" realisiert werden. Man könnte also auch von einer zweifachen Interpretation sprechen.

Wie man sich leicht vorstellen kann, ist dieses BASIC daher extrem langsam (verglichen mit anderen Versionen): hierbei spielt natürlich auch die ohnehin geringe Arbeitsgeschwindigkeit des SC/MP eine Rolle. Zum Glück gibt es genug Anwendungen, in denen die Geschwindigkeit keine Rolle spielt.

So raffiniert manche der NIBL-Eigenschaften sind - einige sind für normale Hobbycomputer-Anwendungen unnötig oder sogar störend (z.B. das automatische Löschen des Programmspeichers nach dem Starten des Interpreters verunmöglicht bzw. erschwert das Cassetten-Laden von NIBL-Programmen). Das erklärt sich daraus, daß NIBL vorwiegend für industrielle Anwendungen (z.B. Maschinensteuerungen) gedacht ist - wie der Name schon sagt. Auch die Ein/Ausgabe-Routinen sind speziell für Teletype-Betrieb mit Lochstreifenzusatz ausgelegt.

Die vorliegende Version verzichtet auf einige der unliebsamen NIBL-Eigenschaften und nutzt den freiwerdenden Speicherplatz für nützlichere Dinge. Da das NIBL inzwischen allgemein bekannt ist und die zugehörigen Unterlagen kostenlos erhältlich sind (siehe Literaturhinweise), soll nun auf die Unterschiede zwischen beiden Versionen hingewiesen werden.

Veränderungen gegenüber NIBL

- NIBL soll aus obengenannten Gründen sofort nach dem Einschalten "da" sein, daher liegt der Interpreter auf Page 0 des SC/MP-Adressbereichs. Für die meisten Hobby-Systeme ist das ungünstig, da hier der System-Monitor liegt. Meist werden auch die Hardware-Komponenten (Tastatur, Display usw.) und eine minimale RAM-Ausstattung in Page 0 adressiert. Deshalb wurde dieses BASIC auf Page "C" gelegt: Dort stört es nicht und "schluckt" auch nichts vom Programmspeicher, der bis 7FFF reicht.

- Die Originalversion stellt dem Programmierer 7 "Seiten" je 4K als Programmspeicher zur Verfügung, nämlich den Adressbereich 1000 ...7FFF. Seite 1 ist allerdings verstümmelt, da ein Teil davon für Variablen, Stacks, den Zeilenpuffer usw. gebraucht wird. Dieses BASIC benutzt dafür den RAM-Bereich von 0C00...0D20, so daß hier Page 1 voll für Programme genutzt werden kann. Ferner kann der verbleibende RAM-Bereich von 0D20 bis 0FFF ebenfalls für Programme in BASIC genutzt werden: Im Gegensatz zur Originalversion ist hier also auch "PAGE 0" erlaubt. Man kann also seine ersten BASIC-Gehversuche bereits ohne teure Speichererweiterungen vornehmen.

- Wenn dieses BASIC gestartet wird (Startadresse ist C000), wird zunächst der Bildschirm des angeschlossenen (Elek-) Terminals gelöscht. Dann meldet sich der Interpreter mit folgender Überschrift:

```
SC/MP TINY BASIC V.3*MR 8/79
)
```

  Eventuell im Programmspeicher stehende Programme (z.B. solche, die gerade zuvor von der Cassette geladen wurden) bleiben - im Gegensatz zum NIBL - hierbei unversehrt. Es kann also auch jederzeit gefahrlos mittels Reset der Interpreter verlassen und nach Belieben neu gestartet werden. Zum Löschen einer Page (n) ist NEW n einzugeben; dies ist immer vor Eingabe eines neuen Programms erforderlich.

- Ein Leckerbissen für flinke Programmierer ist die in diesem BASIC geschaffene Möglichkeit, Buchstaben (von Variablen, Hexzahlen oder "Keywords") wahlweise als Groß- oder Kleinbuchstaben eingeben zu können, damit man nicht ständig einen Finger auf der SHIFT-Taste hat...
  Auf "Notlösungen" wie z.B. elektr. Verriegelung der SHIFT-Taste ("SHIFT-LOCK"), die auch ihre Nachteile hat, kann also verzichtet werden.

- In diesem BASIC wie auch in der Originalversion darf eine Programmzeile max. 72 Zeichen enthalten. Was das NIBL macht, wenn dieser Wert aus Versehen überschritten wird, ist ausgesprochen unpraktisch: Es übernimmt ungefragt die unvollständige Zeile, als ob man CARRIAGE RETURN gedrückt hätte! Wenn man es merkt, ist es zu spät: man muß die ganze Zeile nochmal eingeben! Dieses BASIC verhält sich vornehmer: Ist man am (unsichtbaren) Zeilenende angekommen, werden einfach keine Zeichen mehr angenommen. Die einzigen Tasten, die jetzt noch wirksam sind, sind Backspace und CR.

- Ein weiterer Unterschied (für Perfektionisten): Die selten gebrauchte "SHIFT-O" - Funktion (Back-Arrow) wurde hier, da sie nicht auf allen Tastaturen zu finden ist, durch "Substitute" (CTRL Z) ersetzt.

- Nach all diesen Änderungen und Erweiterungen war immer noch genügend Speicherplatz frei, um die BASIC-Fehlermeldungen etwas komfortabler zu gestalten. Folgende Auflistung zeigt die Unterschiede (*):

	MR-TINY BASIC	Original-NIBL	Bedeutung
	AREA ERROR	AREA ERROR	Kein Platz mehr in der jeweiligen Speicherseite
	CHAR ERROR	CHAR ERROR	Zeichen nach dem Ende eines Statements (z.B. PRINT 3+3 X)
*	DIV/0 ERROR	DIV0 ERROR	Division durch Null
	END" ERROR	END" ERROR	" am Stringende fehlt
	FOR ERROR	FOR ERROR	Gebrauch von FOR ohne NEXT
	NEST ERROR	NEST ERROR	Zu tiefe Schleifverschachtelung ("Nesting") bei FOR-NEXT, DO-UNTIL, GOSUB usw.
	NEXT ERROR	NEXT ERROR	Gebrauch von NEXT ohne FOR
	NOGO ERROR	NOGO ERROR	Sprung zu nicht existierender Zeilennummer
*	RETURN ERROR	RTRN ERROR	Gebrauch von RETURN ohne GOSUB
*	SYNTAX ERROR	SNTX ERROR	Syntax-Fehler
	STMT ERROR	STMT ERROR	Falscher Gebrauch eines Statements
*	UNTIL ERROR	UNTL ERROR	Gebrauch von UNTIL ohne DO
*	VALUE ERROR	VALU ERROR	Zahlenbereich überschritten

- Eine weitere Änderung gegenüber NIBL ist die Anpassung der Ein/Ausgabe-Routinen an Video-Terminals: die Übertragungsgeschwindigkeit wurde auf 1200 Baud erhöht (statt 110), und FLAG 1 wurde von der Aufgabe der "Reader-Relay-Steuerung" befreit und steht nun dem Programmierer zur Verfügung.

- Die letzte Änderung des NIBL betrifft den Ausdruck von Variablen: Diesen wird im NIBL ein Leerzeichen (oder Minuszeichen) vorangestellt, dann folgt die Variable, und danach noch ein Leerzeichen. Es gibt keine Möglichkeit, dies zu verhindern, obwohl es in manchen Programmen wünschenswert wäre. In diesem TINY BASIC wurden die obligatorischen Leerzeichen weggelassen. Wo sie wirklich gebraucht werden, sind sie leicht softwaremäßig zu realisieren.
  Beispiel: PRINT A, " "

Befehlssatz

Diese Aufstellung fasst alle Möglichkeiten dieses TINY BASIC in Kurzform zusammen. Das kann natürlich keinen Programmierkurs ersetzen; der Anfänger sei daher auf die entsprechende (NIBL-) Literatur verwiesen (vgl. Literaturverzeichnis).

- **Programmeingabe**

  Vor Eingabe eines neuen Programms ist NEW einzugeben; das folgende Programm "landet" dann in Page 1. Wird eine andere Page gewünscht, ist NEW n einzugeben (n = Page-Nr. 0...7).
  Eingaben ohne Zeilennummer werden sofort ausgeführt; Zeilen mit Nummern werden in aufsteigender Reihenfolge im Programmspeicher abgelegt. Eine bereits existierende Programmzeile wird bei Eingabe einer neuen Zeile mit gleicher Nummer ersetzt. Wird nur eine Zeilennummer (ohne folgende Programmzeile) eingegeben, wird die entsprechende Zeile aus dem BASIC-Text entfernt.
  Mit CTRL U ("NAK") vor Carriage Return kann eine falsch eingegebene Zeile verlassen werden, ohne daß diese in den Programmspeicheraufgenommen wird.
  Es sind positive Zeilennummern von 0 bis 32767 zulässig.

- **Konstanten und Variablen**

  Es sind 26 Variablen möglich, bezeichnet durch die Buchstaben A..Z. Sie können, ebenso wie die Konstanten, nur ganzzahlige Werte von -32768 bis +32767 annehmen. Konstanten können auch in Hexadezimahlform eingegeben werden, wenn das Zeichen # vorangestellt wird.

- Pseudovariablen

Es existieren zwei sog. Pseudovariablen, STAT und PAGE. STAT repräsentiert das SC/MP-Statusregister; PAGE enthält die Nummer der gerade benutzten Programm-Page. Anwendungsbeispiele:

```
STAT = STAT OR 2          (setzt Flag 1)
STAT = 2                  (setzt Flag 1)
STAT = STAT AND # FD      (löscht FLAG 1)
IF (STAT AND # 10) = 1 GOTO...  (Bedingte Verzweigung
                                 abhängig von Sense A)
PAGE = PAGE + 1           (zum Weitermachen auf der nächsten
                                 Seite)
PRINT PAGE                (Ermitteln der gegenwärtig benutzten
                                 Page)
```

- Programmkontrollkommandos

LIST	Listet das komplette Programm der jeweiligen Page auf; kann durch die BREAK-Taste unterbrochen werden.
LIST n	Listet das Programm ab Zeile n auf.
RUN	Startet das Programm, beginnend bei kleinster Zeilennr.
GOTO n	Startet das Programm, beginnend bei Zeile n.
CLEAR	Löscht alle Variablen und Stacks (Bei RUN geschieht dies, im Gegensatz zum Original-NIBL, nicht!)
NEW n	Löscht Programm, aber nicht die Variablen, in Page n und setzt den Wert von Page auf n. Wenn n fehlt, ist Page 1 betroffen. (Nach dem Starten des Interpreters ist PAGE ebenfalls auf 1 gesetzt).
BREAK	(-Taste) unterbricht das laufende Programm.
END	unterbricht das laufende Programm.

- Sonstige Statements

LET	Zuweisung eines Wertes zu einer Variablen, z.B. LET A=5; LET PAGE=4. "LET" kann auch fortgelassen werden.

IF/THEN Bedingte Ausführung eines Programmschritts. "THEN" kann auch fortgelassen werden.

GOTO n Sprung zur Programmzeile n

GOSUB n Subroutinenaufruf Zeile n

RETURN Subroutinenrückkehr

DO/UNTIL Bedingte Schleifenwiederholung.

Beispiel:
```
10 DO
20 A=A+1 : PRINT A, " ";
30 UNTIL A= 10
)RUN
1 2 3 4 5 6 7 8 9 10
```

FOR/NEXT Schleifenwiederholung mit verschiedenen Parametern.

Beispiel:
```
10 FOR I=1 TO 11 STEP 2
20 PRINT I, " ";
30 NEXT I
)RUN
1 3 5 7 9 11
```

INPUT n Zuweisung des eingegebenen Wertes zur Variablen n.

LINK n Aufruf einer Maschinen - Subroutine bei Adresse n.

REM Für Kommentare (wird bei Programmausführung ignoriert)

PRINT n Ausgabe von n. n kann eine Variable, eine Zahl, ein String oder ein Text in Anführungszeichen sein; auch mehrfach hintereinander, durch Kommas getrennt. Ein Semikolon am Schluß kann den Zeilenvorschub unterdrücken.

$ n Bezeichnung eines Strings, der bei Adresse n beginnt und mit CR endet. Anwendungsbeispiele:

```
PRINT $ n
INPUT $ n
$ n =  $ m
```

- Funktionen

RND(X,Y) liefert eine Zufallszahl zwischen X und Y. Y muß dabei größer als X sein, und die Differenz zwischen beiden darf 32767 nicht übersteigen.

MOD(X,Y) liefert den Divisionsrest von X/Y

TOP liefert die Adresse (dezimal) des ersten freien RAM-Bytes nach dem BASIC-Text in der jeweiligen Page.

- **Arithmetische Operatoren**

X+Y	Addition
X*Y	Multiplikation
X-Y	Subtraktion
X/Y	Division (Stellen hinter dem Komma im Ergebnis werden verschluckt)

- **Verhältnisoperatoren**

=	gleich
<=	kleiner gleich
>=	größer gleich
<>	ungleich
<	kleiner als
>	größer als

- **Logische Operatoren**

AND	log. UND
OR	log. ODER
NOT	log. Invertierung

- **Indirekter Operator**

@ n bezeichnet den Inhalt des Bytes an der Adresse n. Anwendungsbeispiele:

```
A=@ n    (entspricht PEEK)
@ n = A  (entspricht POKE)
```

Anstelle von A könnte auch eine Zahl zwischen 0 und 255 stehen.

- **Sonstiges**

Mehrere Statements in einer Zeile werden durch Doppelpunkt getrennt. Nach einem IF/THEN - Statement hängen dann alle folgenden Schritte in der gleichen Zeile von der Bedingung ab.

Zwischenräume können nach Belieben eingefügt werden, um die Übersichtlichkeit zu erhöhen, sofern dadurch keine Kommandoworte aufgetrennt werden. An den meisten Stellen können sie auch ganz fortgelassen werden.

Die Rangordnung von arithmetischen und logischen Operationen hängt von den entsprechenden Gesetzen ab (z.B. Punkt- vor Strichrechnung usw.), kann aber durch Setzen von Klammern beliebig verändert werden. Im Zweifelsfall lieber mehr Klammern verwenden.

Hardware

Wie bereits erwähnt, wurde beim Aufbau dieses Interpreters größter Wert auf minimalsten Hardware-Aufwand gelegt. Bereits das SC/MP-System nach ELEKTOR in der Ausbaustufe November 1977 beispielsweise ist vollauf geeignet. Ein beliebiges Video-Terminal (z.B. ELEKTERMINAL) mit 1200 Baud kann direkt an FLAG 0 (Eingang) und SENSE B (Ausgang) angeschlossen werden, sofern es über TTL-kompatible Ein- und Ausgänge verfügt. Der Eingang muß ferner invertierend sein, was z.B. bei Verwendung des RS 232 - Interfaces nach ELEKTOR 101 sichergestellt ist. Es wird also die gleiche Anschlusskonfiguration verwendet wie auch in allen National-Applikationen; zusätzliche Interface-Platinen wie bei einigen anderen NIBL-Versionen sind also nicht nötig. 4-K-RAM-Karten können nach Belieben hinzugefügt werden.

Cassetten-Interface

Zum Abspeichern von BASIC-Programmen sollte das System über ein Cassetten-Interface verfügen. Anfangs- und Endadresse des Programms werden wie folgt ermittelt:

Anfangsadresse =	Startadresse der betreffenden Page (außer bei Page 0; dort ist die Startadresse 0D1F).
Endadresse =	TOP-1; sie kann also durch den Befehl PRINT TOP-1 ermittelt werden. Sie muß allerdings noch in HEX umgewandelt werden. Für Besitzer des WILLY-Programms geht das per Tastendruck; ansonsten leistet ein kleines BASIC-Programm nützliche Dienste.

Für Besitzer des ELEKTOR-Systems ist zu beachten, daß der RAM-Bereich ab etwa 0FC9 für das ELBUG-Monitorprogramm (also auch die Cassetten-Routinen) gebraucht wird. Daher können Programme in Page 0 nur abgespeichert werden, wenn der TOP-Wert kleiner als 4042 ist. (Für nicht abzuspeichernde Programme kann aber der RAM-Bereich bis 0FFF voll genutzt werden!).
Ein von der Cassette geladenes Programm kann nach dem Starten des Interpreters und "RUN" sofort gestartet werden, wenn es auf Page 1 steht; andernfalls ist zuvor die Pseudovariable PAGE auf den entsprechenden Wert zu setzen.

Anmerkung

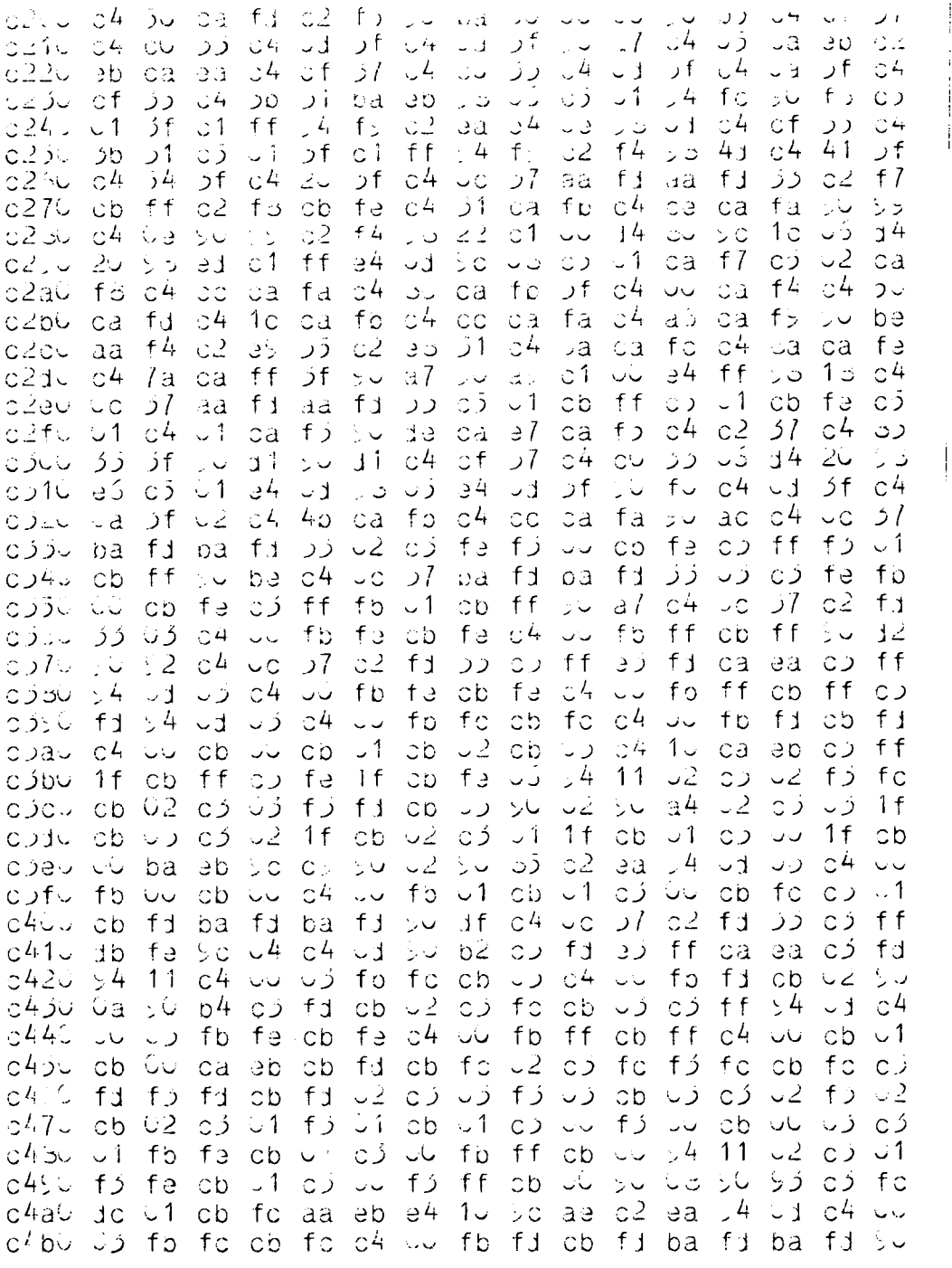
Eine Cassette mit der Aufzeichnung dieses Interpreters im ELEKTOR-Format (Cansas City) ist beim Autor für DM 89,-- erhältlich, ebenso auf Wunsch entsprechend programmierte EPROMs (5204, 2708, 2716 usw.).

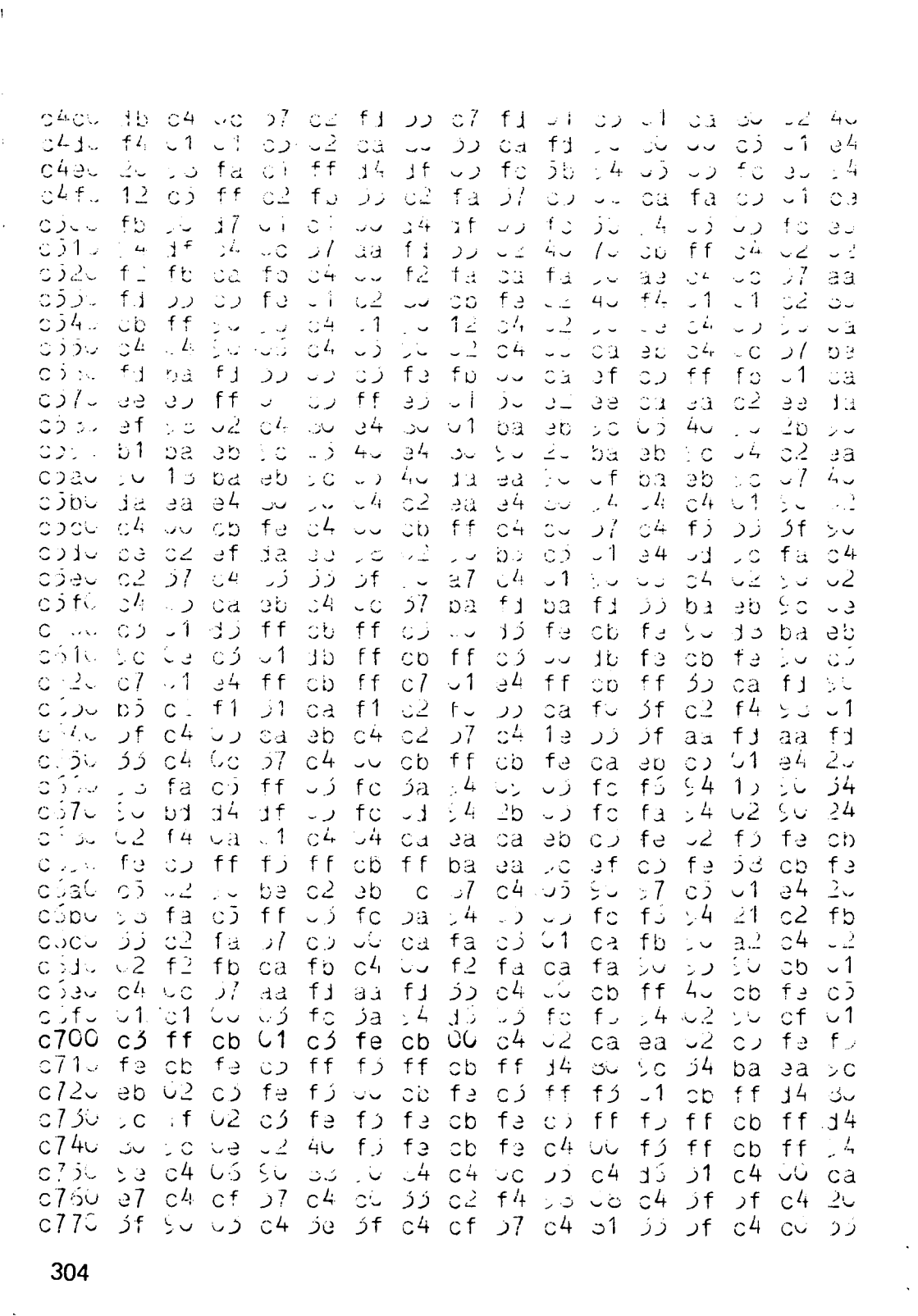
ing. (grad.) manfred reimer elektronik / hard- u. softwareentwicklung
haubachstr. 8, 1000 berlin 10, tel.: 030/34 21 42 8

mem-dump

title: TINY - BASIC für SC/MP

```
c000 c4 cf 37 c4 ec 33 c4 0c 35 c4 1c 32 c4 c0 33 c4
c010 1d 31 c4 01 ca f5 3f c5 01 3c fb 30 21 33 43 2f
c020 4d 30 2c 34 49 4e 39 20 42 41 33 43 43 2c 35 2e
c030 33 20 2c 2a 2c 4d 32 2c 36 2f 37 39 ca c0 ca f4
c040 ca f3 c4 1c ca f0 c4 cc ca fa c4 c0 ca ea 01 c4
c050 c0 ca 30 aa ea 01 c4 34 c0 3c f4 c4 30 ca f1 c4
c060 7a ca ff c4 ca ca fc c4 a3 ca f3 c4 da ca fe c2
c070 fb 33 c2 fa 37 c7 01 01 c7 01 33 ca fb 40 14 0f
c080 dc c0 37 ca fa 40 d4 f0 e4 2c 38 2f e4 a0 36 07
c090 e4 c0 36 e1 3f 30 16 c2 f3 e4 d6 3c c4 c4 ca 30
c0a0 37 e4 d5 33 ca ea c4 0c 37 01 c2 f0 cf 01 c2 fa
c0b0 cf 01 c2 ea 33 ca f3 40 37 30 ba ca e7 c5 01 e4
c0c0 2c 36 fa c5 ff c2 fa 37 ca ec c2 fb 33 ca e1 c7
c0d0 01 01 ba e7 c1 c0 14 40 1c e4 ff d3 01 30 14 7f
c0e0 3c 07 40 34 ea 30 ce 30 85 c2 e7 01 c5 c0 c2 e1
c0f0 33 c2 ec 37 3c c5 c4 0c 37 c2 f3 33 c7 ff 01 c7
c100 ff 33 ca f3 40 37 30 b1 30 41 c2 fc e4 7a 36 1c
c110 aa fc aa fc 33 c4 0c 37 c2 f4 36 0a 33 cb ff 33
c120 31 cb fe 31 30 c1 c4 ff cb ff 30 bb c4 0a 30 1b
c130 c5 01 e4 2c 36 fa e4 21 30 04 e4 37 3c 01 3f c4
c140 04 30 06 c2 fc e4 6a 3c 04 c4 03 30 40 ba fc ba
c150 fc 33 c4 0c 37 c5 01 34 06 c4 00 ca f4 30 88 33
c160 c5 00 31 c4 01 ca f4 30 f4 c2 f2 34 04 c4 06 30
c170 1c c4 01 ca f4 3f c4 cf 37 c4 c0 33 c5 01 e4 22
c180 36 db e4 2f 36 05 e4 01 3f 30 eb c4 07 30 33 c4
c190 0c 37 aa f1 aa f1 33 c4 0a cb fe c4 00 cb ff c4
c1a0 03 ca e7 c4 ff cb 05 c5 f1 34 15 c4 21 cb 04 c4
c1b0 00 03 fb fc cb fc c4 00 fb f1 cb f1 30 3f c4 00
c1c0 cb 04 30 13 30 37 aa f1 aa f1 31 c4 0c 33 aa e7
c1d0 01 c1 01 dc 30 c3 00 c1 f1 d3 fc 38 0a c4 cf ca
c1e0 fa c4 33 ca fb 30 db c4 cf 37 c4 c0 33 c2 f5 3c
c1f0 03 c1 04 3f c2 e7 01 c5 30 c1 00 3f c5 ff 34 fb
```





```
c7e0  4c  e3  f3  e4  ca  e8  ef  4c  e4  cd  e8  dc  4c  e4  1a  e8
c7f0  41  4c  e4  c3  e8  d3  4c  e4  1d  e3  6f  4c  e4  c3  e8  1a
c800  c4  de  3f  c4  4d  3f  c4  ce  9c  a3  c4  de  3f  c4  33  3f
c810  c4  cd  3f  c4  ca  3f  ce  3f  9c  9b  40  cd  01  aa  e7  e4
c820  4c  e8  c3  c4  ce  3f  9c  c4  ce  12  e8  e7  c4  2c  3f  c4
c830  ce  3f  32  e7  e3  ac  ba  e7  c3  ff  ce  ca  4c  cd  01  c4
c840  ca  3f  c4  ce  c3  c4  dc  31  ce  ce  c4  ce  37  c2  f3  33
c850  c3  ff  33  01  c3  fe  31  ca  ef  31  ce  cb  fe  c4  ce  cb
c860  ff  c2  ef  31  4c  33  9c  bc  c4  ce  37  c2  f3  33  c7  fe
c870  01  c7  ff  ca  ea  c7  ff  33  ca  f3  c2  ea  37  4c  cb  ce
c880  9c  c6  9c  ac  c4  ce  37  c2  f3  33  c3  01  ca  f7  c3  ce
c890  ca  f8  c2  f1  33  c2  fc  37  c4  04  ca  e7  c7  01  e4  cd
c8a0  98  04  aa  e7  9c  fc  c2  e7  e4  04  9c  02  ca  e7  c2  e7
c8b0  01  c2  f2  94  c6  d4  7f  ca  f2  9c  18  c3  03  40  02  f4
c8c0  fc  01  c3  01  e4  cd  98  cb  4c  02  f4  ff  01  9c  f3  9c
c8d0  af  9c  af  4c  da  e7  98  f7  c4  7a  ca  ff  c4  6a  ca  fc
c8e0  c4  3a  ca  fe  4c  98  6c  94  1c  c1  cc  c9  8c  c3  01  94
c8f0  f8  c1  cc  94  f4  c9  8c  9c  4e  c1  fe  ca  ea  c4  ff  c9
c900  e7  cf  01  c3  01  cf  01  e4  cd  9c  f8  9c  dc  c4  c0  37
c910  c4  3e  33  3f  9c  cf  ba  fd  ba  fd  33  c4  0c  37  c3  cc
c920  ca  ef  c3  01  ca  ee  9c  e3  c2  ff  01  40  e4  7a  9c  04
c930  c4  0f  9c  e0  c2  ef  da  ee  98  06  ba  ff  ba  ff  9c  cd
c940  40  33  c4  0c  37  c3  ff  33  c3  fe  31  9c  c0  c2  ef  d4
c950  f7  07  9c  b9  9c  be  c4  0c  37  aa  fd  aa  fd  33  06  cb
c960  fe  c4  cc  cb  ff  9c  eb  c2  ee  37  c2  ef  33  c7  ff  3f
c970  c4  0c  36  c4  1c  32  9c  da  c2  ff  e4  3a  9c  04  c4  0a
c980  9c  d2  aa  ff  aa  ff  33  c4  0c  37  33  cb  ff  33  31  cb
c990  fe  31  9c  be  c2  e9  37  c2  e8  33  c3  cc  94  02  9c  07
c9a0  c3  02  01  c7  e0  9c  f3  c7  02  aa  fd  aa  fd  33  01  c4
c9b0  0c  37  cb  ff  40  cb  fe  9c  d9  c3  01  e4  cd  9c  fa  3f
c9c0  c2  fd  33  c4  0c  37  c3  03  cb  fe  c3  02  cb  ff  9c  c2
c9d0  9c  ae  c4  08  ca  eb  c2  e3  01  c2  e4  ca  e9  c2  e3  02
c9e0  70  01  c2  e4  02  f2  e9  ca  e4  ba  eb  9c  f0  40  02  f4
c9f0  07  01  c2  e4  02  f4  07  1e  ca  e4  aa  e6  9e  03  40  ca
ca00  e3  c2  fd  33  c4  0c  37  c4  01  cb  00  c4  cc  cb  01  c3
ca10  fe  cb  02  c3  ff  cb  03  c3  fc  cb  04  c3  fd  cb  03  c2
ca20  e4  cb  fe  c2  e3  e4  ff  d4  7f  cb  ff  c7  06  33  ca  fd
ca30  9c  9c  9c  9c  aa  fd  aa  fd  33  c4  0c  37  c4  00  cb  ff
ca40  c4  01  cb  fe  9c  ea  c2  fe  e4  a6  9c  04  c4  0a  9c  e2
```

```
cab0 e4 a6 51 ca f1 c4 cc 55 ca fc c2 fa 55 c4 cc 57
cab0 c5 f5 ca c1 c5 fc ca c1 c5 f1 ca c1 c5 fe ca c1
ca70 c5 ff ca c1 c2 f1 ca c1 c2 fc ca c1 55 c2 f1 51
ca80 ca fe c7 fc 55 ca f5 56 a7 c2 fe e4 aa 5c c4 c4
ca90 cb 56 bb e4 ca 51 ca f1 c4 cc 55 ca fc c2 fa 55
caa0 c4 cc 57 c7 ff e1 f5 5e c4 c4 cc 56 a1 e1 f5 c1
cab0 c2 a6 c2 f1 fc ca a6 cb c6 c5 c1 c2 a6 f1 fa ca
cac0 a6 cb c1 c5 ff c1 fa cb c2 c1 fb cb c5 c1 fa 54
cad0 10 c4 c4 ca eb c7 c1 e4 ff cb ff ba eb 5c f5 56
cae0 c2 c7 c4 55 ca fa c2 f1 51 c2 fc 55 56 99 c2 ef
caf0 5a c8 c2 fe c2 f4 f5 ca fe 5f c2 fe 55 c4 cc 57
cb00 c5 ff 55 c5 fe 51 5c e4 56 a1 c2 ee 55 c2 ef 51
cb10 c4 cf 57 c4 cc 55 c5 c1 e4 cd 55 56 e4 cd 5f c6
cb20 d4 26 9c f2 56 c5 c2 ee 57 c2 ef 55 c5 c1 cf c1
cb30 e4 cd 5c fa 56 b5 c2 ef 55 c2 ee 57 c5 c1 e4 22
cb40 5a ca e4 2f 5c c4 c4 c7 56 be e4 cd cf c1 56 ec
cb50 c4 cd cb c6 56 de c2 fa 55 c4 cc 57 c7 ff 55 c7
cb60 ff 51 c7 ff c1 c7 ff 55 ca fa 40 57 c5 c1 cf c1
cb70 e4 cd 5a c6 c6 d4 26 9c f5 56 b5 aa fa aa fa 55
cb80 c4 cc 57 c2 f5 cb fe c4 c6 cb ff 56 a7 c2 ef d4
cb90 c7 ca f5 5f c6 c6 c6 c6 c6 c6 c6 c6 c2 f5 5c c9
cba0 c4 cd ca e9 c4 26 ca e8 5f c1 c4 c4 ca eb 46 c2
cbb0 76 c1 ba eb 5c fa 46 ca e9 c4 c2 ca e8 5f c2 e9
cbc0 55 c2 e8 51 5f 55 c1 40 55 46 1c 1c 1c 1c ca f5
cbd0 5f c2 e9 55 c2 e8 51 c4 cd c9 ff c4 ff c9 c6 c9
cbe0 c1 5f c2 e9 55 c2 e8 51 c1 c6 e4 ff 54 12 c5 c1
cbf0 c1 fa ef c1 c6 fa ee 54 c7 c1 c2 c1 cb 86 56 e8
cc00 51 ca f5 51 55 ca f2 55 c2 ef e1 c1 5c c7 c2 ee
cc10 e1 c6 5c c1 5f c2 f2 dc 86 ca f2 5f c2 c6 c7 56
cc20 2c 25 8d 4c 1e c6 ab cc 55 cb 9b c6 30 c9 15 cb
cc30 e1 c8 25 4c 1e 2c 51 4c 49 55 d4 cb 9b c6 ab cc
cc40 47 c9 15 cb e1 4c 49 cb bd c2 d8 8f 2f c5 c5 4c
cc50 1c 2c 60 52 55 ce c1 2f cb 9b cb bd c2 bf c2 85
cc60 2c 8d 45 4c 45 41 d2 c1 2f c6 49 c2 85 2c 86 4e
cc70 45 d7 c6 ab cc 78 4c 7a 0a 55 c1 2f c9 15 cb 8c
cc80 cb 9b cb d6 c2 85 2c 8b 4c 45 d4 c4 dc cc 9a 2e
cc90 2f bd 8e 55 c4 c6 c1 2f c2 85 2c aa c6 8e ac 2e
cca0 2f bd 8e 55 c8 c7 c1 2f c2 85 2c bc 49 c6 8e 55
ccb0 2c b6 54 48 45 ce c9 15 c5 d6 4c 86 2c d1 55 4e
ccc0 54 49 cc c6 5b 8e 55 c1 2f c9 15 c9 27 cb c4 c2
ccd0 85 2c dd 44 cf c6 5b c1 2f c9 77 c2 85 2d c6 47
cce0 cf 2c eb 54 cf 8e 55 c1 2f 4c f5 2e 2f 55 5c c2
ccf0 8e 55 c1 2f c1 c9 cb 9b c9 15 cb e1 c1 c8 c2 85
```

```
cd00 2d 10 52 45 54 55 52 ce 01 2f 01 42 0b c4 02 85
cd10 2d 2a 4e 45 53 d4 06 5b 04 dc ce 2f 01 2f 0a 88
cd20 8e 5f 09 15 0a ed 0b c4 02 85 2d 54 46 4f d2 06
cd30 5b 04 dc ce 2f 2e 2f bd 8e 55 2e 2f 54 cf 8e 55
cd40 2d 4a 55 54 45 d0 8e 35 4d 4c 0a 55 01 2f 0a 45
cd50 04 c0 02 83 2d 67 53 54 41 d4 2e 2f bd 8e 35 09
cd60 15 09 4c 01 2f 02 85 2d 7e 50 41 47 c5 2e 2f bd
cd70 8e 55 01 2f 09 15 0b 3c 0b 9b 0b bd 02 85 2d 9e
cd80 a4 8e ac 2e 2f bd 2d 8f a2 09 15 0b 55 4d 9a 2e
cd90 2f a4 8e ac 06 50 0b 55 06 50 01 2f 02 83 2d d1
cda0 50 d2 2d a7 49 4e d4 2d ae a2 01 75 4d c1 2d bd
cdb0 a4 8e ac 06 50 09 15 0b 09 06 50 4d c1 8e 55 8f
cdc0 2f 2d c6 ac 4d a7 2d cb bb 4d cd 02 0c 01 2f 02
cdd0 85 2e 09 49 4e 50 55 d4 06 5b 04 dc cd f6 06 50
cde0 07 56 8e 55 04 c0 06 50 2e 05 ac 04 dc ce 2f 06
cdf0 50 2e 2f ac 4d e2 2e 2f a4 8e ac 06 50 07 56 09
ce00 15 0b 25 06 50 01 2f 02 85 2e 12 45 4e c4 01 2f
ce10 02 7f 2e 26 4c 49 4e cb 8e 55 01 2f 06 50 09 15
ce20 09 66 06 50 02 85 2e 2f 52 45 cd 09 b8 02 85 02
ce30 1a 8f 2f 02 a9 8e 61 2e 3e bd 8e 61 05 43 2e 53
ce40 bc 2e 48 bd 8e 61 05 4f 2e 4f be 8e 61 05 47 8e
ce50 61 05 4b 2e 8b be 2e 5d bd 8e 61 05 57 8e 61 05
ce60 53 2e 6a ad 8e 8d 05 5a 4e 6f 2e 6d ab 8e 8d 2e
ce70 78 ab 0e 8d 05 2c 4e 6f 2e 81 ad 8e 8d 05 45 4e
ce80 6f 2e 8b 4f d2 8e 8d 05 eb 4e 6f 00 f5 8e ac 2e
ce90 98 aa 8e ac 05 71 4e 8f 2e a1 af 8e ac 04 07 4e
cea0 8f 2e 8b 41 4e c4 8e ac 05 e7 4e 8f 04 dc ce b4
ceb0 05 2b 00 f5 06 ab ce ba 00 f5 2e c1 a5 06 4b 00
cec0 f5 2e cb a8 8e 55 2e 2f a9 00 f5 2e d4 c0 8e ac
ced0 07 e9 00 f5 2e df 4e 4f d4 8e ac 05 ef 00 f5 2e
cee0 e9 53 54 41 d4 09 55 00 f5 2e f4 54 4f d0 0b 9b
cef0 09 93 00 f5 2f 01 4d 4f c4 8f 20 04 07 09 bf 00
cf00 f5 2f 15 52 4e c4 8f 20 09 d1 05 45 03 2c 04 07
cf10 09 bf 03 2c 00 f5 2e 2f 50 41 47 c5 0b 7a 00 f5
cf20 2e 2f a8 8e 55 2e 2f ac 8e 55 2e 2f a9 00 f5 06
cf30 50 01 8e 04 07 01 c5 06 50 00 f5 20 45 52 52 4f
cf40 52 a0 41 52 45 c1 55 54 4d d4 43 48 41 d2 53 59
cf50 4e 54 41 d8 56 41 4c 55 c5 45 4e 44 a2 4e 4f 47
cf60 cf 52 45 54 55 52 ce 4e 45 53 d4 4e 45 58 d4 46
cf70 4f d2 44 49 56 2f b0 42 52 45 41 4b a0 55 4e 54
cf80 49 cc c4 08 ca eb 06 d4 20 9c fb c4 52 8f 00 06
cf90 d4 20 9c f2 06 dc 01 07 c4 85 8f 00 06 d4 20 98
cfa0 02 c4 01 ca ea 1f 01 1d 01 05 dc 01 e2 ea 07 ba
cfb0 eb 9c e5 06 d4 fe 07 8f 01 40 d4 7f 01 40 5f 90
```

```
cfc0 c1 01 c4 ff 8f 01 06 dc 01 07 c4 09 ca e8 c4 82
cfd0 8f 00 ba e8 98 10 40 d4 01 ca e9 01 1c 01 05 dc
cfe0 01 e2 e9 07 90 e8 05 d4 fe 07 5f 90 d4 c4 cf 37
cff0 c4 c0 33 c4 0c 5f 8f c0 c4 c0 37 c4 16 33 90 ea
```

```
zeichenerklaerung fuer telex-listing
====================================

das sonderzeichen ä gibt dem darauffolgenden zeichen eine
besondere bedeutung (ascii-sonderzeichen, die im telex-alphabet
nicht enthalten sind).

es bedeuten:

äe = ausrufungszeichen
äh = hashmark
äs = dollar
äp = prozent
ä+ = 'und' - zeichen
äm = multiplikation (stern)
ä, = semikolon
äk = kleiner als
äg = groesser als
äa = 'klammeraffe' (at)




 1 rem            + + + + + + + + + + +
 2 rem            + divisionsprogramm +
 3 rem            +  (c)  m. reimer   +
 4 rem            + + + + + + + + + + +
 5 rem
 6 rem
 7 pr'dieses programm ermoeglicht die division zweier ganzer'
 8 pr'zahlen mit beliebiger genauigkeit (max. 32767 stellen'
 9 pr'hinter dem komma.) eine grenze ist nur durch den be-'
10 pr'grenzten 'zahlenhorizont' von tiny basic (-32767 bis'
11 pr'+32767) gegeben. wenn dieser bei einem internen'
12 pr'zwischenergebnis ueberschritten werden sollte, wird'
13 pr'die rechnung abgebrochen.'
14 pr'': pr'':pr'zum start bitte die 'cr'-taste drueckenäe'
15 input äs6
16 pr'':pr'':pr'dividend'ä, : input d
17 pr'divisor'ä,  : input r
18 if r=6 pr 'fehler: div/6äe' : goto 17
19 pr'vieviel stellen hinter dem komma'ä, : input g
20 pr 'das ergebnis: ',d/r,','ä,
21 d6
22 d=mod (d,r) : if däk3277 goto 26
23 pr'':pr'tut mir leid, weiter kann ich nicht. mit dieser'
24 pr'aufgabe bin ich ueberfordert. vielleicht eine andere?'
25 until a=a : goto 16 :rem  'dummy-until' wegen do-stack
26 d=dämi6
27 pr d/rä,
28 g=g-1
29 until (g=6) or (d=6)
30 goto 16
```

```
  2 rem      + + + + + + + + + + + + + + + + + + + + +
  3 rem      + berechnung des geburts-wochentages  +
  4 rem      +        (c)  manfred reimer          +
  5 rem      + + + + + + + + + + + + + + + + + + + + +
  6 rem
  7 rem
 10 pr'weisst du eigentlich, an welchem wochentag du geboren'
 15 pr'wurdest?'
 20 pr'':pr'der computer rechnet es dir aus§e':pr''
 25 pr'dazu musst du dein geburts-'
 30 print 'datum in der reihenfolge: tag, monat, jahr eingeben,'
 40 print 'jeweils durch kommas getrennt. (z.b. 23,5,55)'
 50 print 'am ende der eingabe ist die 'cr'-taste zu druecken.'
 60 print ''
 70 print 'bitte gib nun dein datum ein§e'
 80 input t,m,j
 90 if (t§k1)or(t§g31)or(m§k1)or(m§g12) goto 140
100 if ((m=4) or (m=6) or (m=9) or (m=11)) and (t§g30) go to 140
110 if (j§k100) j=j+1900
120 if (j§k1855) or (j§g1984) go to 135
125 q=0: if mod (j,4)=0 q=1:  rem schaltjahr q=1
130 if (m=2) and (t§g(28+q)) go to 140
132 go to 150
135 print 'die jahreszahl muss zwischen 1855 und 1984 liegen§e'
140 print 'fehlerhafte eingabe§e - nochmal§e': go to 60
150 s=(1984-j)/4:rem  s=schaltj. zwischen geburt u. referenz-
160 go to 160 + m§m10:rem                        jahr (1984)
170 d=0:  go to 285: rem  bestimmung von d (=anzahl der tage
180 d=31: go to 285: rem  vom beginn des geburtsjahres bis
190 d=60: go to 285: rem  zum geburtsdatum)
200 d=91: go to 285
210 d=121: go to 285
220 d=152: go to 285
230 d=182: go to 285
240 d=213: go to 285
250 d=244: go to 285
260 d=274: go to 285
270 d=305: go to 285
280 d=335
285 d=d+t
290 if (q=0) and (d§g60) d=d-1: rem   geburt im schj. nach 28.2.?
300 z=((1984-j)§m365)-d+s: rem z=gesamtzahl der tage bis 1984
320 w=mod(z,7):              rem w=wochentag
330 if w=6 pr 'gratulation§e du bist ein sonntagskind§e': goto 410
340 print 'du wurdest geboren an einem '§,: go to 350+10§mw
350 print 'samstag§e': goto 410
360 print 'freitag§e': goto 410
370 print 'donnerstag§e': goto 410
380 print 'mittwoch§e': goto 410
390 print 'dienstag§e': goto 410
400 print 'montag§e'
410 print ''
420 print 'nochmal? wenn ja, 1 eingeben, sonst 0.'
430 input q
440 if q=1 goto 60
445 if q#k§0 pr'wie bitte'§,:goto 430
450 print ''
460 print 'es war mir ein vergnuegen. auf wiedersehen.
```

```
 1 rem         + + + + + + + + + + + + + + + + + + + + + + + +
 2 rem         +  berechnung des geburtstags-wochentages     +
 3 rem         +                (kurzversion)                +
 4 rem         +          (c)  manfred reimer                +
 5 rem         + + + + + + + + + + + + + + + + + + + + + + + +
 6 rem
 7 rem
 8 rem  hier wird anhand eines aehnlichen programms demonstriert,
 9 rem  was an speicherplatz gespart werden kann, wenn mehrere
10 rem  statements in eine zeile gepackt werden und statt der
11 rem  basic-'lookup-table' (im vorigen programm die zeilen-
12 rem  nummern 170 ... 280) eine solche aus einfachen bytes
13 rem  im ram verwendet wird. zusaetzliche ersparnis bringt
14 rem  hier der verzicht auf eine ueberpruefung der eingege-
15 rem  benen daten (voriges programm zeilen 90 ... 150).
16 rem
17 rem  wichtig:
18 rem
19 rem  folgende tabelle muss vor programmstart noch im ram
20 rem  beginnend bei adresse 1ff4 abgelegt werden:
21 rem  00, 1f, 3c, 3b, 79, 98, be, d3, f4, 12, 31, 4f
22 rem
23 rem
24 rem
25pr'geburtsdatum nach folgendem muster eingeben: 23,3,33'
26input,m,j:r=317 :d=t+da(r+m)+256dm(mdg9)
27w=mod.(365dm(84-j)+(mod(j,4)dkdj0)dm(jdj00)+(84-j)/4-d),7)
28ifw=0pr'gratulationde du bist ein sonntagskindde':end
29pr'du wurdest geboren an einem 'd,:gosub30+w:end
30pr'samstagde':return
31pr'freitagde':return
32pr'donnerstagde':return
33pr'mittwochde':return
34pr'dienstagde :return
35pr'montagde':return
```

```
 1 pr'         + + + + + + + + + + + + + + + + + +'
 2 pr'         + x/y - funktionsdarstellung  +'
 3 pr'         +    (c)   manfred reimer     +'
 4 pr'         + + + + + + + + + + + + + + + + + +'
 5 pr''
 6 pr''
 7 rem  9 verschiedene funktionsgleichungen (nr. 1...9)
 8 rem  koennen in den entsprechenden zeilen 100...
 9 rem  900 abgelegt werden. das programm ermittelt selbst
10 rem  den darstellungsmasstab, bei dem die schirmflaeche
11 rem  optimal ausgenutzt wird. es koennen mehrere funk-
12 rem  tionen uebereinander gezeichnet werden, wenn nach
13 rem  'run' - auch wiederholt - die nr. der entsprechenden
14 rem  funktion eingegeben wird. durch vorherige eingabe
15 rem  von '0' (nach dem erststart obligatorischde) wird
16 rem  alles geloescht und (mittels maschinen-unterprogramm)
17 rem  ein koordinatensystem generiert.
18 rem  wenn in der funktionsgleichung ein konstanter wert
19 rem  vorkommt (z.b. y=xdmx-7), ist dieser bei eingabe
20 rem  der gleichung unter 100...900 mit m (= masstab)
21 rem  zu multiplizieren (z.b. y=xdmx-7dmm).
22 rem
```

```
23 rem  die darstellung ist auf ein 15 x 64 terminal abgestimmt.
24 rem
25 rem  fuer neuzeichnen eines bereits berechneten inhalts
26 rem  ohne neuberechnung kann aus zeitersparnisgruenden
27 rem  einfach im 'immediate mode' der befehl pr #6 a #,
28 rem  eingegeben werden.
29 rem
30 rem
31 rem
32 pr'eingabe:  0 = loeschen'
33 pr'         1-5 = gewuenschte funktion'
34 a=top : input q : if q=0 link 8112 : goto 34
35 m=1 : for x=-31 to 31 : gosub 100#mq :rem   maximum suchen
36 if y#k0 y=-y                          :rem   betrag von y
37 if (y#>7) and (m #k y/7) m=y/7    :rem   wenn noetig, m (mass-
38 next x                           :rem     stab)  up to date
39 print 'masstab: ',m
40 for x=-31 to 31 : gosub 100#mq       :rem   funktion berechnen
41 y=y/m                               :rem   masstab anpassen
42 #a(a+343+x-y#m34)=#h30+q               :rem   punkt in kurve
43 next x
44 pr #s#,                           :rem  plotten (als string)
45 pr'   verkleinerungsfaktor in y-richtung: ',m#,
46 goto 46             :rem  um prompt + linefeed zu unterdr.
47 rem
48 rem  im folgenden sind einige beispielfunktionen eingetragen:
49 rem
100 y=x                  :rem  einfacher geht's nimmer
110 return
200 y=x#mx               :rem  quadratfunktion
210 return
300 y=x#mx#mx             :rem  kubikfunktion
310 return
400 if x#k#g0 y=100/x      :rem  reziprokfunktion
410 return
510 return
610 return
710 return
810 return
910 return
```

```
o goto 44
 1 y=x#mx-25#mm
 3 return
 4 rem
 5 rem
 6 rem        + + + + + + + + + + + + + + + + +
 7 rem        +  y/x - funktionsdarstellung +
 8 rem        +      (c)  manfred reimer    +
 9 rem        + + + + + + + + + + + + + + + + +
10 rem
11 rem
12 rem
13 rem  das vorige programm demonstrierte, wie begrenzt die
14 rem  moeglichkeiten der grafischen darstellung mit einem
15 rem  nicht-grafischen terminal sind. die ohnehin schon
16 rem  grobe aufloesung ist ausserdem in y-richtung sehr
17 rem  viel geringer als in x-richtung obwohl es anders
18 rem  herum juenstiger waere.
```

```
19 rem  dieses programm kehrt daher die darstellung um:
20 rem  die senkrechte achse ist die x-achse. da nun wesent-
21 rem  lich weniger werte berechnet werden muessen, ist das
22 rem  programm auch schneller. aus geschwindigkeitsgruenden
23 rem  wurde auch diesmal sehr dicht gepackt und auf kommen-
24 rem  tare verzichtet. die funktion ist auch praktisch
25 rem  identisch zum vorigen programm, nur dass x und y ver-
26 rem  tauscht sind.
27 rem  es kommt das gleiche maschinen-unterprogramm fuer
28 rem  das koordinatensystem zur anwendung.
29 rem  bei diesem programm ist nur eine funktion zur zeit
30 rem  darstellbar, die in zeile 1 abgelegt werden muss.
31 rem  nach dem plotten werden am unteren bild-
32 rem  rand die funktionsgleichung und der masstab vermerkt.
33 rem  als beispiel ist die funktion y=x↑mx-25↑mm angegeben.
34 rem
35 rem  der wert 4112 in zeile 43 ist die adresse der programm-
36 rem  zeile 1. der wert gilt fuer das 'reimer-tiny-basic',
37 rem  bei dem 'page 1' bereits bei 1000 (hex) beginnt. fuer
38 rem  alle anderen nibl-varianten (national, elektor, usw.)
39 rem  muss in zeile 48 der wert 7470 eingesetzt werden.
40 rem  in beiden faellen soll das programm in page 1 stehen.
41 rem
42 rem
43 rem
44 a=top:link8112:m=1:forx=-7to7:gosub1:ify↑k0y=-y
45 if (y↑g31)↑m(m↑ky/31+1)m=y/31+1
46 nextx
47 forx=-7to7:gosub1:y=y/m:↑a(a+343+y+x↑m64)=42:nextx:pr↑sa↑,
48 pr'  funktion: ',↑s4112,'                    y-masstab (m): ',m↑,
49 goto49
```

maschinen-unterprogramm zur erzeugung des koordinatensystems
===

fuer plot-programme
===================

(anmerkung: das programm musste zum auflisten in einen anderen
speicherbreich verlegt werden. es steht normaler-
weise bei 1fb0 ... 1ffb (startadresse in dez.
= 8112)

```
0cb0  08
0cb1  c200
0cb3  31        ptr 1 aus variable 'a' laden ('start' des
0cb4  c201      stringbereichs)
0cb6  35
0cb7  c404      zwei erschachtelte schleifenzaehler laden:
0cb9  c841      einen auf 4
0cbb  c400      einen weiteren auf 0 (= 256), produkt = 1024 =
0cbd  c83c                                        schirmflaeche
0cbf  c420      bildschirmflaechen-speicherbereich mit 'space'
0cc1  cd01      fuellen
0cc3  b836
0cc5  9cf8      fertig?
0cc7  b833
0cc9  9cf4
0ccb  c40d      carriage return anfuegen (fuer ausgabe als string)
0ccd  c900
0ccf  03
```

```
0cd0  31
0cd1  fc00        ptr 1 durch subtraktion auf anfang der mittleren
0cd3  31          zeile (= x-achse) stellen
0cd4  55
0cd5  fc02
0cd7  55
0cd8  c440        zeilenlaenge in zaehler
0cda  c61f
0cdc  c42d        aus '-' die x-achse bilden
0cde  cd01
0ce0  b819
0ce2  9cf8        fertig?
0ce4  03
0ce5  31
0ce6  fc21        ptr 1 durch subtraktion auf die mitte der obersten
0ce8  31          zeile bringen
0ce9  55
0cea  fc02
0cec  55
0ced  c410        zeilenanzahl in zaehler
0cef  c60a
0cf1  c421        aus ausrufungszeichen die y-achse bilden
0cf3  cd40
0cf5  b804
0cf7  9cf8        fertig?
0cf9  3f          wenn ja, zurueck ins basic-programm
0cfa  00
0cfb  00          zwei zaehler-bytes
```

```
 1 rem       + + + + + + + + + + + + + + + + + + +
 2 rem       +  dezimal - hex - umwandlung       +
 3 rem       +       (c)  manfred reimer         +
 4 rem       + + + + + + + + + + + + + + + + + + +
 5 rem
 6 rem
 7 rem  funktion: das ergebnis ist ein 4-byte-string, dessen
 8 rem  einzelne bytes durch fortgesetzte division der eingabe
 9 rem  durch 16 (gleichbedeutend mit rechts-schieben um 4 bit-
10 rem  positionen) gebildet werden.
11 rem  der entsprechende ascii-wert entsteht durch addition
12 rem  von 48 (zeile 26) und  - bei a...f -  von 7 (zeile 27)
13 rem  die programmzeilen 25 und 29 erlauben auch die eingabe
14 rem  von negativen dez.-werten. das ergebnis erscheint dann
15 rem  als zweierkomplement. bei nur positiver eingabe koennen
16 rem  diese zeilen entfallen.
17 rem
18 rem
19 da(top+4)=13:rem  stringende mit 'carriage return' versehen
20 pr'':pr 'decimal to hex conversion'
21 pr'':pr 'dec';d,
22 input d
23 w=0: if d<0 d=d and &h7fff: w=&h800
24 for i=3 to 0  step -1
25   a=top+i
26   da=(d and 15)+48
27   if da>&h39 da=da+7
28   d=d/16
29   d=d or w: w=0
30 next i
31 pr 'hex: ', as top
32 pr'':pr'':pr''
33 goto 20
```

SC/MP wird erwachsen - oder:
Wie man aus einem Mikroprozessor einen "Mikroprofessor macht . . . . .

Dieser Aufsatz beschreibt ein Programmpaket von 1K, das nicht nur einen kompletten Hexadezimalrechner enthält, sondern neben zahlreichen weiteren Routinen auch eine softwaremäßige Erweiterung des SC/MP - Befehlssatzes beinhaltet.

Haben Sie sich auch schon mal über die "Beschränktheit" des SC/MP bei bestimmten Aufgaben geärgert? Zum Beispiel über die umständlichen Prozeduren, die zum Aufruf einer Subroutine nötig sind - und die vermutlich zu dem Gerücht geführt haben, daß dies beim SC/MP überhaupt nicht möglich sei?

Selbst das relativ komfortable Monitorprogramm des Elektor-SC/MP-Systems mit seinem Software-Stack, das ELBUG, erfordert immerhin 15 Bytes an Programm, um eine Subroutine (mit Register-Save) aufzurufen. Noch komplizierter wird es, wenn Sie der Subroutine auch noch Daten übermitteln wollen: das geht nur auf dem Umweg über einen Zwischenspeicher im RAM (was beim SC/MP wiederum etliche Bytes erfordert).

Besitzer von SC/MP-Systemen mit ELBUG, die mal versucht haben, beispielsweise die GETHEX- oder PUTHEX- Routine von eigenen Programmen aus aufzurufen, wissen ein Lied davon zu singen.....

Für diese Leute wurde "WILLY" gemacht.

Kennen Sie WILLY .... ?

WILLY - das ist ein eigenwilliger Name für das eigenwillige, entwickelte Konzept der modularen Monitorprogrammerweiterung. Es besteht aus einigen sehr wirkungsvollen Grundroutinen, die ab Adresse FOOO (oder einer anderen) im 1. K untergebracht werden (RAM oder EPROM).

Darauf aubauend, ist eine schrittweise, nahezu beliebige Erweiterung möglich, die inzwischen auf 1,5 K angewachsen ist, aber noch längst nicht abgeschlossen ist. Sie werden auch in Zukunft über die hinzukommenden Erweiterungen informiert. Wichtig ist, daß das System zu jedem Zeitpunkt bereit voll entsprechend der jeweiligen Ausbaustufe genutzt werden kann, da alle benötigten Grundroutinen im 1. K vorhanden sind und alle zusätzlichen "Programm-Module" unabhängig voneinander arbeiten können. Jeder kann daher auch eigene, individuell wichtige Routinen selbst hinzufügen, da er nicht an eine vorgegebene Programmstruktur gebunden ist.

Da der weitere Ausbau also unterschiedlich sein kann, sollen hier nur die Grundroutinen im 1. K - also das eigentliche "WILLY"-Programm - vorgestellt werden. Oberstes Ziel bei der Entwicklung war eine völlige Kompatibilität mit dem ELBUG, dessen Vorteile voll erhalten bleiben bzw. noch ausgebaut werden.

Subroutinen aufrufen - komfortabler als beim Z80

WILLY bietet u.a. die Möglichkeit, den SC/MP-Befehlssatz softwaremäßig zu erweitern. In der 1-K - Grundausstattung von WILLY wird das ermöglicht, was man beim SC/MP am meisten vermißt: ein Befehl. Es handelt sich um einen 3-Byte-Befehl, bestehend aus einem "Opcode" und zwei Adreßbytes. Der SC/MP - "CALL-Befehl" des WILLY hat folgendes Aussehen:

Mnemonic:	Opcode:	
CALL	3F HH LL	(HH=Zieladresse high; LL= Zieladresse low.)

Für den SC/MP ist "3F" natürlich nach wie vor ein "XPPC 3"; die besondere Bedeutung entsteht dadurch, daß Ptr 3 (und "ROUTAD" im ELBUG) bereits vor Ansprung eines Anwenderprogrammes im WILLY entsprechend vorbereitet werden.

Werden diese Parameter in Anwenderprogrammen nicht verändert, gilt

jedes XPPC 3 darin als "CALL-Befehl" und die folgenden 2 Bytes als Adresse. Ein konventionaller Subroutinenansprung ist natürlich nach wie vor möglich, da er das (Neu-) Laden von Ptr 3 ohnehin beinhaltet.

Bei Ausführung des "CALL-Befehls" werden die Stack-Funktionen des ELBUG automatisch ausgeführt, so daß nach Rückkehr vom Unterprogramm die Register wieder ihren alten Inhalt haben - außer Akku und Extension-Register: diese bringen ihren Inhalt aus der Subroutine mit, was meist wesentlich praktischer ist. Außer der "normalen" Stack-Funktion werden die Registerinhalte auch ins Unterprogramm mitgenommen und können dort benutzt, aber auch nach Belieben verändert werden. A- und E- Register bilden wieder eine Ausnahme: sie erhalten den Wert 0, der ja auch oft gebraucht wird.

Beispielsweise kann Ptr 2 mit 3 Bytes (32, 40, 36) auf 0000 als RAM-Pointer geladen werden, und zwar so, daß das Displacement direkt dem lower Byte der anzusprechenden Adresse entspricht, was Rechenarbeit spart. Aus dem RAM kann bei Bedarf aber auch der alte Inhalt von A- und E- Register geholt werden.

Bei Beachtung der "Spielregeln" ist übrigens Subroutinenverschachtelung ("Nesting") bis zu beliebiger Tiefe möglich, sofern genügend RAM-Bereich für den wachsenden Stack freigehalten wird.

Da Ptr 3 zur Realisierung des "CALL-Befehls" benutzt wird, ist er normalerweise für den WILLY-Benutzer tabu (fast!). Das ist trotz der sparsamen Ausstattung des SC/MP mit Indexregistern kein Nachteil: Wer mit WILLY Erfahrungen gesammelt hat, wird sich ohnehin bald eine neue, äußerst zeit- und bytesparende Programmiertechnik zulegen (leider kein klangvoller Name):

Programmieren in Einzelschritten

Durch die "Entkrampfung" des Subroutinenaufrufs lohnt es sich nun, auch sehr kleine oder nur einmalig benutzte Programmteile als Subroutine zu schreiben. So kann man ein Problem in zahlreiche Einzelblocks (Subroutinen) mit genau definierten Aufgaben zerlegen

und erhält ein unglaublich übersichtliches und leicht änderbares Gesamtprogramm. In der Regel geht das Programmieren mit WILLY schneller vonstatten als mit einem Assembler!

Die "Dauerbelegung" von Ptr 3 stört nun überhaupt nicht mehr, da Ptr 1 & 2 in jeder Unterprogrammebene und in jedem Programmmodul erneut frei verwendbar sind!

Bestes Beispiel für die Effizienz dieser Programmiertechnik ist WILLY selbst: Wenn im folgenden die weiteren WILLY-Eigenschaften besprochen werden, werden Sie staunen, was sich in 1 K alles unterbringen läßt - selbst beim SC/MP . . . !

Zusätzliche Monitorkommandos

WILLY erlaubt das Hinzufügen von weiteren acht Kommandotasten, die sämtlich mit Doppelfunktionen (ähnlich wie bei manchen Taschenrechnern) belegt werden können, und zwar beliebig. Die Zahl der direkt durch Tastendruck aufrufbaren Kommandos wir also im Vergleich zum ELBUG max. verdreifacht. Der Ausbau ist außerordentlich einfach, da außer den 8 Zusatztasten keine weitere Hardware nötig ist: Die bisher bei den meisten Systemen wohl unbenutzten Mux-Eingänge der Erweiterungsplatine dienen jetzt zum Anschluß der Tasten und erhalten nun dadurch endlich eine sinnvolle Beschäftigung. Die komplette Software zum Abfragen und Auswerten der Mux-Eingänge ist im WILLY enthalten und kann auch von Anwenderprogrammen aus aufgerufen werden.

Mit der "UP"-Taste des Elektorsystems kann durch wiederholtes Drücken zwischen der 1. und 2. Tastenfunktion hin- und hergeschaltet werden. Im WILLY ist eine Tabelle enthalten, die für jede Taste zwei Doppel-Bytes enthält. Hier können, entsprechend den gewünschten Funktionen der einzelnen Tasten, die zugehörigen Startadressen der bei Tastendruck anzuspringenden Routinen hinterlegt werden. Allerdings sollten Sie den Bereich der ersten Tastenfunktion für die bei späteren WILLY-Erweiterungen hinzukommenden WILLY-Kommandos

freihalten. Die zweite "Tastenebene" hingegen steht Ihnen voll zur Verfügung, damit Sie acht Ihrer wichtigsten Programme in Zukunft einfach per Tastendruck starten können.

Brauchen Sie einen Hex-Rechner .... ?

Natürlich, jedenfalls dann, wenn Sie nicht nur "fertige Kost" konsumieren, sondern auch Spaß an selbstentwickelten Programmen haben. Das ELBUG enthält ja zu diesem Zweck bereits ein Subtraktionsprogramm für Berechnungen bei relativer Adressierung. Für alle anderen Zwecke gibt es Taschenrechner (z.B. TI - Programmer), die in mehreren Zahlensystemen (meist dez. und hex.) arbeiten können und auch die entsprechenden Umwandlungen gestatten.

Einen solchen Taschenrechner brauchen Sie nicht, wenn Sie WILLY besitzen: darin ist nämlich ein kompletter 8-stelliger Rechner enthalten, der intern sogar mit 9 Stellen arbeitet. Er bietet alle Funktionen, die man beim Programmieren braucht:

- Hexadezimale Addition
- Hexadezimale Subtraktion
- Dezimale Addition
- Bildung des Zweierkomplements
- Umwandlung dezimal in hexadezimal
- Umwandlung hexadezimal in dezimal

Im Rechner-Modus "landen" Sie automatisch, wenn Sie nach Reset eine der Hex-Tasten zur Zifferneingabe drücken. Als "Funktionstasten" des Rechners dienen wieder - und das ist deren dritte Funktion - die 8 Zusatztasten an den Mux-Eingängen. Für die Rechner-Funktion werden vorerst 6 davon genutzt, so daß auch der Rechner später noch um zwei Kommandos erweitert werden kann.

Eine Verwirrung bezüglich der zahlreichen Funktionen der 8 Zusatztasten ist nicht zu befürchten, da Kommando- und Rechnerfunktion sachlich eindeutig getrennt sind und die entsprechenden Tastendrücke jeweils in einem ganz bestimmten Zusammenhang erfolgen.

Die Bedienung des Rechners (und auch die Anzeige) ist genauso, wie Sie es von einem Taschenrechner mit "Umgekehrter Polnischer Notation" (UPN) gewohnt sind, d.h. es wird zuerst der (oder die) Operand(en) eingegeben und danach die gewünschte Funktionstaste gedrückt. Übrigens sind auch hier Kettenrechnungen möglich, d.h. das in der Anzeige stehende Ergebnis kann unmittelbar für weitere Berechnungen benutzt werden.

Auch die mathematischen Funktionen können Sie nach Belieben von Ihren eigenen Programmen aus aufrufen - mittels "CALL-Befehl"!

Einzelbefehle ersetzen ganze Programme

Die Liste der WILLY-Möglichkeiten geht noch weiter: Für sämtliche Hardware-Systemkomponenten (Tastatur, Display usw.) sind komplette Ein/Ausgabe - Routinen vorhanden. In Zukunft können Sie durch Aufruf der entsprechenden Routine, also mit einem 3-Byte-Befehl, die Tastatur abfragen, das Display beschriften oder ein Byte über eine serielle Schnittstelle ausgeben und vieles mehr. Und das, ohne vorher erst einen Pointer laden zu müssen!

Der Platz reicht nicht, um alle Routinen ausführlich behandeln zu können; gestatten Sie daher nur eine Aufstellung mit Kurzbeschreibung. (Es werden hier die Original-Routinennamen aus dem WILLY-Manual benutzt):

GETKEY	Dient zum Laden der Hextasten; Rückkehr mit geladenen Wert im Akku.
MUXIN	Dient zum Laden der Mux-Eingänge. Rückkehr mit Nummer der gedrückten Taste in E.
KEYREL	(Key Release) Entprellung und Warten auf Loslassen der Taste.
CLDISP	Löscht das Display.

GHEX — WILLY-Äquivalent von GETHEX (ELBUG)

PHEX — WILLY-Äquivalent von PUTHEX (ELBUG). Enthält Zusatzfunktion: Wurde zuvor mit GHEX eine "END-adresse" eingegeben, bis zu deren Erreichen ein bestimmter Vorgang periodisch wiederholt werden soll, prüft PHEX bei der Anzeige "nebenbei" automatisch, ob die anzuzeigende Adresse bereits die Endadresse erreicht hat. Die Anwendung dieser häufig gebrauchten Eigenschaft demonstriert das Programmbeispiel am Schluß.

DISPI — (Display immediate). Dient zur sofortigen Anzeige des Akku-Inhalts auf einem beliebigen Display-Digit.

DISPS — (Display String) Dient zum Beschreiben mehrerer oder aller Display-Digits auf einmal mit Buchstaben oder Ziffern (z.B. zur Anzeige von Programmnamen o.ä.). Der gewünschte Display-Inhalt wird dem "CALL DIPS" Befehl einfach als "String" angehängt - dieser wird bei Rückkehr automatisch übersprungen.

TELEXIN — Zum Empfang von Telexzeichen. Nach Rückkehr steht das empfangene Zeichen im Akku. (Der Anschluß eines Telex-Fernschreibers als preiswerte weitere Ein/Ausgabeeinheit bzw. als Hardcopy-Gerät wird im WILLY-Manual ausführlich beschrieben).

TELEXOUT — Zum Senden von Telexzeichen. Das im Akku befindliche Telexzeichen wird ausgegeben. Evtl. nötige Buchstaben/Ziffern - Umschaltung wird automatisch ausgeführt. Ferner wird ein RAM-Byte auf die aktuelle Position des Schreibkopfes gesetzt, so daß Tabulatorfunktionen einfach realisiert werden können.

TABASTE — Tabelle zur ASCII-Telex - Umwandlung (und umgekehrt). Die zugehörigen Programme sind Bestandteil der nächsten WILLY-Erweiterung.

Zwei flotte Beispielprogramme

Lassen Sie mich nun anhand zweier einfacher Beispielprogramme demonstrieren, wie einfach das Programmieren mit WILLY ist. Im ersten Beispiel soll die Aufgabe darin bestehen, seriell eintreffende Telexzeichen aufeinanderfolgend im RAM abzuspeichern und auf dem Display gemeinsam mit der momentanen Adresse anzuzeigen. Die RAM-Anfangsadresse soll zuvor über die Hex-Tastatur eingegeben werden.

Wenn Sie mit WILLY vertraut sind, schreiben Sie ein solches Programm in weniger als 5 Minuten:

```
0C00  3FF097  START: CALL GHEX      ;Startadr. holen
0C03  31             XPAL 1         ;Ptr 1 damit laden
0C04  40             LDE            ;(low-Byte aus Akku,
0C05  35             XPAH 1         ;high-Byte aus Ext.)
0C06  3FF0BE  LOOP:  CALL TELEXIN   ;Telexzeichen holen
0C09  CD01           ST@ +1 (1)     ;in RAM speichern
0C0B  3FF171         CALL PHEX      ;Zeichen & Adr. anzeigen
0C0E  90F6           JMP LOOP       ;nächstes Zeichen
```

Es ist kaum zu glauben; ganze 16 Bytes werden für diese Aufgabe benötigt!

Im folgenden Beispiel sollen aus einem beliebigen RAM-Bereich alle Bytes mit dem Inhalt "0" herausgesucht und die entsprechenden Adressen angezeigt werden. Anfangs- und Endadresse des RAM-Bereichs sollen zuvor über die Tastatur eingegeben werden. Und so könnte man dieses Problem meistern:

```
0C00  3FF097  START: CALL GHEX      ;Startadresse holen
0C03  314035         wie oben       ;Ptr 1 damit laden
0C06  3FF097         CALL G
0C00  3FF097  START: CALL GHEX      ;Startadresse holen
0C03  314035         wie oben       ;Ptr 1 damit laden
0C06  3FF097         CALL GHEX      ;Endadresse holen
0C09  C501    LOOP:  LD@ +1 (1)     ;Byte laden
0C0B  3FF171         CALL PHEX      ;Anzeige & Ende-Test
0C0E  9C01           JNZ CONT       ;Nicht Null: weiter
```

```
0C10  00          HALT       ;sonst Halt
0C11  40    CONT: LDE        ;Ende-Kennz. ist in E
0C12  9CF5        JNZ LOOP   ;nicht fertig: LOOP
0C14  3F0000      CALL ELBUG ;Ende: ELBUG-Rücksprung
```

Auch dieses Programm - es ist ganze 23 Bytes lang - ist in wenigen Minuten erstellt.

Wenn Sie mehr über WILLY wissen möchten, rufen Sie an, oder schreiben Sie.

Sie können WILLY auch kaufen: für 125, – DM auf Casette oder für 150, – DM auf einem EPROM 2708. In beiden Fällen ist das 42-seitige, humorvoll geschriebene WILLY-Begleitbuch mit Benutzungs-Anleitung und ausführlich dokumentiertem Programmlisting im Preis eingeschlossen, ebenso die notwendige geringfügige Umprogrammierung des ELBUG-1 - EPROMs (bitte bei Bestellung mitschicken). Auf die Gründe der ELBUG-Änderungen kann ich hier nicht eingehen. Sie sind jedoch so geringfügig, daß das ELBUG auch ohne WILLY nach wie vor voll funktionsfähig bleibt.

Info: Manfred Reimer
Haubachstraße 8
1000 Berlin 10
Tel. 030/342 14 28

SC/MP Intro-KIT

Bis zum heutigen Tage wurden viele Tausend SC/MP Intro-KIT's verkauft und die Besitzer haben jetzt die Möglichkeit dieses System zu einem kompletten Microcomputer auszubauen. Wer keinen Introkit besitzt oder nicht ändern will, kann diesen Abschnitt überspringen und sofort mit dem Aufbau der CPU-Latine beginnen.

Lassen Sie uns hier noch einmal kurz die wichtigsten Daten besprechen:

1.Der Intro-KIT-Bausatz enthält:

SC/MP Mikroprozessor-Baustein.
Ein 256 x 8 Bit RAM (Schreib-/Lesespeicher) zur Speicherung des Anwendeprogramms (2 x MM 2102).

Ein 512 x 8 Bit vorprogrammiertes ROM, welches das interne Betriebsprogramm (DEBUG-Programm) und die notwendigen Routinen zur Ansteuerung der Teletypenmaschine enthält (1 x MM 5204 oder MM 5214).

Einen 8 Bit Pufferspeicher zwischen den Speichern (RAM und ROM) und dem Datenbus (DM81LS95).

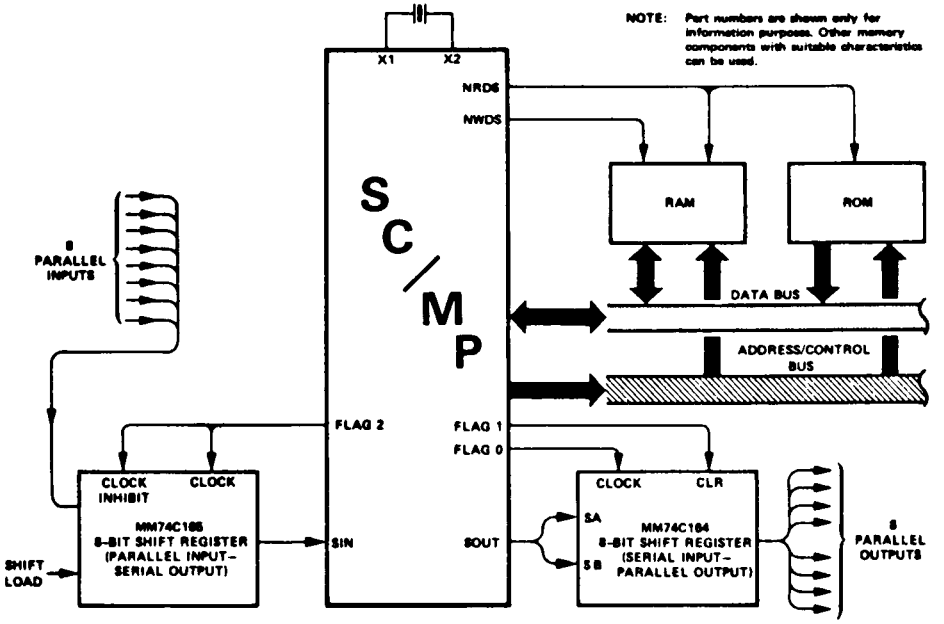
Diverse Interface-Logik zur Pegelumsatzung und zur Ansteuerung der seriellen Ausgangsschnittstelle.

Ein Spannungsregler erzeugt aus den beiden Eingangsspannungen + 5 V und - 12 V die für die Bausteine erforderlichen Betriebsspannungen von - 7 V, - 12 V und + V.

Ein Kristalloszillator sorgt für einen genauen Takt. Ferner befinden sich im Intro-KIT-Bausatz eine gebohrte Platine, Sockel, Steckerleiste und alle anderen diskreten Bauelemente.

Mit diesem KIT können Anfänger und erfahrene Fachleute ein lehrreiches und ausbaufähiges Microcomputersystem aufbauen.

Kleine Programme können über die Tastatur oder Teletype eingegeben und gestartet werden. Man kann sich auf diese Weise einfach mit der grundlegende Charakteristik eines Microcomputers und im besonderen mit dem Befehlssatz des SC/MP vertraut machen. Da der SC/MP-Befehlssatz nur aus 46 Befehlen besteht, ist er als Lernsystem besonders für den Anfänger interessant.



NS 10440

Serielle Daten können direkt über die SIN und SOUT-Ports des SC/MP-Prozessors mit der CPU in Verbindung treten.
Parallel vorhandene Daten können vorher über Schieberegister in serielle Daten umgewandelt werden. Die Steuerung der Schieberegister erfolgt über die Flag-Anschlüsse Flag 0, Flag 1, Flag 2.
In anderen Anwendungsfällen, wie z.B. beim KIT können die Flags und die beiden Sense-Eingänge Pin 17 und 18) direkt für die Übertragung von seriellen Daten verwendet werden. Hier wird der Teletype (TTy) über 7414 Schmitt-Trigger direkt mit den Sense- und Flag-Eingängen verbunden.

Der begrenzte Speicherbereich (156 Byte RAM u. 512 Bytes ROM) erlaubt es nicht mit dem NIBL (National Industrial BASIC Language) zu arbeiten. Weiterhin können die Daten und Adressleitungen nur eine TTL-Last treiben. Aus diesem Grunde sind einige Änderungen nötig, um die zusätzliche Logik anzupassen. Die Änderungen können entsprechend den nachfolgenden Schaltbildern durchgeführt werden. Die zusätzlichen Elemente werden in den freien Raum der Intro-KIM Platine mit Wire Wrap Sockeln gesetzt. Für Keyboardkit- Anwender muß vorher der entsprechende Zusatzteil entfernt werden.

Es werden zusätzlich benötigt:
Bauelemente Sockel vom ROM MM5214 (5A) entfernen, und durch einen Wire Wrap Sockel ersetzen. Das ROM selbst wird nicht mehr benötigt.

1 x DM 81LS95
2 x DM 80LS92 (6 x Puffer Fristate)
1 x 74LS175 Latch für 4 Bit Mostsegment Bit
2 x Sockel für 2716 für NIBL
1 x DM 8131 Dekoder
2 x 74LS04 Status + Flags

Da der RAM-Bereich des Intro-KIT entfallen ist, benötigen Sie für den Betrieb eine zusätzliche RAM-Karte.
Die Zusammenschaltung erfolgt über die gemeinsame Mutterplatine.

Grafik mit NIBL auf Standarddisplay

Da NIBBL keine TAB-Funktion hat, läßt sich mit einem kleinen Programmiertrick eine recht leistungsfähige Grafikeigenschaft erreichen.

```
1000    FOR x = 0 TO 24
1005    REM 24 = Anzahl der Messungen
1010    Y = RND (1,80)
1015    REM Y = Funktion zum Zeichnen
1020    ROR J = 1 TO y ← REM
1025    REM Y = max. Zeichen pro Zeile
1030    PRINT "*"; REM ";" Vermeidung von CR
1040    NEXT J
1050    PRINT "  " REM Drückt ein CR + Linefeed am Ende jeder
        Zeile
1060    NEXT X
```

Für eine einfache Kurve folgendes Programm:

```
1000    FOR y = 0 TO 24
1010     y = A*A
1020       FOR J = 1 TO Y-1
1030         PRINT "  " ;
```

```
1040        NEXT J
1050       PRINT "*"
1060      PRINT " "
1070     NEXT X
```

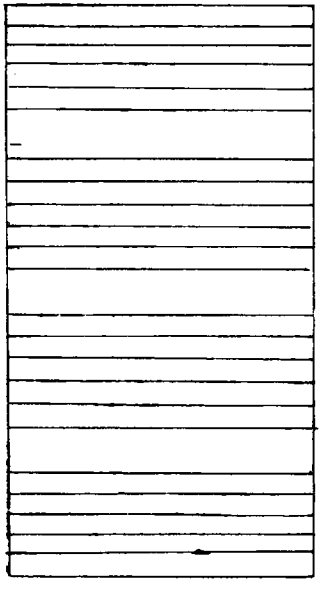
Beide Routinen können als Unterprogramme von ihrem Hauptprogramm aufgerufen werden.
Dazu muß eine Zeile mit RETURN am Ende des Programmes angefügt werden.

Programmiertrick für NIBL

Da der NIBL Interpreter kein DIM-Statement besitzt, kann man mit folgenden Trick zweidimensionale Felder programmieren. Eine Erweiterung auf multidimensionale Arrays ist ohne weiteres möglich.

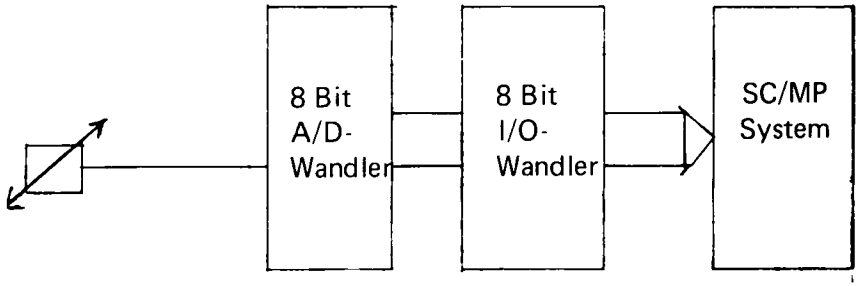
DIM (x,y) = @ (TABLE + x (5 * y)+ 1)

TABLE ist die Anfangsadresse des freien RAM-Bereiches, wo das Array abgelegt werden soll. Die Zahl 5 gibt den Maximalwert an, den x erreichen kann.



Dieses Beispiel zeigt ein Array, wo X = 5 ist und Y = 4 ist.

Applikationsbeispiel
Analog/Digital-Wandler



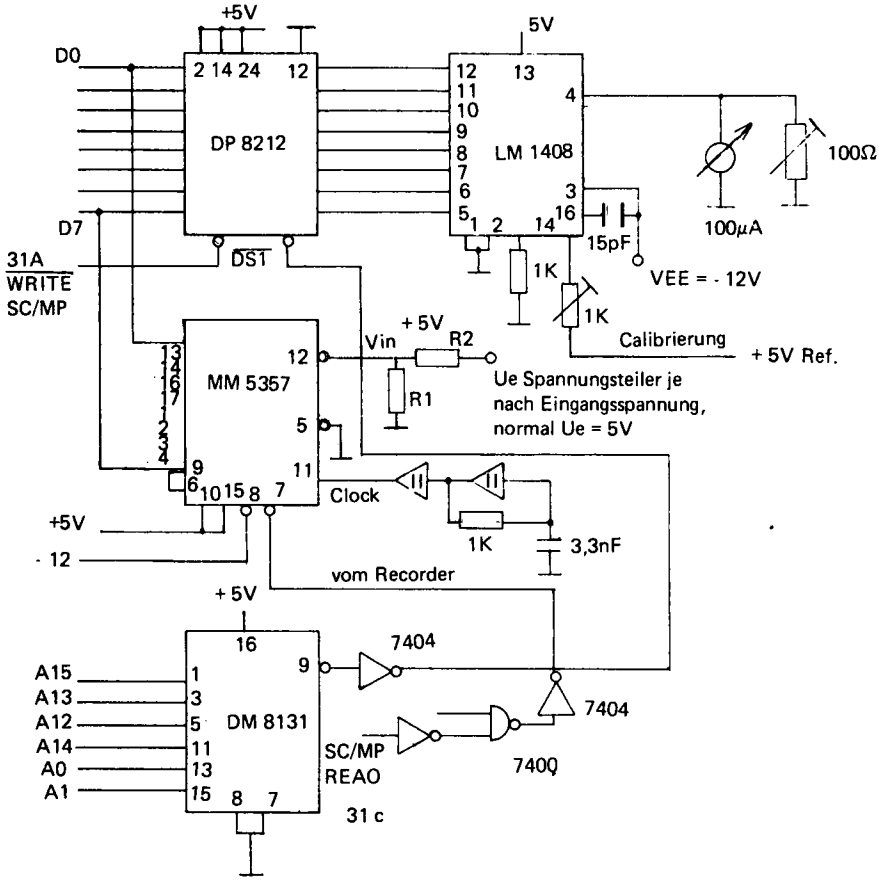

```
10      FOR X = 0 TO 500
20      Y = @ II 8500) 4
30      FOR I 1 TO Y REM Y = Zeichen/Zeile
40      PRINT "  ";
50      NEXT I
60      PRINT "*"
70      PRINT "  "
80      GO SUB  1000
90      NEXT X
100     PRINT "THIS IS END OF TO - TEST"
110     END

1000    REM DELAM ROUTINE
1010    FOR Z = 1 TO 50
1020    LET Z = Z
1030    NEXT Z
1040    RETURN
```

Zusammen mit Sensoren und einem A/D-Wandler können mit diesen Routinen grafische Profile auf einem Bildschirm gezeichnet werden.

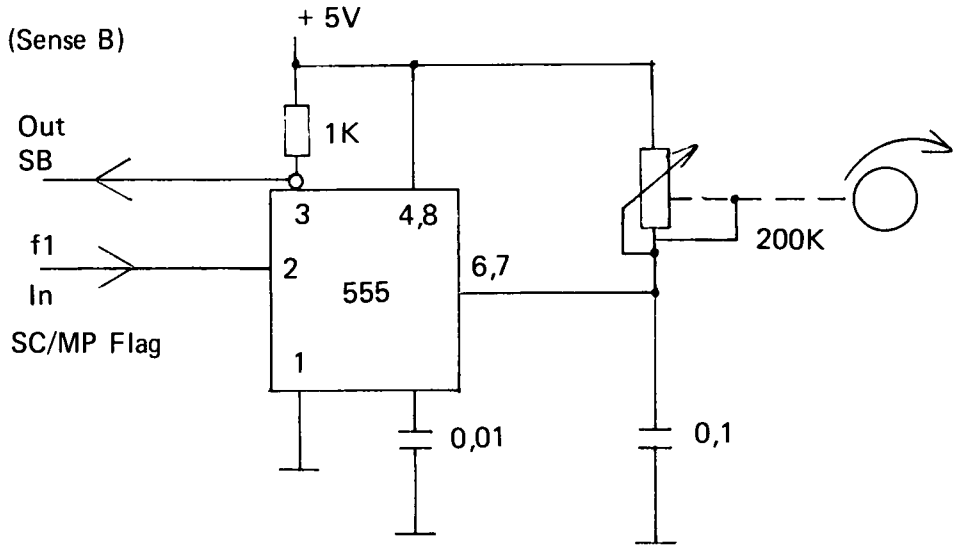
Da Y bei einem 8 Bit-Wandler 256 Möglichkeiten für Y entstehen, muß bei einem Datensichtgerät mit 80 Zeichen pro Zeile 4 geteilt werden, damit die max. Zahl 80 erreicht wird.
Eine Meßbereichserweiterung kann durch Einsetzen der folgenden Zeile erreicht werden.

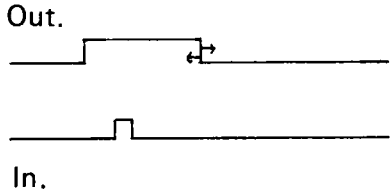
```
20      Y = ( @ # 8500) - 180
```

Zu Analog/Digital-Wandler, Digital Analog-Wandler



1. Analoger Eingang und analoger Ausgang
2. Relais-Karte / Input / Output
3. Nachlaufsteuerungs- Abtastung
   Potentiometer-Stellung wird abgefragt.





Der Timer 555 wird als Monoflop betrieben.

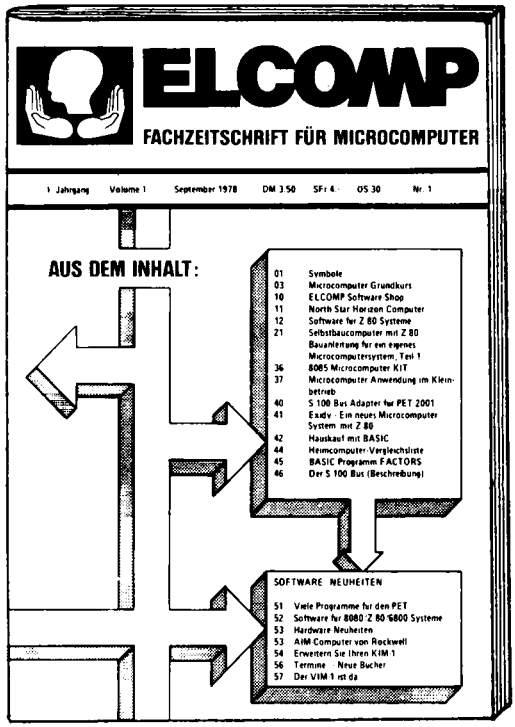
3. Programm zur Nachlaufsteuerung

```
100 LINK @ # 9000

                        Machine routine

09000   CSA                         ; set F1 to start
        ORI 02                        one shot
        CAS
        LDI     2(RAM)              ; set pointer to free
        XPAL    P1
        LDI     H(RAM)              ; RAM location
        XDAH    P1                  ; for a counter
        LDI     0
        ST      0(P1)               ; clear counter
        CSA                         ; reset F2
        ANI     0FD
        CAS
LOOP:   ILD     0(P1)               ; increment counter
        CDI     DELAY               ; Timing unit
        DLY     TIME                ; for counter
        CSA
        ANI     020                 ; book it end of pulse
        JNZ     LOOP                ; if hot try again
        LD      0(P1)               ; take countersvalue
        ST      0(P2)               ; store counter in
        XPPC    P3                  ; variable  a from NIBL
110  PRINT   A
          .
          .
          .
          .
          .
          .
```

Auch Sie brauchen ELCOMP!

Jahresabonnement DM 59,-- incl. Mwst. und Versand.

Zurückliegende Hefte zu Originalpreisen noch verfügbar.



ELCOMP

HOFACKER-VERLAG
Ing. W. Hofacker GmbH
Tegernseer Straße 18
D-8150 Holzkirchen/Obb.

Die Fachzeitschrift für MICROCOMPUTER
Eine unentbehrliche Informationsquelle für alle Elektroniker

Microcomputer-Anwendungsbeispiele
Künstliche Intelligenz
Block-Strukturierte Programme
Datenverarbeitung im Kleinbetrieb
Club-Neuheiten
Computer und Kunst
Musik mit dem Computer
Monitore für 8080, 6800, 6502, Z 80, SC/MP, 2650, 1802
Eigenbau-Computersysteme
Interface-Techniken
Microcomputer KITs
Neue Produkte
Betriebssysteme für Floppys
Programmiertechniken
Software-Quellen
Programmierbeispiele
Soziale Aspekte der Microcomputertechnik
Technologische Neuheiten
Anwendungen in der Meß- und Regeltechnik
Anwendungen bei Funk-Amateuren

Verlagsprogramm

Übersicht lieferbarer Bücher

Bestell Nr.	ISBN	Verfasser	Titel	Preis DM
1	3-921682-01-0	Hofacker	Transistor Berechnungs- und Bauanleitungshandbuch, Band 1	19,80
2	3-921682-02-9	Hofacker	Transistor Berechnungs- und Bauanleitungshandbuch, Band 2	19,80
3	3-921682-03-7	Gebauer	Elektronik im Auto	9,80
4	3-921682-04-5	Lorenz	IC-Handbuch, TTL, CMOS, Linear	19,80
5	3-921682-05-3	Steinbach	IC-Datenbuch, TTL, CMOS, Linear	9,80
6	3-921682-06-1	Steinbach	IC-Schaltungen, TTL, CMOS, Linear	9,80
7	3-921682-33-9	Hofacker	Elektronik Schaltungen	5,–
8	3-921682-08-8	Lorenz	IC-Bauanleitungs-Handbuch	19,80
9	3-921682-09-6	Lorenz	Feldeffekttransistoren	5,–
10	3-921682-34-7	Lorenz	Elektronik und Radio, 4. Auflage	19,80
11	3-921682-11-7	Lorenz	IC-NF Verstärker	9,80
12	3-921682-12-6	Bernstein	Beispiele Integrierter Schaltungen (BIS)	19,80
13	3-921682-13-4	Lorenz	HEH, Hobby Elektronik Handbuch	9,80
14	3-921682-14-2	Lorenz	IC-Vergleichsliste	29,80
15	3-921682-15-0	Lorenz	Optoelektronik Handbuch	19,80
16	3-921682-16-9	Bernstein	CMOS Teil 1, Einführung, Entwurf, Schaltbeispiele	19,80
17	3-921682-17-7	Bernstein	CMOS Teil 2, Entwurf und Schaltbeispiele	19,80
18	3-921682-18-5	Bernstein	CMOS Teil 3, Entwurf und Schaltbeispiele	19,80
19	3-921682-19-3	Lorenz	IC-Experimentier Handbuch	19,80
20	3-921682-20-7	Lorenz	Operationsverstärker	19,80
21	3-921682-21-5	Lorenz	Digitaltechnik Grundkurs	19,80
22	3-921682-22-3	Bernstein	Mikroprozessoren, Eigenschaften und Aufbau 2. Aufl.	19,80
23	3-921682-23-1	Lorenz	Elektronik Grundkurs, Kurzlehrgang Elektronik	9,80
24	3-921682-35-5	Hans Peter Blomeyer-Bartenstein	Mikrocomputer Technik	29,80
25	3-921682-25-8	C. Lorenz	Hobby Computer Handbuch	29,80
26	3-921682-26-8	H. Bernstein	Mikroprozessor Teil 2	19,80
27	3-921682-27-4	C. Lorenz	Mikrocomputer Software Handbuch	29,80
28	3-921682-28-2	C. Lorenz	Lexikon + Wörterbuch für Elektronik und Mikroprozessortechnik LEM	29,80
29	3-921682-29-0	C. Lorenz	Mikrocomputer Datenbuch	49,80
30	3-921682-30-4	C. Lorenz	Aktivtraining-Mikrocomputer	49,80
31	3-921682-31-2	C. Lorenz	57 Programme in BASIC	39,–
32	3-921682-32-0	C. Lorenz	ATARI BASIC Handbuch	29,80
33	3-921682-36-6	Dr. Hatzenbichler	Microcomputer Programmierbeispiele	19,80
34	3-921682-50-7	H. Hermann	TINY-BASIC Handbuch	19,80
35	3-921682-35-5		Der freundliche Computer	29,80
41	–	–	Experimentierplatine für 14, 16, 24, 28 und 40 polige DIL IC's	79,–
1051	–	–	TTL-Experimentierbuch	5,–
1061	–	–	CMOS-Experimentierbuch	5,–
108	3-921682-42-8	C. Lorenz	SC/MP Mikrocomputer-Handbuch	29,80
109	3-921682-43-6	P. Heuer	6502 Microcomputer Programmierung	29,80
110	3-921682-49-5	C. Lorenz	Programmierhandbuch für PET	29,80
111	3-921682-45-2	M. Stübs	Programmieren mit TRS-80	29,80
113	3-921682-48-7	C. Lorenz	BASIC Programmierhandbuch	19,80
114	3-921682-60-6	L. Oswald	Der Microcomputer im Kleinbetrieb	39,80
118	3-921682-61-4	C. Lorenz u. R. Lullus	Programmieren in Maschinensprache mit dem 6502	98,–
119	3-921682-62-2	C. Lorenz	Programmieren in Maschinensprache (Z80)	49,–
120	3-921682-64-9	M. Stübs	Anwenderprogramme für TRS-80	29,80
150	3-921682-52-5		Care and Feeding of the Commodore PET (engl.)	19,80
151	3-921682-51-7		8K Microsoft BASIC Reference Manual (engl.)	19,80
152	3-921682-67-3	S. Roberts	Expansion Handbook for 6502 and 6800 (engl.)	19,80
153	3-921682-54-1		Microcomputer Application Notes (engl.)	29,80
154	3-921682-53-3		Complex Sound Generation using the SN76477 (engl.)	19,80
155	3-921682-56-8		The First Book of 80-US(TRS-80) (engl.)	19,80
156	3-921682-68-1	S. Roberts	Small Business Programs (engl.)	29,80

Ing. W. Hofacker GmbH Verlag 8 München 75