Z80-KIT Einfach-Computer und Lernsystem

Der Z80-KIT ist ein voll funktionsfähiges Einfach-Rechner-System, das auf der Basis des Mikroprozessors Z80 von ZILOG aufgebaut ist. Es besteht aus einer autonomen Rechnereinheit, einer Ein-Ausgabemöglichkeit zur Bedienung des Gerätes, sowie einem Betriebsprogramm, das die Aktivitäten des Z80-KIT ermöglicht.

Der Z80-KIT arbeitet auf Maschinen-Code-Ebene, d.h. die Ein- und Ausgabe von Werten wird in sedezimaler (hexadezimaler) Form durchgeführt.

Eingabemöglichkeiten bestehen in Form der Sedymialtastatur (0 . . . F) für einzugebende Werte, sowie der Kommandotastatur und deren Hilfe direkte Anweisungen an das System gegeben werden können.

Vom Betriebsprogramm ausführbare Anweisungen sind

- Register auf einen Wert setzen
- Registerinhalt anzeigen
- Speicherzelle auf einen Wert setzen
- Speicherzelleninhalt anzeigen
- Blockeingabe
- Blockausgabe
- Abspeichern von Speicherinhalten auf Cassette
- Laden von auf Cassette gespeicherter Information in den Speicher
- Starten eines Anwenderprogramms
- Einzelbefehlausführung
- Setzen, Anzeigen und Löschen eines Haltepunkts (Breakpoint)

Die Ausgabe von Werten erfolgt auf einer insgesamt sechsstelligen sedezimalen Anzeige.

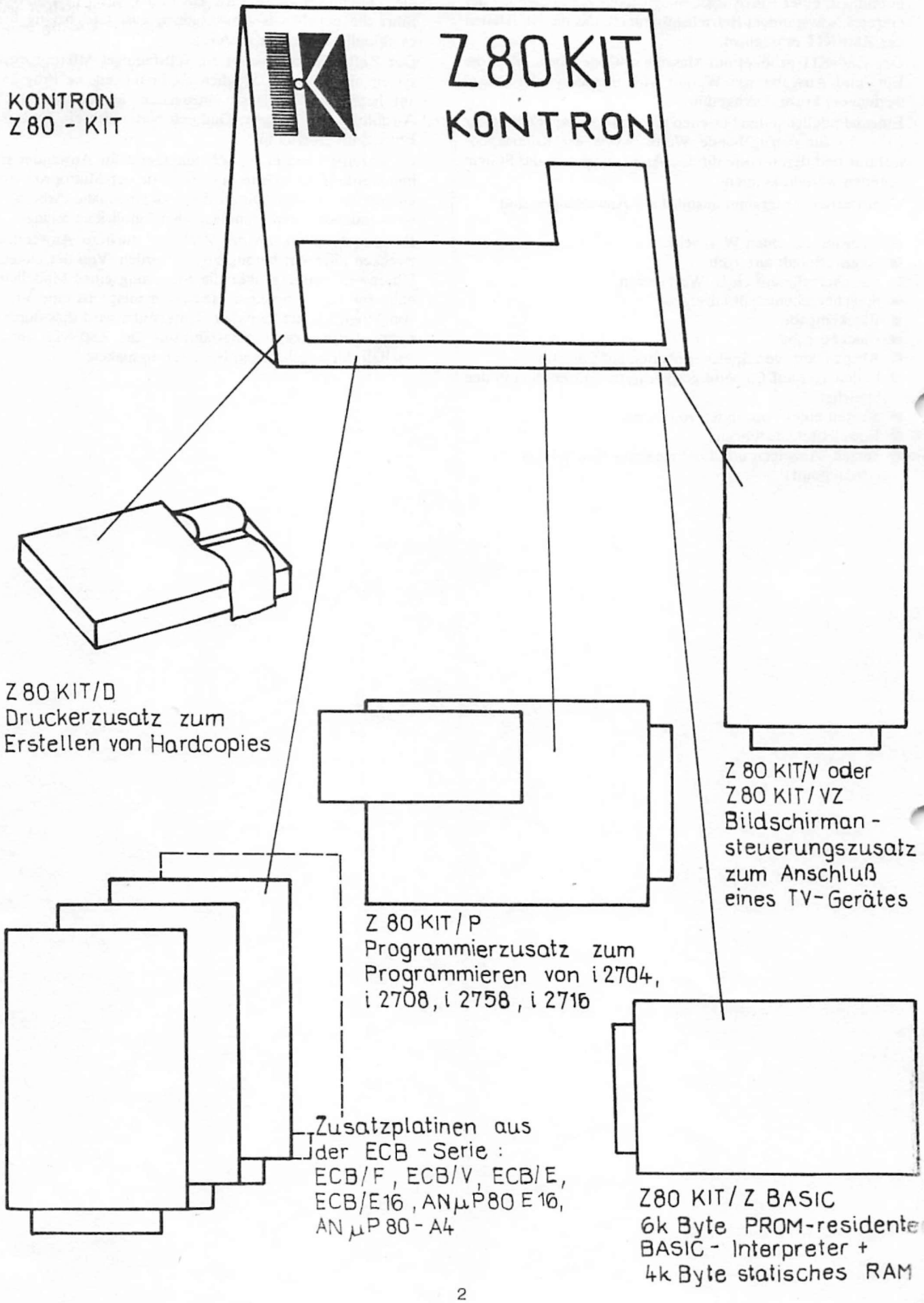
Bei Ein- und Ausgabe ist jeweils das Betriebsprogramm für die Durchführung verantwortlich. Es liest, dekodiert und führt die gegebenen Anweisungen aus. Gleichzeitig bringt es aktuelle Werte zur Anzeige.

Der Z80-KIT stellt somit ein vollständiges Mikroprozessorsystem dar, das die Möglichkeit bietet, eigene Programme auf hexadezimaler Basis einzugeben, auszutesten und zur Ausführung zu bringen. Dadurch bietet sich ein Einsatz als Entwicklungsgerät an.

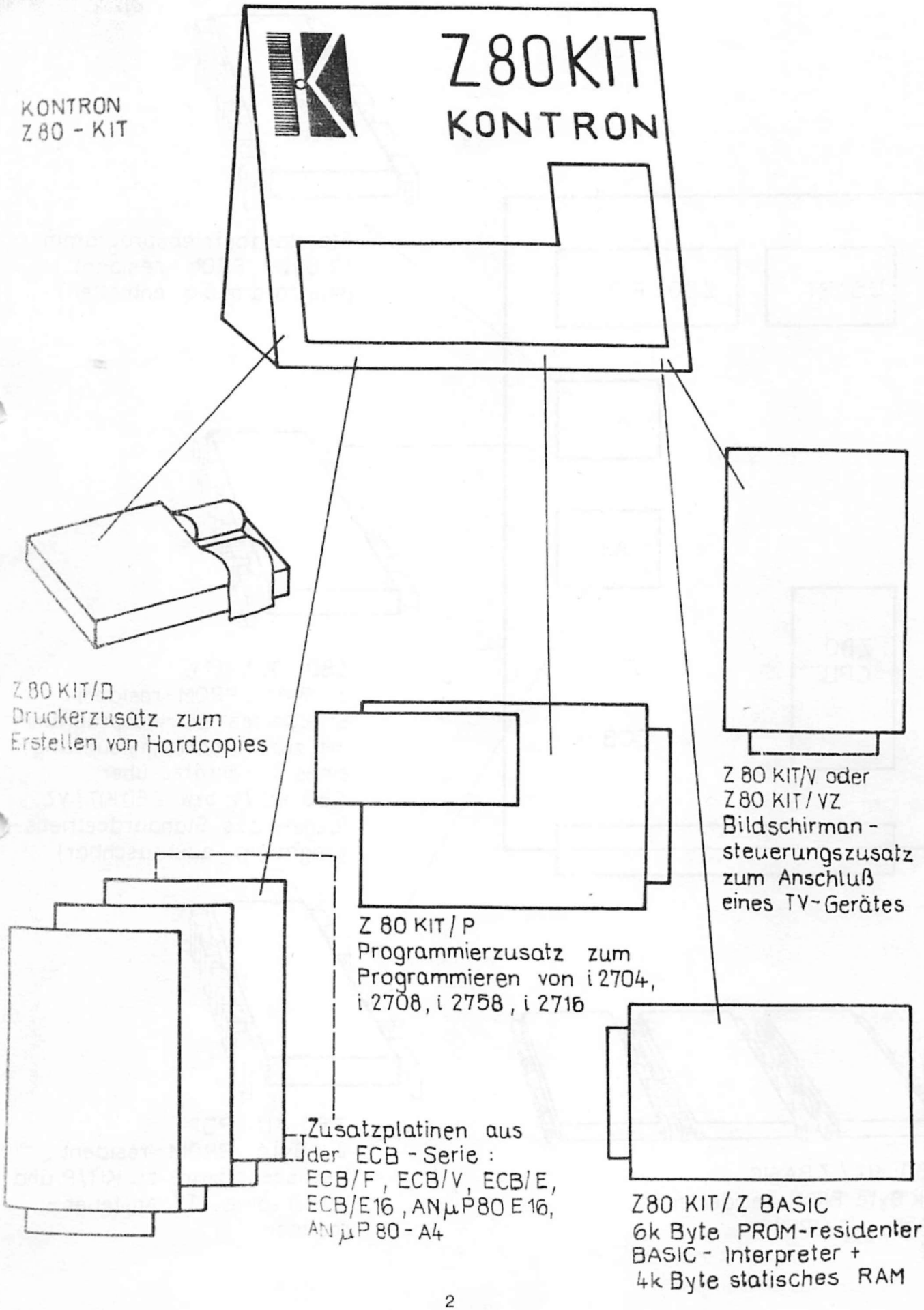
Gleichzeitig kann er als Schulungsgerät für Anwender gesehen werden, die sich in die Thematic der Mikroprozessoren einarbeiten wollen. Durch die maschinennahe Arbeitsweise ist es möglich einen grundlegenden Einblick zu erlangen.

Darüber hinaus kann der Z80-KIT auch zu Anwendungszwecken aller Art herangezogen werden. Von der einfachen Uhrenwerkshaltung über die Steuerung einer Modelleisenbahn bis zur kompletten Hausalarmanlage ist eine Vielzahl von Möglichkeiten denkbar. Unterstützt wird dies durch die beispielhafte Erweiterungsfähigkeit des Z80-KIT und der Vielfalt der angebotenen Erweiterungspakete.

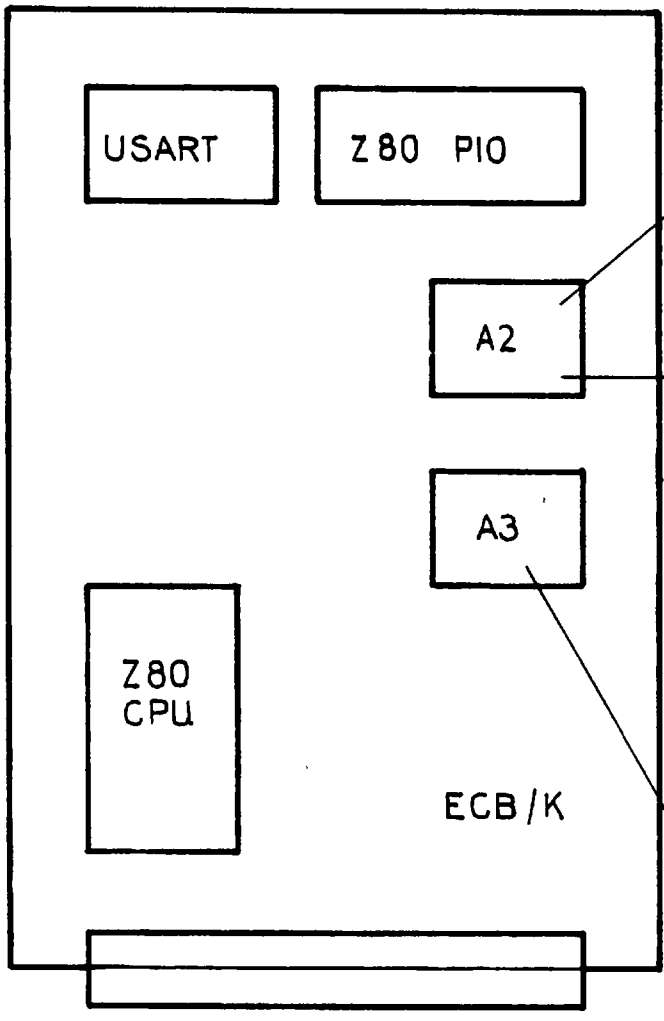
HARDWARE ÜBERSICHT

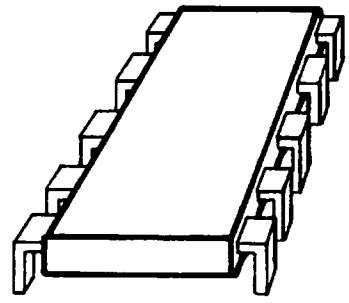


HARDWARE ÜBERSICHT

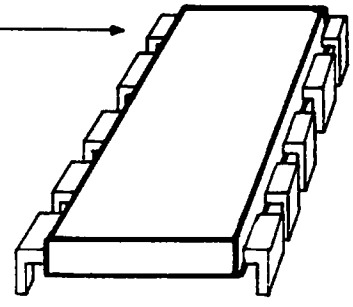


SOFTWARE ÜBERSICHT

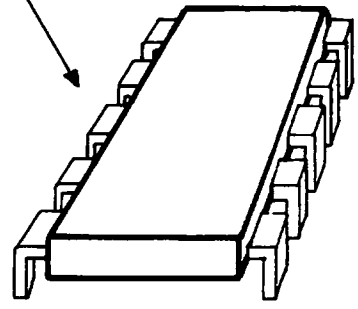




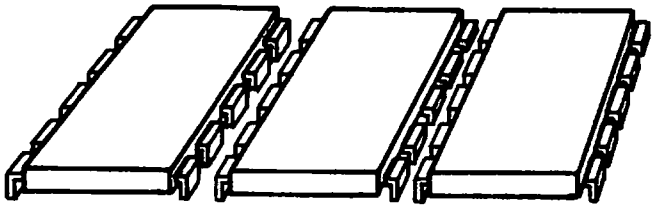
Standardbetriebsprogramm
1k Byte , PROM- resident .
(standardmäßig enthalten)



Z80 - KIT /TV
2 k Byte , PROM-resident .
Erweitertes Betriebsprogramm
mit zusätzlicher Ansteuerung
eines TV-Gerätes über
Z80 KIT/V bzw. Z80 KIT/VZ.
(gegen das Standardbetriebs-
programm austauschbar)



Z 80 KIT / PDT
2 k Byte , PROM - resident .
Betriebssoftware zu KIT/P und
KIT/D sowie TTY Ansteuer -
routinen .



Z 80- KIT / Z BASIC
6 k Byte PROM - residenter
BASIC - Interpreter

Anmerkung

Die Fa. KONTRON GmbH garantiert die volle Funktionsfähigkeit aller **im Bausatz enthaltener Teile.** Sie leistet bei nachweislich zum Zeitpunkt der Auslieferung defekten Bauteilen Ersatz.
Für das vom Kunden zusammengebaute Gerät kann grundsätzlich keinerlei Gewährleistung übernommen werden, die Funktionsfähigkeit ist vom Anwender zu realisieren.
Für Auskunft steht die Applikationsabteilung des Geschäftsbereichs Mikrocomputer jederzeit zur Verfügung.

INHALTSVERZEICHNIS

Das vorliegende Anwenderhandbuch beinhaltet folgende Teilbereiche:

Hardware

Software

KONTRON KIT®

Einfach-Computer-System
und Lernsystem
KONTRON-KIT

KONTRON
ELEKTRONIK GMBH

Z80
MIKROCOMPUTER-SYSTEME

I Z80-KIT

Inhalt:

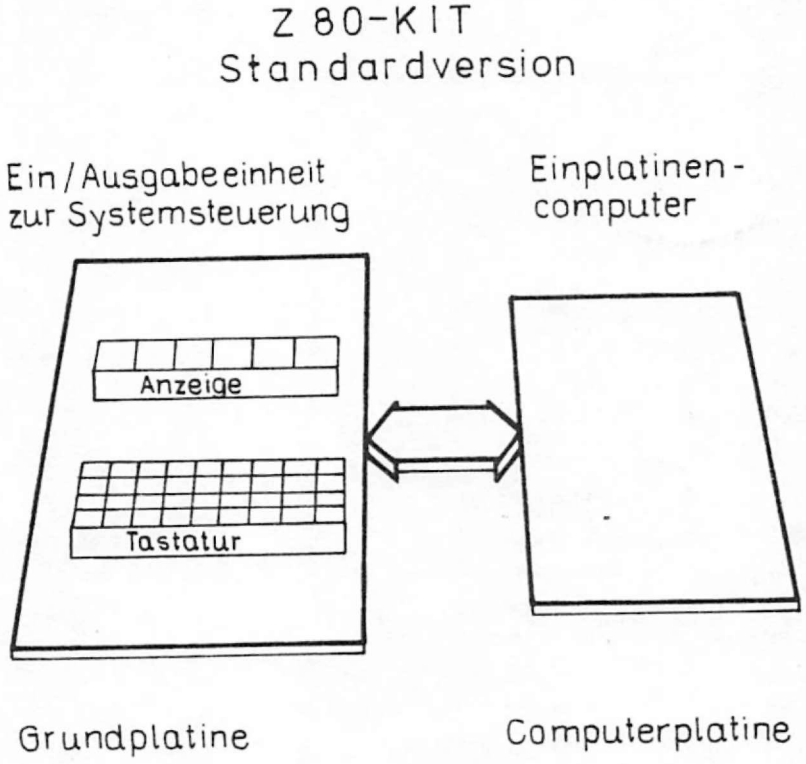
1. Einführung

1.1. Vorbemerkung

Der Z80 KIT ist ein vollständiges Z80-Mikrocomputer-System in Bausatzform, das sich in wenigen Stunden zu einem voll funktionsfähigen System zusammenbauen läßt. Es eignet sich sowohl in hervorragender Weise als Starthilfe bei der Einarbeitung in die Mikrocomputer-Technik als auch als Prototyp für Geräteentwicklungen. Dies umso mehr, da durch umfassende Erweiterungsmöglichkeiten eine Anpassung an den jeweiligen Fortbildungsgrad bzw. an entsprechende Aufgaben jederzeit möglich ist.
Zur Inbetriebnahme zusätzlich notwendig ist lediglich eine 5 V Stromversorgung (±5% Stabilität).
Die Bauanleitung ist vor Beginn des Zusammenbaus genau durchzulesen und dann beim Aufbau des Systems in der vorgegebenen Reihenfolge anzuwenden.

1.2. Grundsätzlicher Aufbau

Mit dem Z80 KIT wurde ein Mikrocomputersystem geschaffen, das in seiner Grundform keinerlei periphere Geräte zu Bedienungszwecken benötigt und dadurch die Verwendung eines funktionsfähigen Mikrocomputers bei minimalen Kosten erlaubt.
Basis dieses Mikrocomputers ist die Grundplatine, auf dem die Bedienungststatur die Anzeigeeinheiten und die dazu notwendige Steuerelektronik untergebracht sind. Ebenfalls auf dem Motherboard wurde ein System-Bus realisiert, der sämtliche Adress-, Daten und Steuersignale über fünf Vielfachsteckverbindungen zur Verfügung stellt. Über einen der genannten Steckplätze erfolgt der Anschluß eines Einplatinencomputers. Erweiterungsmöglichkeiten bezüglich Speicher etc. sind durch die restlichen Bus-Anschlüsse gegeben.



1.3. Erweiterungsmöglichkeiten

Der Z80-KIT läßt sich aufgrund seiner Architektur und seines elektronischen Aufbaus auf insgesamt fünf Platinen erweitern, wobei die vier Zusatzplatinen nach Wunsch gewählt werden können.

Angeboten werden folgende Erweiterungen:

1.3.1. Z80-KIT-Erweiterungspakete

- Z80-KIT/P — Programmier- und Peripherie-Baugruppe zur Programmierung von Festwert-Speicherbausteinen, incl. 4 kByte statische Speichererweiterung (auf 8 kByte ausbaubar) und Ein/Ausgabeerweiterung (um 32 I/O-Leitungen) (Bausatz). (Software auf Cassette).
- Z80-KIT/PZ — wie Z80-KIT/P, jedoch geprüft und zusammengebaut (Software auf Cassette)
- Z80-KIT/D — Drucker-Zusatz zum Z80-KIT zur Anfertigung von Hardcopies von Speicherinhalten. (Software auf Cassette).
- Z80-KIT/V — Bildschirmansteuerungszusatz für Z80-KIT (Anschluß an PIO, ECB/K)
- Z80-KIT/VZ — wie Z80-KIT/V, jedoch geprüft und zusammengebaut, (ECB-Bus kompatible Steckkarte)
- Z80-KIT/PDT — PROM-residente Software für KIT/P und KIT/D sowie zur Ansteuerung der TTY-Schnittstelle der ECB/K
- Z80-KIT/TV — PROM-residenter, erweiterter KIT-Monitor zur zusätzlichen Ansteuerung eines TV-Gerätes (Voraussetzung: KIT/V oder KIT/VZ)
- Z80-KIT/ZBASIC — PROM-residenter Basic-Interpreter incl. 4 kByte statische RAM-Erweiterung.

1.3.2. Baugruppen der ECB-Serie

(Ausnahme: dynamische Speicherkarten)

- Z80-ECB/E — Erweiterungsplatine 1 kB stat. RAM + Sockel für max. 8 kB PROM (2708)
- Z80-ECB/E16 — Erweiterungsplatine 1 kB stat. RAM (auf 4 kB nachrüstbar und Sockel für max. 16 kB Festwertspeicher (i2758, i2716)
- Z80-ECB/I — Ein/Ausgabeplatine mit 4 × 8 bit-Ports und 4 Zähler/Zeitgeber-Kanälen
- Z80-ECB/V — 4 kByte-CMOS-RAM, nicht flüchtig mit Batterie und Speicherschutzschalter
- Z80-ECB/F — Floppy-Disk-Controller und allgemeine Serien- und Parallel-Ein/Ausgabe
- AN-μP80-E 16 — Platine mit 16 analogen Eingängen (bzw. 8 Gegentakt) und 1 analogen Ausgang
- AN-μP80-A4 — Platine mit 4 analogen Ausgängen und Spannungswandler auf der Platine

1.3.3. Stromversorgungen

Alternativ zum Betrieb mit einer externen Quelle, kann der Z80-KIT auch mit einem, direkt am Bus ansteckbaren Stromversorgungsmodul ausgerüstet werden. Das Modul ist im Europakartenformat aufgebaut und kann in einem, extra dafür vorgesehenen Steckplatz (N) eingesteckt werden. (Steckplatz 1 und Steckplatz N können nur alternativ bestückt werden).

NB.:
Die Stromaufnahme des Z80-KIT beträgt in der Grundausstattung ca. 1A. Bei den Erweiterungen kann ein Bedarf von ca. 0,6 bis 0,8 A pro zusätzlicher Platine zugrunde gelegt werden.

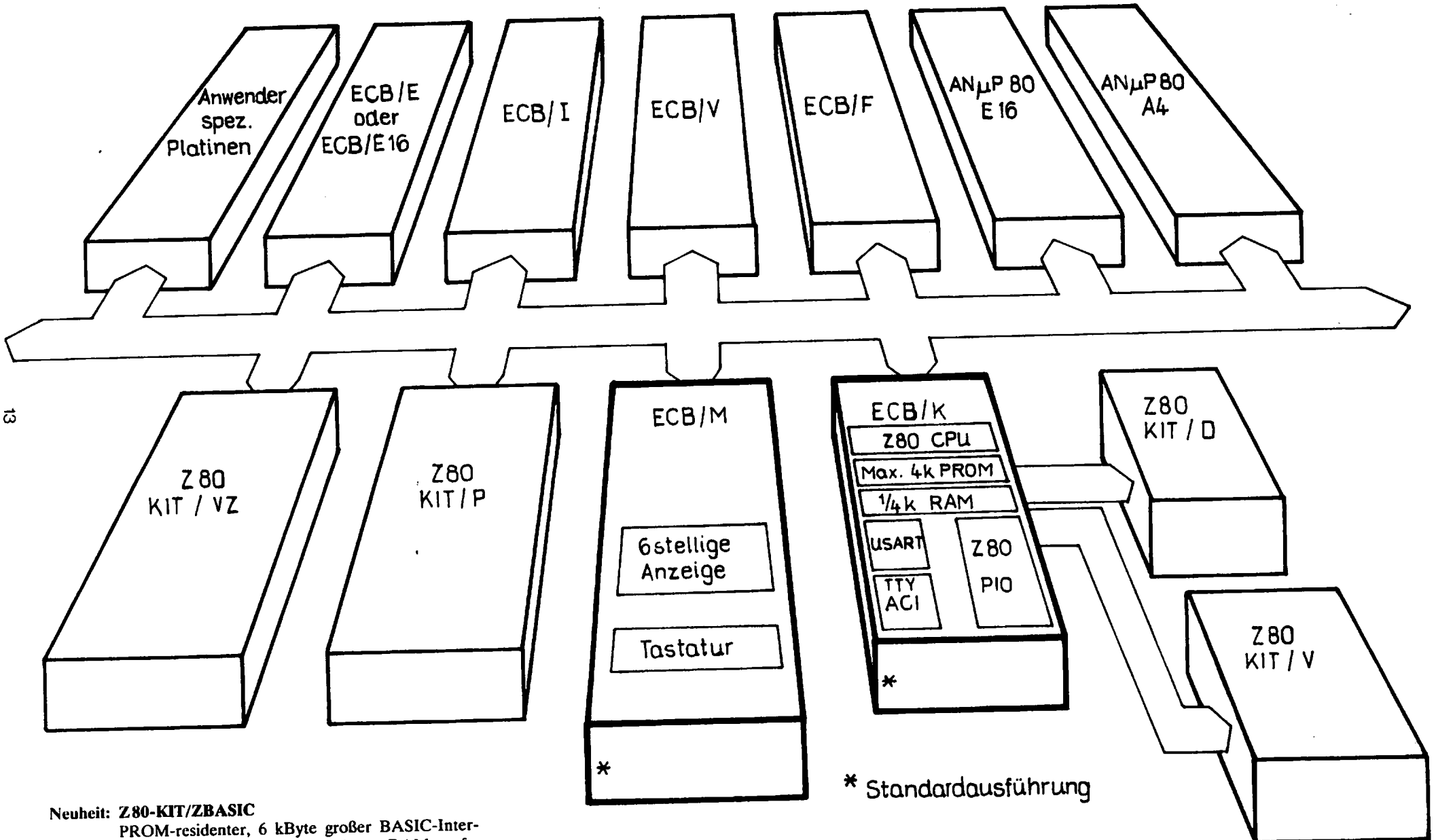
Angeboten werden folgende Stromversorgungskarten:

- KPS 535/1 — 5 Volt/3,5 A
- KPS 535/2 — 5 Volt/3,5 A, ±15 V/0,25 A
- KPS 535/3 — 5 Volt/3,5 A, ±12 V/0,3 A
- KPS 535/4 — 5 Volt/3.5 A, +12 V/0,3 A/−5 V/0,7 A.

1.3.4. Anwenderspezifische Platinen

Selbstverständlich ist auch der Anschluß von Platinen, die vom Anwender aufgebaut werden möglich. Dabei ist lediglich darauf zu achten, daß die elektrischen Grenzwerte der ECB/K nicht überschritten werden.

Erweiterungsübersicht



Neuheit: Z80-KIT/ZBASIC
PROM-residenter, 6 kByte großer BASIC-Interpreter incl. 4 kByte statischem RAM auf KONTRON-Bus-kompatibler Europakarte.

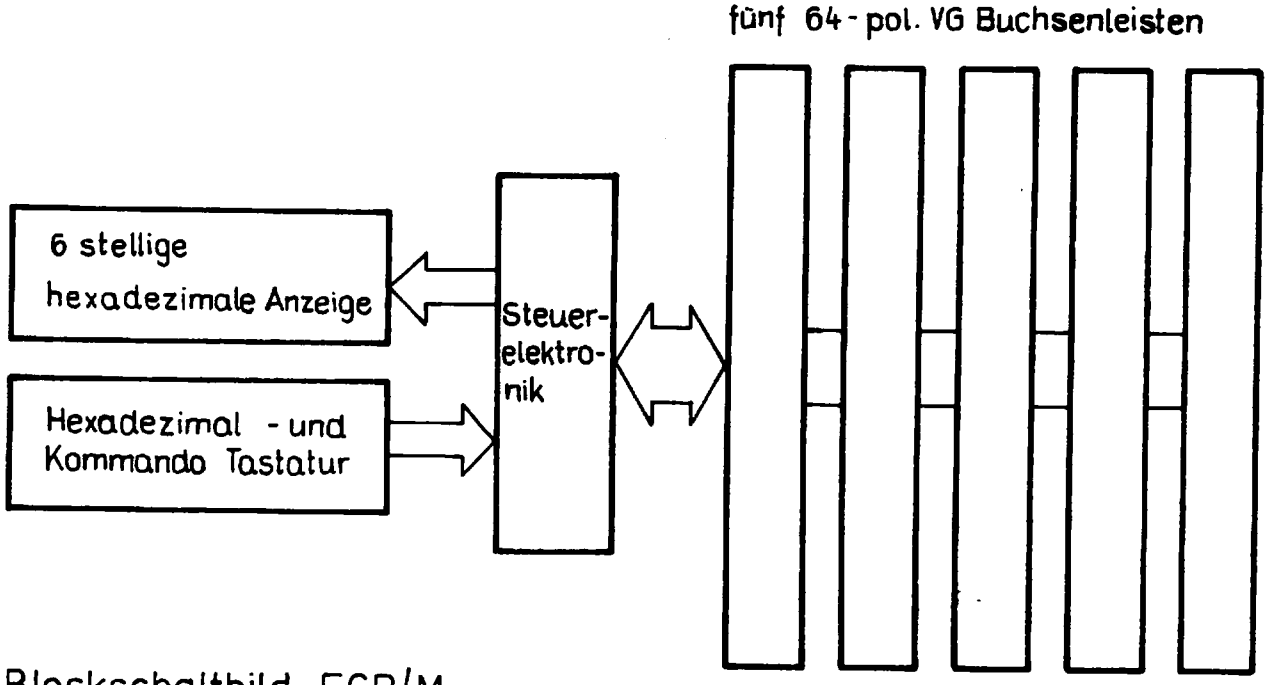
2. Schaltungsbeschreibung des Z80-KIT

2.1. Hardwareausstattung

Die serienmäßige Grundausstattung (Lieferumfang des Z80-KIT) umfaßt folgende Systemkomponenten:

2.1.1. Grundplatine ECB/M

- Eingabetastatur: 26 Tasten zur Bedienung des KIT
- Anzeigeeinheit: 4 hexadezimale Anzeigen für Adressen und 2 hexadezimale Anzeigen für Daten

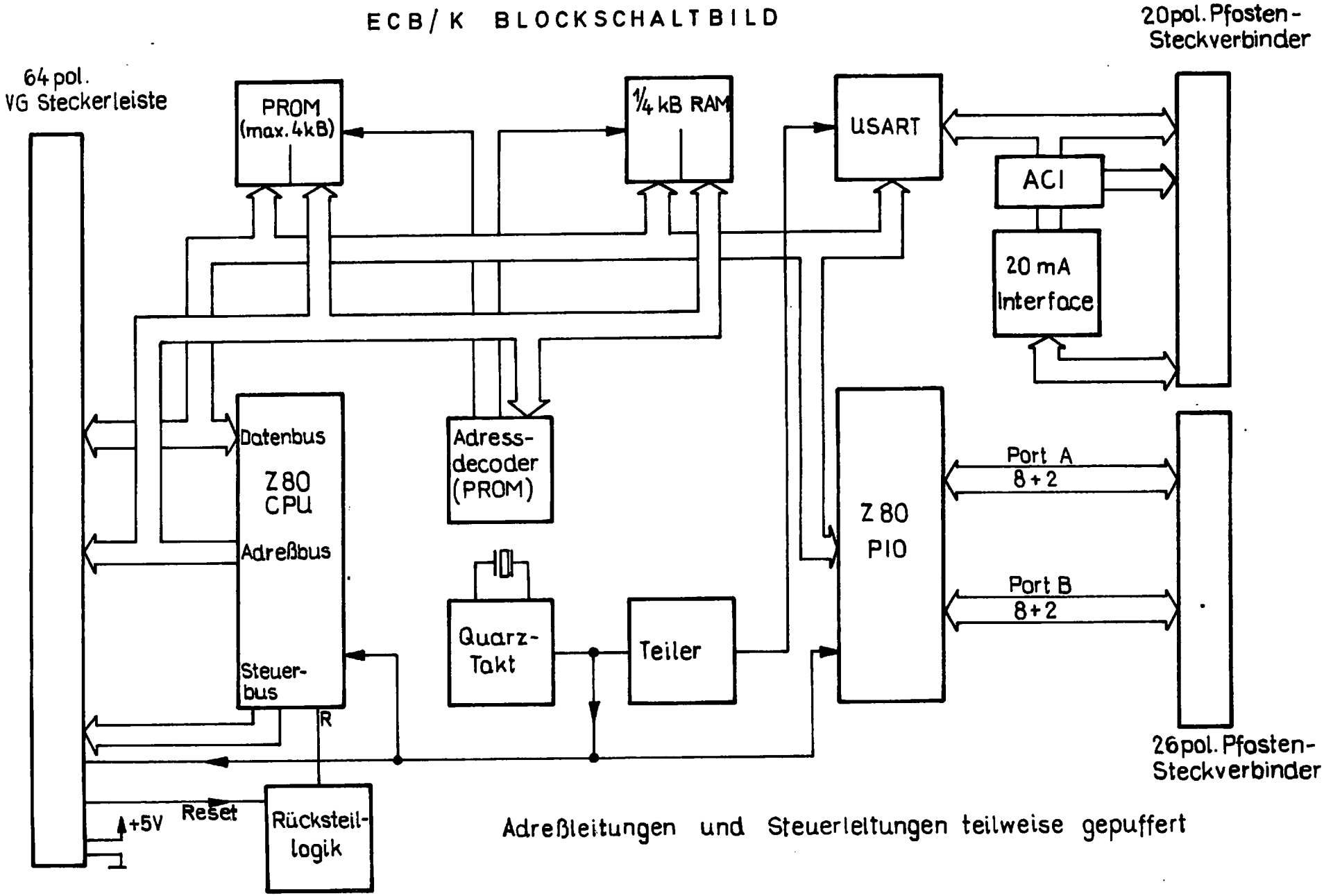


Blockschaltbild ECB/M

2.1.2. Einplatinencomputer ECB/K

- Mikroprozessor: 1 × Z80-CPU/PS
- Taktfrequenz: 2,5 MHz
- Programmspeicher: Max. 4 kByte, 1 kByte Betriebsprogramm serienmäßig mitgeliefert
- Schreib/Lesespeicher: 2 × 2112/A, Statische RAM's (= ¼ kByte)
- Ein/Ausgabeeinheiten: 1 × Z80-PIO 2 × 8 bit Programmierbare Parallelschnittstelle
  1 × 8251 oder 9551 USART Programmierbare Serien-Schnittstelle mit Audio-Cassetten-Interface (ACI) und vorbereiteter TTY-Schnittstelle

ECB/K BLOCKSCHALTBILD



Adreßleitungen und Steuerleitungen teilweise gepuffert

2.2. Funktionsbeschreibung

2.2.1. Grundplatine ECB/M

2.2.1.1. Das Tastenfeld

Im Sinne minimaler Hardwarekosten wurde von zwei Voraussetzungen ausgegangen:

- Verwendung billiger Standard-Drucktasten mit einem einzigen Arbeitskontakt
- Vermeidung des Einsatzes eigener Ein/Ausgabe-Bausteine und Mitbenützung der Adreßleitungen zu Ein-/Ausgabezwecken.

Ein Tastenfeld von max. 32 Tasten ist in 4 Reihen und 8 Spalten angeordnet. Die Reihen (= Eingänge) sind durch die, in einem Decoder decodierten Adressen gebildet. Die Spalten (= Ausgänge) sind an die Eingänge eines 3-State-Buffers angeschlossen. Der Zustand der Tasten wird mit dem Signal TRD (Tasten Read) über den Daten-Bus (D0 bis D7) in die CPU übertragen. Dabei ist TRD = (0F ∨ 0E ∨ 0D ∨ 0C) ∧ IORQ ∧ RD.

2.2.1.2. Die Anzeigeeinheiten

Bei der Konzeption der Low-Cost-Anzeige wurde von den gleichen Voraussetzungen ausgegangen. Die Anzeige umfaßt 6 ungepufferte Standard-7-Segment-LED's, die ohne Zwischenschaltung eines Hexadezimal 7-Segment-Decoders direkt per Tabelle aus dem Anwendungsprogramm angesteuert werden.

Dadurch können beliebige numerische, mit gewissen Restriktionen auch hexadezimale und alphanumerische Anzeigen durch Vorgabe einer geeigneten Tabelle im Anwenderprogramm ausgegeben werden.

Die 6 LED's werden kontinuierlich und zyklisch mit den anzuzeigenden Daten über den Datenbus versorgt. Die Anwahl der LED's erfolgt über einen Adreß-Decoder, der durch eine Stromsenke jeweils 1 LED ansteuert. Die Helligkeit der LED's läßt sich über die Zeitkonstante beeinflussen. Das Zeitglied erzeugt mit der Konstante ein WAIT-Signal, das die Dauer der Ausgabe entsprechend dehnt.

2.2.1.3. Interruptprioritätskette

Die mit den Z80-Bausteinen mögliche Prioritätsverteilung per „Daisy-Chain", kann selbstverständlich auch über mehrere Platinen hinweg aufgebaut werden. Dazu ist es notwendig, jeweils den IEO (Interrupt Enable OUT) einer Karte (PIN 16C-STA) mit dem IEI (Interrupt Enable IN) der nachfolgenden Karte (PIN 11C-STA) zu verbinden.
Die fünf Steckplätze des Z80-KIT sind bereits in dieser Weise miteinander verbunden, wobei Steckplatz 5 die höchste Priorität innehat. Soll ein nicht belegter Steckplatz überbrückt werden, so ist dies mit Hilfe einer vorgesehenen Lötbrücke möglich (siehe Schaltplan ECB/M).

2.2.1.4. Mechanischer Umbau der Grundplatine

Die Grundplatine des Z80-KIT ist so aufgebaut, daß der System-Bus vom Tasten- und Anzeige-Block getrennt werden kann. Dadurch ist der Einbau in einen 19''-Einschub möglich, während die Bedienungs- und Anzeigeeinheit in einem getrennten Gehäuse untergebracht werden kann.
Für eine Trennung ist lediglich ein Schnitt in Höhe der unteren Befestigungslöcher der VG Buchsenleisten nötig. Bei Einbau des BUS in einen 19''-Einschubrahmen muß die Platine auf beiden Seiten, entlang der Masse- bzw. +5 V-Fläche abgesägt werden. Die Befestigung im Einschub erfolgt ausschließlich über die VG Buchsenleisten!
Die elektrische Verbindung der geteilten Platine wird am besten über Flachbandkabel hergestellt. Bei Kabellängen über 20 cm empfiehlt es sich zur Abschirmung zwischen jeder Signalleitung eine Masseleitung zu führen.

2.2.2. Einplatinencomputer ECB/K

Die Baugruppe umfaßt folgende Funktionseinheiten:

- Z80-CPU
- Decoder und Puffer für Speicher- und I/O-Ausbau
- Bis zu 4 kByte PROM
- 256 Byte RAM
- 2 Parallelschnittstellen (1 Stück Z80-PIO)
- 1 Serienschnittstelle mit ACI- und TTY-Interface
- Busseitig 64-poliger Stecker nach DIN 41612 (VG 95324)
- I/O-seitig 3M-WWP-Pfostenverbinder.

Aus den Schaltplänen ist die Verknüpfung der einzelnen Funktionseinheiten zu erkennen. Die Busleitungen sind teilweise gepuffert. Die Adreßeingänge der auf der Z80-ECB/K untergebrachten Speicherbausteine werden über diese Puffer, die Datenleitungen vom CPU-Datenbus direkt angesprochen.

Festwert- und Schreib/Lese-Speicherbereich auf der Platine haben durch den Steuerbaustein CIC III vorgegebene Anfangsadressen (siehe Kap. 2.3.2.1. ff).

2.2.2.1. CPU-RESET

Das Rückstellen der Z80-CPU kann über die zwei Standardgatter sowohl durch Aus/Einschalten der Stromversorgung als auch elektronisch oder elektromechanisch über den Eingang RESET (1 Low-Power-Schottky-Eingangslast) erfolgen.

2.2.2.2. Takterzeugung

Aus Stabilitäts- und Kostengründen wird der Systemtakt über einen 9,8304 MHz-Quarz erzeugt, der zusammen mit 1/3 Standard LS-Baustein einen Oszillator bildet. Der für die Z80-CPU nötige Takt Φ von 2,4576 MHz wird über einen D-FlipFlop-Teiler (2 × 1/2 74LS74) gewonnen, von dem auch das 2Φ -Signal abgeleitet und auf den Bus herausgeführt wird.

2.2.2.3. Programmspeicher

Auf der ECB/K sind zwei PROM-Sockel (A2 und A3) für bis zu 4 kByte PROM vorhanden.
Folgende PROM-Typen können verwendet werden:

a)	HARRIS HM 7641	512 × 8 bit organisiert bipolar Bild 13, Kap. 5
b)	HARRIS HM 7681	1 k × 8 bit organisiert bipolar Bild 13, Kap. 5
c)	2708	1 k × 8 bit organisiert E-PROM. (−5 V, +12 V!), Bild 14, Kap. 5
d)	2758	1 k × 8 bit organisiert (single 5 V E-PROM), Bild 15, Kap. 5
e)	2716/6716/2516	2 k × 8 bit organisiert (single 5 V E-PROM) Bild 15, Kap. 5

Eine Einstellung auf einen der genannten Typen erfolgt durch Setzen der Jumper J1 . . . J4 entsprechend den Bildern 13 bis 15 in Kapitel 5 (Zusammenbau). Beachten Sie bitte, daß ein jeweils passender Steuerbaustein CIC III notwendig ist (siehe 2.2.1.). Die Standardausrüstung enthält einen Steuerbaustein CIC III passend zu 2 kByte umfassenden Festwertspeichern.

2.2.2.4. Schreib/Lese-Speicher

Der 256 Byte große Schreib/Lese-Speicher besteht aus 2 statischen 2112/A-RAM-Bausteinen, die 256 × 4-bitweise organisiert sind.

2.2.2.5. Parallel-Ein/Ausgabe

Für den Datenverkehr zwischen Z80-CPU und der „Außenwelt" über parallele Schnittstellen wurde 1 Baustein Z80-PIO auf der Baugruppe Z80-ECB/K untergebracht. Dieser Baustein umfaßt zwei 8bit-Ports, die als Eingabe-Port, Ausgabe Port, bidirektionales Port oder aber in Einzelbit-Schaltung arbeiten können (Einzelheiten siehe Z80-PIO-Produktspezifikation und Z80-PIO-Technical Manual). Beide Ports verfügen über eine interne Logik, die ohne zusätzliche Hardware Quittungs-(="Handshaking")-Betrieb (2 eigene Leitungen pro Port) und priorisierten Vektor-Interrupt ermöglicht. Die die Priorisierung festlegenden Signale IEI (="Interrupt Enable In") und IEO (="Interrupt Enable Out") sind zur Verkettung mit weiteren Ein/Ausgabebausteinen busseitig herausgeführt.

2.2.2.6. Serielle Ein/Ausgabe

Die Serienschnittstelle (1 Duplex-Kanal) ermöglicht den seriellen Datenverkehr zwischen peripherer Elektronik und der Z80-CPU.
Die Serienschnittstelle wurde mit einem Standard-USART-Baustein realisiert, da die Verwendung des Serien-Ein/Ausgabe-Bausteins Z80-SIO, der über 2 Duplex-Kanäle zur Arbeit mit SDLC- und ähnlichen Prozeduren und Logik zur Floppy-Disk-Steuerung verfügt, an dieser Stelle redundant wäre.
Der USART ist übrigens selbst auch interruptfähig, hat aber nicht die Möglichkeit zur automatischen Interrupt-Vektor-Behandlung, die alle Zilog Z80-Ein/Ausgabe-Bausteine aufweisen. Soll die Serienschnittstelle in Vektor-Interrupt-Betriebsart gefahren werden, müßte deshalb eine zusätzliche Logik zur Interrupt-Vektor-Erzeugung aufgebaut werden.
Angeschlossen an diese Serienschnittstelle sind ein Audio Cassetten Interface und ein TTY-Interface. Der Einfachheit halber liegen beide Schaltungen parallel; es ist aber jeweils nur eine von beiden benutzbar!

Das Audio Cassetteninterface

Es besteht aus einer einfachen Modulation/Demodulation. Die Ausgangsschaltung wandelt ein high-Signal in eine Frequenz um, ein low Signal hat keine Wirkung, so daß sich eine Folge von Frequenz/keine Frequenz ergibt. Der Ausgang wird direkt mit dem NF Eingang eines Cassettenrecorders verbunden.
Bei der Eingangsschaltung wird das vom Band gelieferte Signal demoduliert und damit wieder in logische low bzw. high Pegel umgesetzt.

Der Anschluß dieser Schaltung erfolgt an den Lautsprecheranschluß des Cassettengerätes. Beachten Sie dabei den Ausgangswiderstand des Gerätes, der in der Größenordnung 10 Ω liegen muß! Sollte dies nicht der Fall sein, muß R 29 entsprechend angepaßt werden. Auf diese Weise ist auch der Anschluß an eine Ohrhörerbuchse möglich.

Das TTY Interface

Beachten Sie bitte, daß es sich hier nur um eine **vorbereitete** TTY Schnittstelle handelt, da im Monitor des KIT keine softwaremäßige Steuerung enthalten ist. Ein entsprechendes Programm kann selbstverständlich jederzeit durch Verwendung von Programmspeichererweiterungen zusätzlich eingebaut werden (siehe Z80-KIT/PDT).

Die Eingangsschaltung der Stromschleifenschnittstelle (TTY-Schnittstelle) besteht aus einem Optokoppler mit nachfolgendem Impedanzwandler und TTL-Puffer. Die Leuchtdiode des Optokopplers kann in zwei verschiedene Arten vom angeschlossenen Terminal angesteuert werden:

- ☐ Liefert das Terminal den Strom für die Stromschleifenschnittstelle, genügt es, die Leuchtdiode in diesen Stromkreis zu schalten.

- ☐ Stellt das Terminal nur einen Kontaktschluß zur Verfügung, wie es üblicherweise bei einem Fernschreiber (z. B. Teletype ASR 33) der Fall ist, ist es notwendig, die Anode der Leuchtdiode über einen Widerstand (150 Ohm) auf + 5 Volt zu legen und den Kontakt zwischen Leuchtdioden-Kathode und Masse zu legen.

Der Fototransistor auf der Ausgangsseite des Optokopplers ist als Emitterfolger geschaltet, der einen weiteren Transistor schaltet. Die Basis des Fototransistors ist hochohmig mit dem Emitter verbunden, so daß der Fototransistor beim Ein- und Ausschalten an seinen Anschlüssen keine große Spannungsdifferenzen auszugleichen hat. Dadurch wird der Einfluß interner Transistorkapazitäten ausgeglichen und eine stabile Datenübertragung selbst bei hohen Baudraten gewährleistet.

Der dem Treibertransistor folgende TTL-Puffer 74LS04 ist zusätzlich auf den Pfostensteckverbinder der Serienschnittstelle zur Realisierung einer Spannungsschnittstelle entsprechend RS 232 herausgeführt, wobei allerdings die Normpegel nicht im vollen Umfang eingehalten werden. Dies hat jedoch für Datenübertragungen auf Strecken < 20 m praktisch keine Bedeutung.

Zur besonderen Beachtung: Bei Anschluß eines RS 232/V 2 SENDERS mit vollem Spannungshub an die Serienschnittstelle der ECB/K muß eine Zener Diode 4.7 V als Schutz parallel zum Eingang geschaltet werden.
Analog ist auch der Senderausgang des USART 8251 über einen TTL-Puffer und einen PNP-Transistor geschaltet, wobei der Ausgang des TTL-Puffers wieder über eine separate Leitung als RS 232 ähnliche Schnittstelle auf den Serieninterface-Steckverbinder herausgeführt ist. Der PNP-Transistor prägt den Strom der Datenübertragungs-Stromschleife.

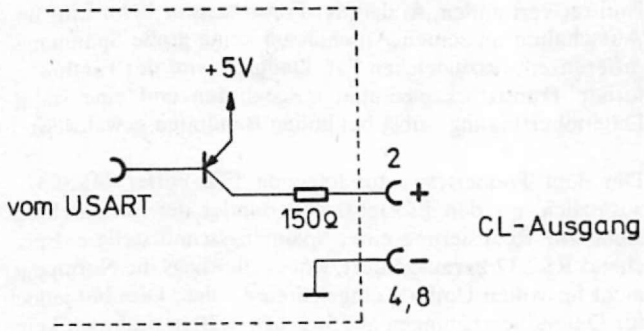
Die restlichen Ausgänge des 8251 sind ungepuffert auf den Serienstecker herausgeführt.

Der Clear to Send Eingang des USART verhindert das Aussenden von Daten, wenn er auf HIGH steht.

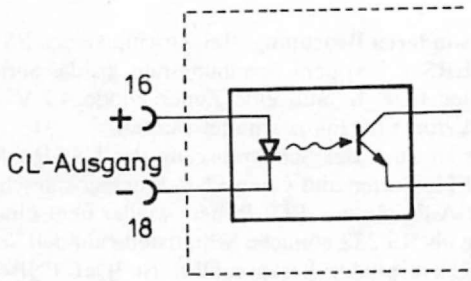
Da dieser Eingang auf der ECB/K nicht abgeschlossen ist empfiehlt es sich, auf dem Anschlußstecker eine Brücke zwischen CTS (PIN 15 Pfostenstecker) und GND (PIN 1 Pfostenstecker) zu legen.

2.1.6.1 Current Loop 20 mA (CL)

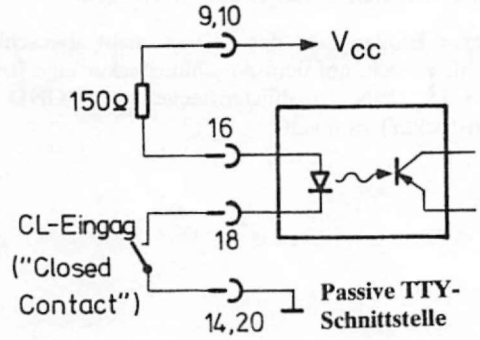
Der CL-Ausgang besteht aus einem PNP-Transistor, der über einen 150 Ω-Widerstand den notwendigen Strom liefert.



Der CL-Eingang besteht aus einem Optokoppler mit nachgeschaltetem Verstärker.



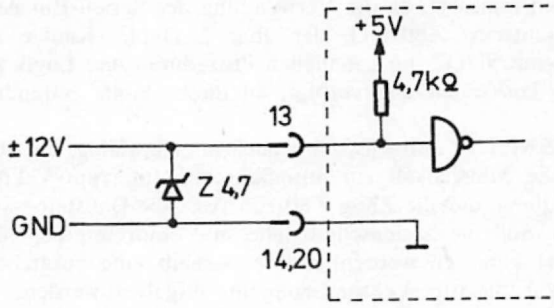
Dieser Eingang ist dafür eingerichtet, daß der Sender den Strom für die Schnittstelle liefert. Ist dies nicht möglich, muß im Verbindungsstecker ein Widerstand eingebaut werden:



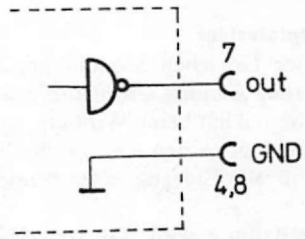
Der CL-Sender liefert dann lediglich ein „closed contact"-Signal, wobei der Strom für die Leuchtdiode über den 150 Ω-Widerstand aus den +5 V der Zentraleinheit erzeugt wird.

2.1.6.2 RS 232 C/V 24 Schnittstelle

Der RS 232 C Eingang der SST ist ein Low Power Schottky TTL Eingang. Um diesen mit echten RS 232 C Signalen steuern zu können, ist im allgemeinen nur eine Zenerdiode im Verbindungsstecker notwendig.



Der RS 232 C Ausgang ist ein Ausgang eines Low Power Schottky Gatters. Dieses liefert zwar die richtigen logischen Signale, nicht jedoch die Normpegel der RS 232 C Norm.



Hat ein anzuschließender RS 232 C Leitungsempfänger eine Hysterese ≥2 V, ist ein Interface zwischenzuschalten.

Taktterzeugung für ACI und TTY

Zur Erzeugung der Übertragungsfrequenzen der seriellen Schnittstelle (USART) dient eine vom Systemtakt gespeiste, vierstufige Teilerkette, deren einzelne Frequenzen durch Setzen einiger Jumper beliebig abgegriffen werden können.
Die Kette hat folgende Teilverhältnisse:

Stufe 1: festverdrahteter 16 Teiler (E4)
Stufe 2: 8 Teiler (D5a) wobei die Teilverhältnisse 2, 4 und 8 möglich sind
Stufe 3: programmierbarer Teiler (E5) mit den Teilverhältnissen 1, 2 . . . 16
Stufe 4: festverdrahteter 2 Teiler(D5b) zur Erzeugung eines Tastverhältnisses von 1

Stufe 1 erzeugt aus dem Systemtakt von
2,4576 MHz
eine Frequenz von 153,6 kHz.
Dieser Wert liegt am Eingang des 8 Teilers (= Stufe 2) an, kann aber auch direkt abgegriffen werden (Punkt B).
Die Ausgänge des Binärzählers sind
Q_B, Q_C und Q_D
und stellen die Frequenzen
76,8 kHz
38,4 kHz und
19,2 kHz zur Verfügung.

Mit Jumper J5 kann eine der vier Frequenzen an den Eingang der dritten Stufe (Punkt CUP) weitergeleitet werden. Stufe 3 besteht aus einem programmierbaren Teiler, der als Aufwärtszähler arbeitet.
Mit den Jumpern J6, J7, J8 und J9 kann ein entsprechendes Teilerverhältnis derart eingestellt werden, daß man das Komplement des gewünschten Teilfaktors nachbildet (z. B. soll durch 11 geteilt werden, ist eine 4 nachzubilden; siehe Standardverdrahtung).

Mit Jumper J10 wird die in Stufe 3 erzeugte Frequenz je nach Größe des Teilerverhältnisses von Punkt Q_A, Q_B, Q_C oder Q_D abgegriffen und zur Erzeugung eines geraden Tastverhältnisses mit Stufe 4 (Punkt A) verbunden. Der Ausgang dieses 2 Teilers ist direkt mit den Takteingängen R×C und T×C des USART verbunden. Durch die im KIT-Monitorprogramm vorgenommene MODE-Einstellung des USART wird noch eine zusätzliche Teilung durch 16 im USART erreicht. Die endgültige Übertragungsfrequenz berechnet sich demnach wie folgt:

$$f_{\ddot{u}} = 2{,}4576\ \text{MHz} \cdot \frac{1}{16} \cdot \frac{1}{X_1} \cdot \frac{1}{X_2} \cdot \frac{1}{2} \cdot \frac{1}{16}$$

$f_{\ddot{u}}$ = Übertragungsfrequenz
X_2 = Teilverhältnis der Stufe 2
X_3 = Teilverhältnis der Stufe 3

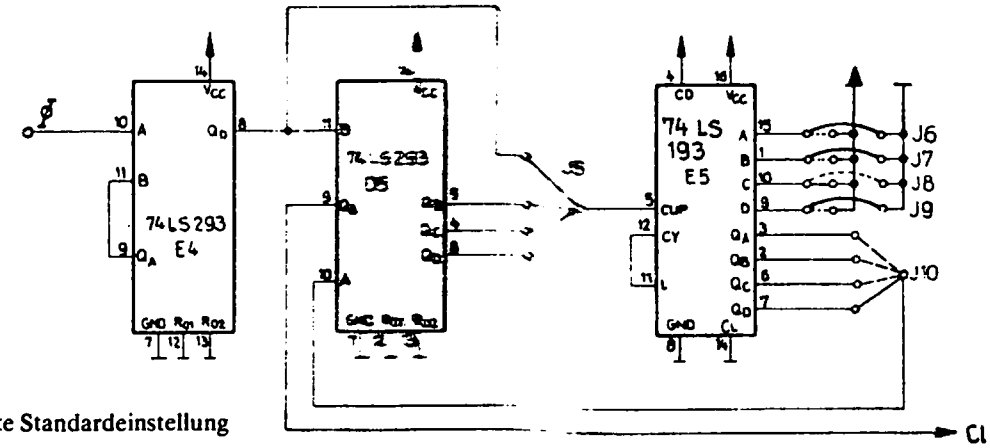
Beispiel: In der Standardeinstellung wird eine Übertragungsrate von 110 Baud (≙ 110 Hz) wie folgt festgelegt:

Eingangsfrequenz ≙ Systemtakt	≙ 2,4576 MHz
Stufe 1 16 Teiler	≙ 153,6 kHz
Stufe 2 4 Teiler	≙ 38,4 kHz
Stufe 3 11 Teiler	≙ 3,4909 kHz
Stufe 4 2 Teiler	≙ 1,7454 kHz
USART intern 16 Teiler	≙ 0,109 kHz ≙ 110 Hz

oder.

$$f_{\ddot{u}} = 2{,}4576\ \text{MHz} \cdot \frac{1}{16} \cdot \frac{1}{4} \cdot \frac{1}{11} \cdot \frac{1}{2} \cdot \frac{1}{16} \approx 110\ \text{Hz}$$

Sollen andere Übertragungsraten eingestellt werden, so ist mit Hilfe der obengenannten Formel zuerst der Ausdruck $\frac{1}{X_1} \cdot \frac{1}{X_2}$ zu ermitteln und dann eine entsprechende Verdrahtung zu wählen.



Teilerkette Standardeinstellung

Spezielle Baudrateneinstellung:

Jumper / Baudrate	J5	J6	J7	J8	J9	J10
4800	B → CUP	A → GND	B → +5 V	C → +5 V	D → +5 V	Q_A → A
2400	B → CUP	A → GND	B → +5 V	C → +5 V	D → +5 V	Q_A → A
1200 a)	Q_B → CUP	A → +5 V	B → GND	C → +5 V	D → +5 V	Q_A → A
1200 b)	Q_C → CUP	A → GND	B → +5 V	C → +5 V	D → +5 V	Q_A → A
600	Q_B → CUP	A → +5 V	B → +5 V	C → GND	D → +5 V	Q_B → A
300	Q_B → CUP	A → +5 V	B → +5 V	C → +5 V	D → GND	Q_C → A
150	Q_C → CUP	A → +5 V	B → +5 V	C → +5 V	D → GND	Q_D → A
110	Q_C → CUP	A → GND	B → +5 V	C → +5 V	D → GND	Q_D → A
75	Q_D → CUP	A → +5 V	B → +5 V	C → +5 V	D → GND	Q_D → A
50	Q_D → CUP	A → +5 V	B → +5 V	C → GND	D → GND	Q_D → A

2.3. Adressen-Belegung

2.3.1. Grundplatine ECB/M

2.3.1.1. Portadressen der Anzeigeeinheiten

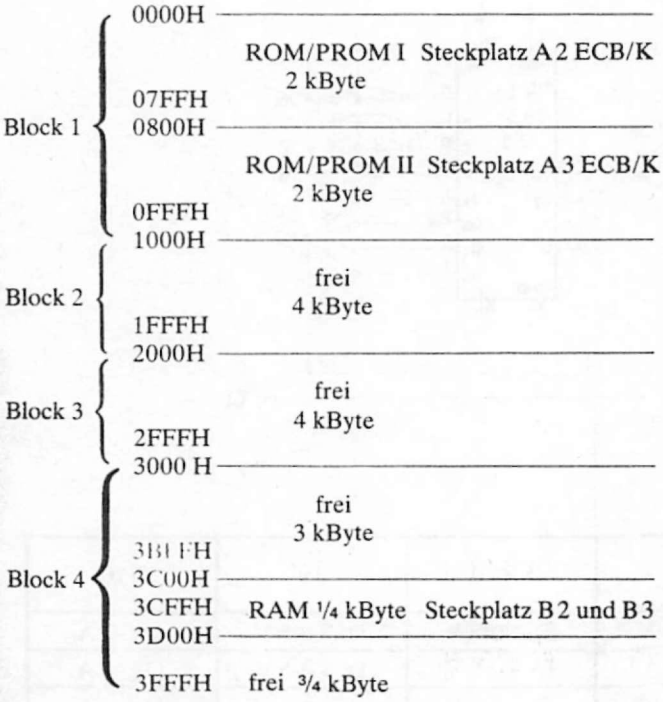
Den Anzeigeeinheiten auf der KIT Grundplatine sind folgende Port-Adressen fest zugeordnet:

DATA LED 1	(rechts)	0CH
DATA LED 2	(links)	0DH
ADDRESS LED 1	(rechts)	0EH
ADDRESS LED 1	(2.v.r.)	0FH
ADDRESS LED 2	(2.v.l.)	1CH
ADDRESS LED 4	(links)	1DH
ERROR LED		1FH

2.3.2. Einplatinencomputer ECB/K

2.3.2.1. Speicher-Adressen

Der Speicher des ECB-Systems ist standardmäßig in 4 kByte Blöcke (Seiten) unterteilt. Die auf der ECB/K liegenden PROM- bzw. RAM-Bereiche haben in der ausgelieferten Standardversion folgende Adressen:



2.3.2.2. Änderung der Speicheranfangsadressen

Die Standard-Verteilung kann selbstverständlich jederzeit geändert werden, indem man einen neuen Steuerbaustein CIC III mit entsprechender Programmierung einsetzt.

Beispiele:

a) CIC III ≙ PROM HM 7611 zur Adressedecodierung bei Einsatz von z. B. i2716 oder Z6716 Programmspeicherbausteinen: (standardmäßig ausgelieferter PROM-Inhalt; bei anderer Adreßbelegung ist ein PROM mit anderem Inhalt zu verwenden).

Adresse (hex)	Inhalt
0	3
1	3
2	3
3	3
4	3
5	3
6	3
7	3
8	5
9	5
A	5
B	5
C	5
D	5
E	5
F	5
3C	6

Anfangsadresse Steckplatz A2 ECB/K: 0000
Anfangsadresse Steckplatz A3 ECB/K: 0800
Anfangsadresse Steckplatz B2/B3 ECB/K: 3C00

b) PROM HM 7611 zur Adreßdecodierung bei Einsatz von HM 7641 Programmspeicherbausteinen:

Adresse (hex)	Inhalt
0	3
1	3
2	5
3	5
3C	6

Anfangsadresse Steckplatz A2 ECB/K: 0000
Anfangsadresse Steckplatz A3 ECB/K: 0200
Anfangsadresse Steckplatz B2/B3 ECB/K: 3C00

c) PROM HM 7611 zur Adreßdecodierung bei Einsatz von HM 7681-oder i2758 Programmspeicherbausteinen:

Adresse (hex)	Inhalt
0	3
1	3
2	3
3	3
4	5
5	5
6	5
7	5
3C	6

Anfangsadresse Steckplatz A 2 ECB/K: 0000
Anfangsadresse Steckplatz A 3 ECB/K: 0400
Anfangsadresse Steckplatz B 2/B 3 ECB/K: 3C00

NB.: Alle anderen Steuer-PROM-Speicherzellen müssen mit 1111 (F_{Hex}) belegt sein! Selbstverständlich sind auch Umadressierungen in Bezug auf das RAM möglich.

2.3.2.3. Ein/Ausgabe-Adressen

Die Ein/Ausgabe-Bausteine auf dem Z80-ECB/K sind im 1 aus 8 Code der Speicheradreß-Bits 0–3 dekodiert.

Z80-PIO:

PORT B/A	A0: LOW = A;	HI = B
CONTROL DATA	A1: LOW = DATA;	HI = CONTROL
CHIP SEL A2:	LOW = CHIP SELECTED	

USART:

CONTROL/DATA	A1: LOW = DATA;	HI = CONTROL
CHIP SEL	A3: LOW = CHIP SELECTED	

Die Adressen für die Ein/Ausgabe-Bausteine ergeben sich damit wie folgt:

PIO:	PORT B:	DATA	09H
	PORT B:	CONTROL	0BH
	PORT A:	DATA	08H
	PORT A:	CONTROL	0AH
USART:		DATA:	04H
		CONTROL:	06H

Es ist zu beachten, daß auf Grund der nicht vollständigen Dekodierung der Peripherbausteine auf der ECB/K bei Erweiterung um weitere I/O-Bausteine die Adreß-Bits 2 und 3 immer HI sein müssen! Grundsätzlich nicht erlaubt sind für weitere periphere Bausteine die I/O Adressen X0 . . . X3H, (X=0 . . . F), da dann entweder PIO oder USART, oder beide zusammen zusätzlich mit angesprochen werden.

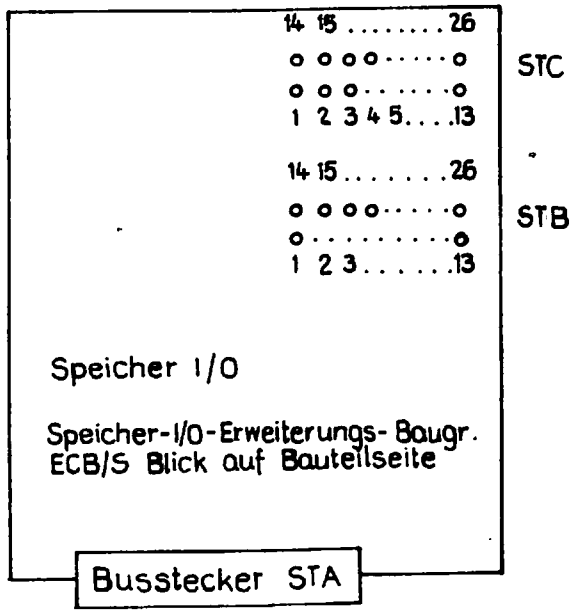
2.4. Pinbelegung

2.4.1. Grundplatine ECB/M

2.4.1.1. Ein/Ausgabe-Stecker STX

Da einige Bus-Leitungen nicht von den angesteckten Karten genützt werden, sind diese Leitungen auf einer 20poligen Stiftleiste herausgeführt.

Stift Nr.	Steckerfeld STX	20poliges WWP Steckerfeld
1	+5 V	
2	+5 V	
3	26 A	
4	10C	
5	12 A	
6	12 C	
7	+12 V	
8	+12 V	
9	−5 V	
10	−5 V	
11	19 A	
12	19 C	
13	21 A	
14	22 A	
15	23 A	
16	23 C	
17	24 A	
18	25 A	
19	GND	
20	GND	



2.4.2. Einplatinencomputer ECB/K

2.4.2.1. Ein/Ausgabestecker STB und STC

Stift-Nr.	Steckerfeld B	26 pol. WWP-Steckerfeld
1	GND	
2	A7	┐
3	A6	│
4	A5	│
5	A4	} PIO PORT A
6	A3	│
7	A2	│
8	A1	│
9	A0	┘
10	$\overline{ASTB}$	A Strobe
11	$\overline{BSTB}$	B Strobe
12	ARDY	A Ready
13	+5 V	
14	GND	
15	B7	┐
16	B6	│
17	B5	│
18	B4	} PIO
19	B3	} PORT B
20	B2	│
21	B1	│
22	B0	┘
23	NC	
24	NC	
25	BRDY B Ready	
26	+5 V	

Gegenstück: 26 pol. Pfosten-Verbinder
z. B. Scotch 3399—0000

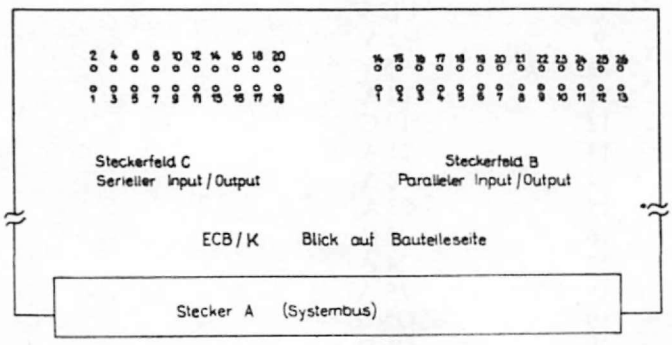
Stift-Nr.	Steckerfeld C	20 pol. WWP Steckerfeld
1	$\overline{DTR}$ (Data Terminal Ready; 8251)	
2	Current loop Out +	
3	$\overline{RTS}$ (Request to send Data; 8251)	
4	Current loop Out — (GND)	
5	$\overline{DSR}$ (Data Set Ready; 8251)	
6	ACI OUT	
7	RS 232 OUT	
8	GND	
9	+5 V	
10	+5 V	
11	ACI IN	
12	Tx EMPTY (Transmitter Empty; 8251)	
13	RS 232 IN +	
14	GND	
15	$\overline{CTS}$ (Clear to send Data; 8251)	
16	Current loop in +	
17	RxRDY (Receiver Ready; 8251)	
18	Current loop in —	
19	Tx Ready (Transmitter Ready; 8251)	
20	GND	

passendes Gegenstück: 20 pol. Pfosten Verbinder
z. B. Scotch 3421—0000

2.4.2.2. Bus-Stecker STA

64 poliger Steckverbinder VG 95 324

Benennung	Stecker Pin	Bezeichnung
A 0	5c	Adresse 0
A 1	7c	Adresse 1
A 2	6a	Adresse 2
A 3	6c	Adresse 3
A 4	7a	Adresse 4
A 5	8a	Adresse 5
A 6	9a	Adresse 6
A 7	9c	Adresse 7
A 8	8c	Adresse 8
A 9	30a	Adresse 9
A 10	18c	Adresse 10
A 11	17c	Adresse 11
A 12	27c	Adresse 12
A 13	29a	Adresse 13
A 14	18a	Adresse 14
A 15	28c	Adresse 15
D 0	2c	Data 0
D 1	14c	Data 1
D 2	4c	Data 2
D 3	4a	Data 3
D 4	5a	Data 4
D 5	2a	Data 5
D 6	3a	Data 6
D 7	3c	Data 7
$\overline{M1}$	20a	Maschinenzyklus 1
$\overline{MRQ}$	30c	Memory Request
$\overline{IORQ}$	27a	IN/OUT Request
$\overline{RD}$	24c	Read
$\overline{WR}$	22c	Write
$\overline{RFRSH}$	28a	Refresh
$\overline{HLT}$	25c	Halt
$\overline{WAIT}$	10a	Wait
$\overline{INT}$	21c	Interrrupt
$\overline{NMI}$	20c	non Mask. Int.
$\overline{RESET}$	31c	Reset
IEI 1	11c	Int. enable in
IEO 1	16c	Int. enable out
$\overline{PWRCL}$	26c	Power on clear
Φ	29c	Clock 2,45 MHz
+5	1a, c	
GND -	32a, c	
$\overline{BUSRQ}$	11a	Busrequest
$\overline{BUSAK}$	31a	Busacknowledge
(+12 V)	13a	
(—5)	15a	



3. Beschreibung des Z80-KIT Betriebsprogramms (Monitor)

3.1. Allgemeines

Für den Z80-KIT werden Betriebsprogramme im Umfang von 1 kByte und 2 kByte angeboten. Die 2 kByte Version (siehe KIT/TV) verfügt im Gegensatz zur 1k Version über eine zusätzliche Ansteuerungsmöglichkeit für ein TV-Gerät. Beide Betriebsprogramme werden als Firmware (also in einem Festwertspeicher) geliefert. Das PROM kann direkt in den dafür vorgesehenen Sockel A2 auf der ECB/K gesteckt werden.

Die Adreßraumaufteilung nach 2.3.2.1. weist dem Sockel A2 in der ausgelieferten Version einen Umfang von 2 kByte zu. Bei Verwendung des 1 k Betriebsprogramms bleibt der Adreßraum 0400H bis 07FFH (= 1 kByte) unbelegt.

Zusätzlich zum Betriebsprogramm kann auf dem Sockel A3 ein zusätzlicher Festwertspeicher von 2 kByte Kapazität untergebracht werden. (Siehe hierzu Teil V, KIT/PDT).

3.2. Aufbau des Betriebsprogramms

3.2.1. Unterprogrammstruktur

Der Z80-KIT-Monitor ist weitestgehend in Unterprogrammtechnik geschrieben. Das heißt, daß neben einem Rahmenprogramm mehrere in sich abgeschlossene Routinen existieren, die jeweils vom Rahmenprogramm her aufgerufen werden. Da die Routinen mit dem RETURN-Befehl (RET) abgeschlossen sind, ist ein Aufruf nicht nur vom Rahmenprogramm des Betriebsprogramms, sondern auch vom Anwenderprogramm her möglich. Dies bedeutet neben Platzersparnissen im Speicher eine wesentliche Erleichterung bei der Erstellung von Programmen, da die zur Verfügung stehenden Routinen nicht selbst programmiert werden müssen.

Das 2 kByte Betriebsprogramm unterscheidet sich von der 1 k Version im Wesentlichen durch erweiterte Routinen, wobei die Erweiterung jeweils zur Ansteuerung des TV-Gerätes dient.

3.2.2. Ein/Ausgabe von Werten

Werte die am KIT zur Anzeige kommen, stehen immer in den jeweils zugehörigen LED-Puffern, d.h. in Speicherzellen, auf die die Anzeigeroutine zyklisch zugreift.

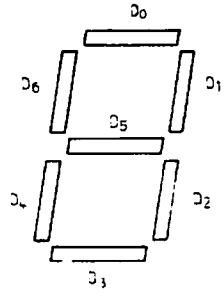
Soll ein Wert, der in einer bestimmten Speicherzelle steht angezeigt werden, so wird eine Routine aufgerufen, die den gewünschten Wert aus der Speicherzelle in die entsprechenden LED-Puffer transferiert.

Umgekehrt existiert eine zweite Routine, die einen, evtl. durch zuvoriges Eintippen in den LED-Puffern stehenden Wert in den Speicher überführt.

Sämtliche Anzeige- und Eingabe-Kommandos verlaufen nach diesem Prinzip.

3.2.3. Ausgabe nicht hexadezimaler Werte

Die Anzeigeroutine selbst verfügt über die Möglichkeit, die in den LED-Puffern stehenden Werte mit oder ohne vorherige Hex-zu-7-Segment-Konversion anzuzeigen (siehe 2.2.1.2.). Dadurch können auch nichthexadezimale Zeichen dargestellt werden. Die Zuordnung zwischen Segment und Daten-bit für nicht hexadezimale Zeichen ist nachfolgend dargestellt.



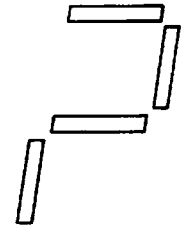
Bei Aufruf der Anzeigeroutine muß das HL Registerpaar die Adresse einer Speicherzelle (hier ‚MARKE' genannt) enthalten, deren Bit 0 entsprechend dem Wunsch, mit oder ohne Umrechnung, gesetzt sein muß.

Bit 0 = 1 → Anzeige mit Hex-zu-7 Segment Konversion

Bit 0 = 0 → Anzeige ohne Konversion

Beispiel für die Anzeige eines nichthexadezimalen Zeichens:

Darstellung des Zeichens:



(Fragezeichen stilisiert)

Auf „1" gesetzt sind die Bits D_0 D_1 D_5 und D_4. Dies ergibt binär:

0 0 1 1 0 0 1 1

oder hexadezimal

3 3

Schreibt man 3 3 in einen der LED PUFFER (Adressen siehe Kap. 3.5.) und ruft die Anzeigeroutine **ohne** Konversion auf, erscheint das obige Zeichen auf der entsprechenden Anzeigeeinheit.

3.2.4. Verwendung von Interrupts

Die Monitorprogramme verwenden für eigene Belange (STEP-Routine) den NMI (Non Maskable Interrupt). Dadurch ist der Einsatz des NMI durch den User ausgeschlossen.

Ebenfalls nicht benutzbar für Userzwecke sind MODE 0 und MODE 1 des maskable Interrupts, da sämtliche Restart-Adressen bereits vom Monitorprogramm belegt sind. Eine Interruptbehandlung durch das Userprogramm muß deshalb ausschließlich im MODE 2 erfolgen.

3.2.5. Besonderheit der Einschrittverarbeitung

Bedingt durch die Hardware der Grundplatine des Z80-KIT, treten bei der Abarbeitung von Befehlen mit Doppelanweisungsteil (Zwei M1 Zyklen) im STEP-Mode sog. „Zwischenschritte" auf. Dabei springt der Befehlszähler nicht wie erwartet auf die Adresse des nächsten Befehls, sondern auf 0068. Bei nochmaligem Drücken der Taste STEP erhält man den erwarteten Programmzählerstand. Diese „Zwischenschritte" haben keinerlei Auswirkungen auf das Userprogramm!

3.3. Mitbenutzbare Routinen des Betriebsprogramms

Im Anschluß finden Sie eine Zusammenstellung der Routinen des 1k Betriebsprogramms. Eine Zusammenstellung des 2k Betriebsprogramms finden Sie in Teil VI, KIT/TV. Neben einer Beschreibung der einzelnen Aktionen werden die jeweils von der Routine benutzten Register und die Einsprungadresse angegeben. Der Aufruf einer Routine erfolgt dann nach folgendem Schema:

Anwenderprogramm
evtl. Retten der Register
Aufruf Monitorroutine
evtl. Rückholen der Register
Anwenderprogramm Fortsetzung

Am Schluß dieser Aufstellung wird dieses Schema an einem konkreten Beispiel erläutert.

3.3.1. Zusammenstellung

OBTAST
Prüft, ob eine Taste des KIT Tastenfeldes gedrückt wurde. Bei der Rückkehr sind zwei Fälle möglich:
a) eine Taste wurde gedrückt
   ZERO FLAG zurückgesetzt
b) kein Tastendruck
   ZERO FLAG gesetzt

Keine Übergabeparameter
Benützte Register: A B C D E F H L IX
Einsprungpunkt: 0251

WETAST
Prüft um welche Taste es sich handelt. Bei der Rückkehr sind zwei Fälle möglich:
a) es war immer noch die gleiche Taste gedrückt (während eines Tastendruckes mehrere Abfragen, da Prozessor sehr schnell). Das A Register enthält den Wert FF
b) neue Taste erkannt. Das A Register enthält den Tastencode der entsprechenden Taste (siehe Schaltplan!)

Keine Übergabeparameter
Benützte Register: A B C D E F H L IX
Einsprungpunkt: 0287

LEDS1
Bringt die, in den zur DATA Anzeige gehörenden LED Puffern stehenden Werte bei DATA zur Anzeige.
Adresse der ‚Marke' beim Aufruf im HL Register-Paar.
a) bit 0 von Marke gleich 0 → Anzeige ohne Konversion
b) bit 0 von Marke gleich 1 → Anzeige mit Konversion
Benützte Register: A B C D E F H L IX IY
Einsprungpunkt: 0309

LEDS2
Bringt die, in den zur Adreß-Anzeige gehörenden LED Puffern stehenden Werte bei Adreß zur Anzeige
Adresse der ‚Marke' beim Aufruf im HL Register-Paar
a) bit 0 von Marke gleich 0 → Anzeige ohne Konversion
b) bit 0 von Marke gleich 1 → Anzeige mit Konversion
Benützte Register: A B C D E F H L IX IY
Einsprungpunkt: 0317

ZEITKO
Zeitschleife von ca. 0,5 Sec. Dauer.
Keine Übergabeparameter
Benützte Register: A B C F
Einsprungpunkt: 03CA

LOAD
Liest auf Kassette abgespeicherte Programme oder Daten ein.
Startadresse des Zielspeichers beim Aufruf im Speicher.
3C44/45 (siehe Kommando LOAD)
Benützte Register: A F H L
Einsprungpunkt: 0090

STORE
Speichert im RAM stehende Programme oder Daten auf Cassette ab.
Startadresse und Endadresse des Quellspeichers beim Aufruf im Speicher 3C40/41 und 3C42/43
(siehe Kommando STORE)
Benützte Register: A B C D E F H L
Einsprungpunkt: 00BC

ABSPE1
Entnimmt den beiden zur DATA Anzeige gehörenden LED Puffern die dort gespeicherten (und angezeigten) Halbworte (0, 1 . . . F), bildet ein Gesamtwort (00, 01 . . . FF) und speichert es in die gewünschte Speicherzelle ab
Adresse der gewünschten Speicherzelle beim Aufruf im IY Register
Benützte Register: A B F H L IX IY
Einsprungadresse: 0361

ABSPE2
Entnimmt den vier zur Adreß-Anzeige gehörenden LED Puffern die dort gespeicherten (und angezeigten) Halbworte, bildet zwei Gesamtworte (high und low Byte) und speichert sie in die gewünschte Speicherzelle (low Byte) bzw. die nächstfolgende (high Byte) ab
Niedrigere Adresse der beiden gewünschten Speicherzellen beim Aufruf im IY Register
Benützte Register: A B F H L IX IY
Einsprungadresse: 0369

RESPE1
Entnimmt dem gewünschten Speicher den Wert, teilt dieses Gesamtwort auf in zwei Halbworte und belegt damit die beiden zur DATA Anzeige gehörenden LED Puffer
Adresse der gewünschten Speicherzelle beim Aufruf im IY Register
Benützte Register: A B F H L IX IY
Einsprungadresse: 0386

RESPE2
Entnimmt dem gewünschten und dem nachfolgendem Speicher den Wert, teilt die beiden Gesamtworte (low und high Byte) jeweils auf in zwei Halbworte und belegt damit die vier zur Adreß Anzeige gehörenden LED Puffer
Niedrigere Adresse der beiden gewünschten Speicherzellen beim Aufruf im IY Register
Benützte Register: A B F H L IX IY
Einsprungpunkt: 038E

FEHLER
Bringt die ERROR Lampe zum Leuchten
Keine Übergabeparameter
Benützte Register: A F
Einsprungpunkt: 02FF

FEHLEX
Löscht die ERROR Anzeige
Keine Übergabeparameter
Benützte Register: A F
Einsprungpunkt: 0303

ANZABF
Zusammengesetzte Routine aus LEDS1, LEDS2, OBTAST und WETAST; zeigt die in den LED Puffern gespeicherten Werte an und führt eine Tastenabfrage durch. Rücksprung bei Erkennen einer neuen Taste.
Adresse der ‚Marke' im HL Registerpaar
Bit 0 siehe LEDS1 bzw. LEDS2
a) Bit 1 von Marke gleich 1 → DATA Anzeige
b) Bit 1 von Marke gleich 0 → keine DATA Anzeige
c) Bit 2 von Marke gleich 1 → ADDRESS Anzeige
d) Bit 2 von Marke gleich 0 → keine ADDRESS Anzeige
Benützte Register: A B C D E F H L IX IY
Einsprungpunkt: 0346

TAREG1
Verschiebt die Inhalte der zur DATA Anzeige gehörenden LED Puffer um eine Stelle in Richtung höherwertiger Stelle. Die dadurch freiwerdende unterste Stelle (DATA 1) wird mit einem neuen Wert belegt
Neu einzulesender Wert beim Aufruf im A-Register
Benützte Register: A B C D E F H L
Einsprungpunkt: 02CC

TAREG2
Verschiebt die, in den zur Addreß Anzeige gehörenden LED Puffern stehenden Werte in Richtung höherwertiger Stellen. Die freiwerdende unterste Stelle (Addreß1) wird mit einem neuen Wert belegt
Neu einzulesender Wert beim Aufruf im A-Register
Benützte Register: A B C D E F H L
Einsprungpunkt: 02E0

3.3.2. Beispiel:
In einem User Programm soll eine Tastenabfrage durchgeführt werden. Der Inhalt der Register H, L und IX soll erhalten bleiben. Während des Wartens auf die neue Taste soll der Inhalt der DATA LED Puffer angezeigt werden.

```
        ↓                   Userprogramm

        POSH HL   }
        PUSH IX   }         Retten der Register
ABFR:
        CALL 0309           Anzeige Inhalt DATA-LED-
                            Puffer bei DATA durch Aufruf
                            von LEDS1
        CALL 0251           Abfrage ob Tastendruck durch
                            Aufruf von OBTAST
        JR Z, ABFR          wenn keine Taste gedrückt
                            zurück!
        CALL 0287           wenn Taste gedrückt, Berech-
                            nung des Tastencodes durch
                            Aufruf von WETAST
        POP IX    }
        POP HL    }         Rückholen der Register

        CP 0      }
        JP Z,.... }
        CP 1      }         Userprogramm Fortsetzung:
        JP Z,.... }         Verzweigung abhängig
        CP 2      }         von der gedrückten Taste
                  }
        ↓         }
```

3.4. Einsprungpunkte des Betriebsprogramms

Für den Rücksprung vom Userprogramm ins Betriebsprogramm existieren vier verschiedene Möglichkeiten:

3.4.1. Zusammenstellung

a) **Einsprung bei 0000**
Das Betriebsprogramm durchläuft die Initialisierungsroutine; sämtliche Userwerte werden gelöscht.

b) **Einsprung bei 0148**
Das Betriebsprogramm durchläuft die Aktualisierungsroutine; sämtliche aktuellen Registerwerte des Userprogramms mit Ausnahme des Programmzählers und des HL Registerpaars werden in dem dafür vorgesehenen RAM Bereich abgelegt (siehe Kap. 3.5.).
NB: In den Speicherzellen 3CFE und 3CFF muß vor dem Einsprung 0B bzw. 00 stehen! HL und PC sind gesondert zu retten.

c) **Einsprung bei 000B**
Das Betriebsprogramm läuft ohne zusätzliche Funktion an. Keine Beeinflussung der Userwerte.

d) **Einsprung bei 0038**
Das Betriebsprogramm durchläuft die Fehlerroutine (ERROR Lampe an). Keine Beeinflussung der Userwerte.

3.4.2. Beispiel:

Nach Beendigung eines Userprogrammes soll ein Rücksprung ins Betriebsprogramm ohne zusätzliche Funktionen erfolgen:

```
  ↓                        Userprogramm

JP 000BH (C3 0B 00)        Rücksprung
```

3.5. Systemkonstante des 1 kByte-Betriebsprogramms

Zu 3.5. Systemkonstante

- Die Anzeige des Z80-KIT besteht aus 6 7-Segment Anzeigen. Jedes dieser Anzeigenelemente wird als eigener Ausgabe-Port angesprochen. Die Adressen dieser Ports sind im Betriebsprogramm in Tabellenform unter den angegebenen Adressen enthalten. Eine Mitbenutzung der Tabelle durch das Anwenderprogramm ist dadurch möglich.
  Um für den Z80-KIT ein „low cost display" aufzubauen, wurde auf die Verwendung von Hex-7-Segment-Decoder verzichtet.
  Die Ansteuerung erfolgt softwaremäßig. Den anzuzeigenden Zahlenwert speichert man zunächst in binärer Form. Die Umwandlung auf 7-Segment-Darstellung erfolgt über den Zugriff auf die 7-Segment-Tabelle SEGM, die zu jeder Zeile das entsprechende 7-Segment-Äquivalent enthält.
  Dabei wird im HL-Registerpaar die Anfangsadresse der SEGM-Tabelle (03DB) gespeichert. Dazu addiert man dann den oben erwähnten, anzuzeigenden Wert. Es ergibt sich eine Adresse, die eine Speicherzelle innerhalb der Tabelle addressiert. Der dort gespeicherte Wert wiederum erzeugt, auf die Anzeige gegeben, den darzustellenden Wert.
- Da es sich bei den Mikroprozessoren um sequentiell arbeitende Maschinen handelt, ist die Bearbeitung von jeweils nur einem Programm möglich. Das gleichzeitige Behandeln zweier verschiedener Programme ist ausgeschlossen. Daraus folgt, daß ein vom Anwender erstelltes Programm und das Betriebsprogramm nicht gleichzeitig sondern nur getrennt voneinander ausgeführt werden können.
  Der Wechsel vom Betriebsprogramm zum Anwenderprogramm erfolgt, nach Angabe des Anfangsbefehlszählerstandes, mittels der Taste START.
  Eine Rückkehr ins Betriebsprogramm nach vollendetem Anwenderprogramm kann durch einen „Sprung" auf einen der vier speziellen Einsprungpunkte des Betriebsprogramms erfolgen. Der Sprungbefehl ist letzter Befehl des Anwenderprogramms.
  Da bei einem Wechsel zwischen Anwender- und Betriebsprogramm die im Anwenderprogramm errechneten Werte nicht verlorengehen dürfen, müssen die jeweils aktuellen Registerinhalte des Anwenderprogramms gerettet werden. Diese Notwendigkeit wird speziell bei der Einzelbefehlabarbeitung deutlich, wo ein ständiger Wechsel zwischen Betriebs- und Anwenderprogramm durchgeführt wird.
  Um dieser Aufgabe gerecht zu werden, wird ein festgelegter Speicherbereich dazu benützt, die Anwender-Register-Werte während des Betriebsprogrammlaufes dort abzuspeichern. Dieser Bereich ist so organisiert, daß bestimmten Speicherplätzen bestimmte Register zugeordnet sind.
  Außerdem wenn Anfangswerte für ein Anwenderprogramm eingegeben werden (z.B. SET B FF EX), so wird die entsprechende Speicherzelle belegt, aus der der Wert bei START oder STEP in die CPU übernommen wird.
  Eine Aktualisierung der besagten Zelle bei Rückkehr ins Betriebsprogramm kann direkt durch einen im Anwenderprogramm stehenden Lade-Befehl, oder durch Benützung des Einsprungs bei 0148 erfolgen.
- Sämtliche zur Anzeige kommenden Werte müssen in den sog. LED-Puffern stehen, auf die die Anzeigenroutinen LEDS1 bzw. LEDS2 zugreifen. Die Puffer haben die unten angegebenen Adressen.

Adresse	Inhalt	Kommentar	
		PORTS:	
03D5	0C	DEFB 0CH	;DATA 1 LED
03D6	0D	DEFB 0DH	;DATA 2 LED
03D7	0E	DEFB 0EH	;ADR. 1 LED
03D8	0F	DEFB 0FH	;ADR. 2 LED
03D9	1C	DEFB 1CH	;ADR. 3 LED
03DA	1D	DEFB 1DH	;ADR. 4 LED
		SEGM:	;Hex zu Sieben-Segment Tabelle
03DB	5F	DEFB 05FH;	0
03DC	06	DEFB 06H;	1
03DD	3B	DEFB 3BH;	2
03DE	2F	DEFB 2FH;	3
03DF	66	DEFB 66H;	4
03E0	6D	DEFB 6DH;	5
03E1	7D	DEFB 7DH;	6
03E2	07	DEFB 07H;	7
03E3	7F	DEFB 7FH;	8
03E4	6F	DEFB 6FH;	9
03E5	77	DEFB 77H;	A
03E6	7C	DEFB 7CH;	B
03E7	59	DEFB 59H;	C
03E8	3E	DEFB 3EH;	D
03E9	79	DEFB 79H;	E
03EA	71	DEFB 71H;	F
		Userregister:	
3C00		SPC	Programmzähler
3C02		SSP	Stackpointer
3C04		SIY	iY Register
3C06		SIX	IX „
3C08		SR	R „
3C09		SI	I „
3C0A		SE0	E' „
3C0B		SD0	D' „
3C0C		SC0	C' „
3C0D		SB0	B' „
3C0E		SF0	F' „
3C0F		SA0	A' „
3C10		SL0	L' „
3C11		SH0	H' „
3C12		SE	E „
3C13		SD	D „
3C14		SC	C „
3C15		SB	B „
3C16		SF	F „
3C17		SA	A „
3C18		SL	L „
3C19		SH	H „
		LED Puffer:	
3C30		LED Puffer 1	DATA 1 Puffer
3C31	3C2F	LED Puffer 2	DATA 2 Puffer
3C32		LED Puffer 3	ADDRESS 1 Puffer
3C33		LED Puffer 4	ADDRESS 2 Puffer
3C34		LED Puffer 5	ADDRESS 3 Puffer
3C35		LED Puffer 6	ADDRESS 4 Puffer

4. Hinweise zur Bedienung Ihres Z80-KIT

4.1. Allgemeines

Vorbemerkung:
Sie sollen sich mit dem Betriebssystem des Z80-KIT vertraut machen, **bevor** Sie das Mikrocomputersystem in Betrieb nehmen (= an +5 V Versorgungsspannung anschließen).
Erst nach vollständigem Durcharbeiten dieser Druckschrift sollten Sie den Z80-KIT an die Versorgungsspannung anschließen.
Hinweise zu dieser endgültigen Inbetriebnahme finden Sie in Abschnitt 6.
Das Betriebsprogramm, das als Firmware im Baustein CIC I (Steckplatz A2) untergebracht ist, ermöglicht Ihnen die Benutzung Ihres Computers. Es liest, decodiert und verarbeitet die über die Tastatur eingetasteten Kommandos und Daten und ist für die Ansteuerung der Anzeigeelemente verantwortlich. Dadurch ist es möglich, die im Anschluß beschriebenen Aktivitäten des Computers auszulösen.
NB.:

- ☐ In allen Teilen des Programmes führen nur dafür vorgesehene Tasten zu einer Aktion. Alle anderen Tasten werden ignoriert und haben keine Wirkung.
- ☐ Richtige Eingaben des Benutzers haben die sofortige Ausführung des gewünschten Kommandos zur Folge.
- ☐ Bei der Eingabe von Daten werden die getippten Hex-Ziffern jeweils von rechts nach links in die Anzeige geschoben. Dieser Vorgang kann beliebig oft ausgeführt werden, wodurch schnelle Korrekturen möglich sind.
  Ein Abspeichern des Wertes (= Übernahme von den LED-Puffern in den Speicher) erfolgt erst nach Drücken der Taste EXECUTE (EX).
- ☐ Das Löschen der ERROR Lampe erfolgt über die Kommandos LOAD, SET oder DIS.

Achtung!
Die Monitorprogramme benötigen zur Zwischenspeicherung verschiedener Werte einen kleinen Teil des RAM Bereiches der ECB/K. Aus diesem Grund steht der Schreib-Lese Speicher dem Anwender nicht vollständig zur Verfügung.
Wie in Abschnitt 2.1.2.1. gezeigt, erstreckt sich der RAM-Bereich

von Adresse 3C00 bis Adresse 3CFF.

Für das Monitorprogramm werden davon folgende Bereiche für Zwischenspeicherungen in Anspruch genommen:

Adresse 3C00 bis Adresse 3C37 und
Adresse 3CF2 bis Adresse 3CFF

Für den Anwender steht demnach der Speicherbereich zwischen den Adressen

3C38 und 3CF1

zur Verfügung.

Das heißt, daß Ihre mit dem Z80-KIT — ohne zusätzliche Speicherplatinen — entwickelten Programme ausschließlich diesen Speicherbereich in Anspruch nehmen dürfen!!
Die Entwicklung längerer Programme und die extensive Verwendung eines Massenspeichers oder PROM-Programmiereinrichtung bedingt den Einsatz eines zusätzlichen Schreib/Lese-Speichers, wofür beim Z80-KIT durch Bus-Pufferung und das Vorhandensein zusätzlicher Steckplätze Vorsorge getragen ist.

4.2. Aufteilung der Anzeigeeinheiten

Die Anzeige besteht aus insgesamt sechs Anzeigebausteinen, von denen vier zur Anzeige von 16 Bits umfassenden Größen (Adressen) dienen, während die restlichen zwei Anzeigebausteine zur Anzeige von 8 Bits umfassenden Größen (Daten) eingerichtet sind.

4.3. Aufteilung des Tastenfeldes

Das Tastenfeld umfaßt 29 Tasten, die entsprechend ihrer Funktion in verschiedenen Farben gehalten sind:

— graue Tasten: DATEN-TASTEN
mit ihnen können Daten oder Adressen in hexadezimaler Form angegeben werden. Gleichzeitig ist mittels der Tasten mit Doppelbedeutung die Adressierung bestimmter CPU-Register möglich

— weiße Tasten: FUNKTIONSTASTEN
mit ihrer Hilfe werden Kommandos gegeben, die bestimmte Aktivitäten einleiten

— orange Tasten: FUNKTIONSTASTEN MIT BESONDERER BEDEUTUNG. Farbe, Lage und Größe dieser Tasten wurde zur Vermeidung von Fehlbedienungen in vorliegender Weise gewählt.

4.4. Bedeutung der Tasten

Vorbemerkung:
In diesem Abschnitt wird eine Übersicht über die Funktion der einzelnen Tastenelemente gegeben. Bitte versuchen Sie an dieser Stelle noch nicht, Ihr System in Betrieb zu nehmen; an späterer Stelle wird Inbetriebnahme, Programm-Eingabe- und -Ausführung noch eingehend behandelt.

Kommandotastatur

RESET Die Rücksetzungs (RESET) Taste gibt Ihnen die Möglichkeit, das System zu jeder Zeit in einen definierten Anfangszustand zu bringen. Sie ist nach jedem Einschalten der Speisespannung zu bestätigen.

EX Abschlußkommando (EXECUTE) veranlaßt das Mikrocomputersystem zur Übernahme und Abspeicherung des zuvor eingetippten, auf der Anzeige sichtbaren Wertes.

LOAD Ladeanweisung, dient zum Laden eines Programms, das auf Cassette gespeichert ist, in einen bestimmten Speicherbereich

STORE Speicheranweisung, dient zum Abspeichern eines bestimmten Speicherbereiches auf Band.

DIS Anzeigeanweisung (DISPLAY)
dient zur Ausgabe eines Speicher- oder Registerinhalts

SET Schreibanweisung
dient zur Eingabe eines Wertes in eine Speicherzelle bzw. in ein Register

MEM Speicheranweisung (MEMORY)
wird benötigt, wenn eine Speicheroperation durchgeführt werden soll (z.B. SET MEM . . .gibt an, daß eine Speicherzelle belegt werden soll)

START Startkommando
dient zum Starten eines Anwenderprogramms

STEP Einzelschrittkommando
bewirkt Ausführung eines Programmschritts (=CPU-Befehl)

BR Haltepunktanweisung (BREAKPOINT)
mit dieser Taste kann ein Haltepunkt auf eine gewünschte Adresse gelegt werden

IN Eingabekommando (INPUT)
versetzt den Mikrocomputer in den Eingabemodus für die Eingabe eines Anwenderprogramms

IDM Ausgabekommando (INCREMENT AND DISPLAY MEMORY)
dient zur zusammenhängenden Ausgabe größerer Speicherbereiche

STORN Aussprunganweisung (STORNO INPUT)
dient zum Verlassen des INPUT-Modus

Hexadezimaltastatur

0/′ Hexadezimalziffer 0 / Kennung der Zweitregister (z.B. H′ L′ etc.)

1 Hexadezimalziffer 1

2 Hexadezimalziffer 2

3/I Hexadezimalziffer 3 / I Register

4/P Hexadezimalziffer 4 / Befehlszähler (Programm Counter = PC Register)

5/S Hexadezimalziffer 5 / Stack Pointer (SP Register)

6/Y Hexadezimalziffer 6 / IY Register

7/X Hexadezimalziffer 7 / IX Register

8/H Hexadezimalziffer 8 / H Register

9/L Hexadezimalziffer 9 / L Register

A Hexadezimalziffer A / A Register

B Hexadezimalziffer B / B Register

C Hexadezimalziffer C / C Register

D Hexadezimalziffer D / D Register

E Hexadezimalziffer E / E Register

F Hexadezimalziffer F / F Register

4.5. Kommandosprache

4.5.1. Setzen von Registern auf einen gewünschten Wert

Dies wird durch folgende Tastenfolge erreicht:

SET register XXX . . . X EX (X = 0,1, . . . F)

Für „register" ist **die** Taste zu drücken, die dem gewünschten Register zugeordnet ist. Nach dem Drücken dieser Taste verlöschen die jeweils nicht benutzten Anzeigen, während auf den verbleibenden LED's der augenblickliche Wert dieses Registers angezeigt wird.

XXX . . . X ist eine beliebig lange Folge von Hex-Zeichen, die von rechts nach links in die entsprechende Anzeige geschoben wird.

Steht der gewünschte Wert auf der Anzeige, kann mit EX abgeschlossen werden. Der neue Wert wird dann übernommen und abgespeichert.

Beispiele:

Das IX Register (16 bit!) soll auf den Wert 2 OCA gesetzt werden.

SET 7/X 2 O/' C A EX

- 7/X: Der augenblickliche Wert des IX Registers bei ADDRESS angezeigt, die DATA-Anzeigen verlöschen.
- 2 O/' C A: Die Hex Ziffern werden von rechts nach links nacheinander in die Anzeige geschoben.
- EX: Der angezeigte Wert wird abgespeichert. Die normale Anzeige kehrt zurück.

Das 8 bit Register A' soll auf den Wert 78 gesetzt werden

SET 0/' A 8/H 7/X 8/H EX

- A: Bei DATA erscheint der augenblickliche Wert von A'
- 8/H 7/X: Durch einen Tippfehler wird 87 statt 78 eingegeeben.
- 8/H: Durch Nachtippen einer 8 wechselt die Anzeige auf den Wert 78.
- EX: Der angezeigte Wert wird abgespeichert. Die normale Anzeige kehrt zurück.

Zusammenstellung

SET:			
4/P	X	X . . . X	EX
5/S	X	X . . . X	EX
6/Y	X	X . . . X	EX
7/X	X	X . . . X	EX
8/H		X . . X	EX
9/L		X . . X	EX
A		X . . X	EX
B		X . . X	EX
C		X . . X	EX
D		X . . X	EX
E		X . . X	EX
F		X . . X	EX
I		X . . X	EX
0/' 9/L		X . . X	EX
0/' 8/H		X . . X	EX
0/' A		X . . X	EX
0/' B		X . . X	EX
0/' C		X . . X	EX
0/' D		X . . X	EX
0/' E		X . . X	EX
0/' F		X . . X	EX

X = Beliebiger Wert zwischen 0 und F

4.5.2. Anzeige von Registerinhalten (DISplay)

Dies wird durch folgende Tastenfolge erreicht:

DIS register

Für „register" ist wiederum die Taste zu drücken, die dem gewünschten Register zugeordnet ist. Der Inhalt des Registers erscheint an den Anzeigeelementen und zwar bei ADDRESS (4 Stellen), wenn es sich um ein 16 bit Register handelt (z.B. PC) bzw. bei DATA (2 Stellen), wenn es sich um ein 8 bit Register handelt (z.B. A Register)

Beispiele: Der Inhalt des 8 bit Registers H soll angezeigt werden

DIS 8/H

Das Ergebnis erscheint bei DATA (2-stellig!) Der Inhalt des I6 bit umfassenden Stack Pointers (SP-Register) soll angezeigt werden

DIS 5/S

Das Ergebnis erscheint bei ADDRESS (4-stellig)

Zusammenstellung

DIS:		angezeigt unter
4/P		ADDRESS
5/S		ADDRESS
6/Y		ADDRESS
7/X		ADDRESS
8/H		DATA
9/L		DATA
A		DATA
B		DATA
C		DATA
D		DATA
E		DATA
F		DATA
3/I		DATA
0/'	8/H	DATA
0/'	9/L	DATA
0/'	A	DATA
0/'	B	DATA
0/'	C	DATA
0/'	D	DATA
0/'	E	DATA
0/'	F	DATA

X = Beliebiger Wert zwischen 0 und F

4.5.3. Eingeben eines Wertes in eine Speicherzelle

Hierfür sind zwei Kommandos notwendig

- Der Befehlszähler wird mit dem in 4.5.1. gezeigten Kommando auf die gewünschte Speicheradresse gesetzt. (Der Wert erscheint bei ADDRESS)

  SET 4/P adresse EX

- Anschließend wird die Tastenfolge

  SET MEM wert EX

  gegeben.
  Nach SET MEM erscheint bei DATA der augenblickliche Inhalt der Speicherzelle. Der Ausdruck „wert" ist durch den gewünschten Wert zu ersetzen.

Beispiele: Die Speicherzelle mit der Adresse 3C83 soll den Wert FF erhalten

SET 4/P 3/I C 8/H 3/I F

SET MEM F F EX

Die Speicherzelle mit der Adresse 3CC7 soll den Wert 12 erhalten

SET 4/P 3/I C C 7/X F

SET MEM 1 2 EX

4.5.4. Ausgabe des Inhaltes einer Speicherzelle auf die Anzeige

Auch hier sind zwei Kommando notwendig. Das erste Kommando dient dabei wie unter 4.5.3. der Identifizierung der gewünschten Zelle, während das zweite Kommando die Ausgabe Ihres Inhalts verursacht.

SET 4/P adresse EX

DIS MEM

Beispiele: Der Inhalt der Speicherzelle 0083 soll ausgegeben werden

SET 4/P 0/' 0/' 8/H 3/I F'

DIS MEM

Bei DATA erscheint der Inhalt dieser Speicherzelle

Der Inhalt der Speicherzelle 3CFF soll ausgegeben werden

SET 4/P 3/I C F F EX

DIS MEM

Bei DATA erscheint der Inhalt dieser Speicherzelle.

4.5.5. Eingabe eines Anwenderprogrammes

Für die Eingabe von Programmen ist der Eingabemodus vorgesehen, der das bequeme Eingeben auch längerer Programmteile ermöglicht. Das Setzen des Befehlszählers ist dabei nur einmal nötig und geschieht danach automatisch.

a) Als erstes wird der Befehlszähler auf **die** Adresse gesetzt, auf der der erste Befehl des Anwenderprogrammes stehen soll (siehe 4.5.1.)

SET 4/P adresse EX

b) Daraufhin wird der Eingabemodus eingeleitet in dem die Taste

INPUT

gedrückt wird.

c) Nun können Sie ein Programm eingeben, und zwar in folgehender Weise

X . . X EX

X . . X EX

X . . X EX

X = gewünschte Daten in hexadezimalem Format

Der Befehlszähler, der jeweils die aktuelle Speicheradresse enthält, wird bei jedem EX um 1 erhöht. Gleichzeitig kommt der aktuelle Inhalt dieser Zelle zur Anzeige. Damit werden immer Adresse und zugehörige Daten **parallel** angezeigt. Der bei DATA stehende Wert **kann** dann durch Eintippen von Hex-Ziffern verändert werden. Ein sofortiges Drücken der Taste EX **ohne** vorherige Veränderung führt zur Wieder-Übernahme der alten (noch auf der Anzeige stehenden) Daten.

d) Haben Sie Ihr Programm vollständig eingegeben, müssen Sie den Eingabemodus wieder verlassen.
Dies geschieht durch Drücken der Taste STORN.
Als sichtbare Kontrolle für das Verlassen des Input-Modus, wird die gesamte Anzeige kurzzeitig ausgeschaltet.

Beispiel:
Das folgende kleine Programm soll eingegeben werden. Anfangsadresse soll 3C70 sein.

3E
0F
D3
0A
78
D3
08
C3
0B
00

Folgende Aktivitäten sind notwendig:

a) SET 4/P 3/I C 7/X 0/' EX

b) INPUT

c) 3/I 0E EX

0/' F EX

D 3/I EX

- - - - - - - - - - - - - -

0/' 0/' EX

d) STORN

4.5.6. Ausgabe von größeren zusammenhängenden Speicherbereichen

Wie bei der Eingabe ist auch bei der Ausgabe längerer Programme eine Bedienungs-Erleichterung vorgesehen. Auch hier ist lediglich die Eingabe der Anfangsadresse notwendig, das Weiterschalten geschieht dann automatisch.

a) Zuerst wird der Befehlszähler auf die Anfangsadresse des gewünschten Speicherbereichs gesetzt (4.5.1.) und der Wert dieser Zelle abgerufen (4.5.4.).

SET 4/P adresse EX

DIS MEM

Der Inhalt dieser Zelle wird bei DATA angezeigt, während die zugehörige Adresse bei ADDRESS steht.

b) Wollen Sie nun den Inhalt der nachfolgenden Zelle ausgeben, drücken Sie nur

IDM

Dies hat zur Folge, daß der Befehlszähler um eins erhöht wird und der Inhalt der damit bezeichneten Speicherzelle bei DATA erscheint.
Durch festgesetztes Drücken von IDM können somit z.B. auch längere Programme durchgesehen und überprüft werden, da nur eine Taste zu betätigen ist und die Speicher-Adresse und ihr Inhalt jeweils gleichzeitig angezeigt werden.

Beispiel:
Das unter 4.5.5. angegebene Programm befindet sich im Speicher mit Anfangsadresse 3C70 und soll schrittweise angezeigt werden.

a) SET 4/P 3/I C 7/X 0/' EX

DIS MEM

		ADDRESS	DATA
Danach ergibt sich:		3C70	3E
b) IDM	→	3C71	0F
IDM	→	3C72	D3
IDM	→	3C73	0A
IDM	→	3C74	78

usw.

4.5.7. Starten eines Anwenderprogramms
Dazu setzen Sie den Befehlszähler (4.5.1.) auf die Anfangsadresse Ihres Programmes und drücken daraufhin die Taste

START

Solange das Anwendrprogramm bearbeitet wird, erlöschen alle Anzeigeelemente, (außer das Anwenderprogramm wird zur Ansteuerung benützt).

Beispiel:
Ihr Anwenderprogramm steht im Speicher mit der Anfangsadresse 3C66. Sie wollen es starten

SET 4/P 3/I C 6/Y 6/Y EX

START

Die Anzeigen erlöschen und leuchten erst nach Beendigung des Anwenderprogramms wieder auf.

4.5.8. Programmabarbeitung im Einzelschrittverfahren
(..single step")
Hierfür setzen Sie Ihren Befehlszähler (4.5.1.) auf die gewünschte Adresse im Programm (oder lassen das Programm per Breakpoint zum Halten kommen) und betätigen anschließend die Taste

STEP

Es wird genau ein Befehl ausgeführt. Bei ADDRESS erscheint der nun aktuelle Stand des Befehlszählers, bei DATA der Inhalt dieser Speicherzelle = Anweisungsteil des nächsten auszuführenden Befehls (siehe auch Kap. 3.2.5.!!).

Beispiel:
Gegeben sei wieder das Beispiel aus 4.5.5. Der Befehlszähler sei bereits auf 3C70 gesetzt und ein DIS MEM durchgeführt.

	ADDRESS	DATA
	3C70	3E
STEP →	3C72	D3
STEP →	3C74	78
STEP →	3C75	D3

usw.

Bitte machen Sie sich an dieser Stelle klar, daß EIN BEFEHL im System Z80 EIN, ZWEI, DREI ODER VIER BYTES umfaßt. Beim Einzelschrittverfahren springt der Befehlszähler bei jedem Betätigen der Taste STEP um 1, 2, 3 oder 4, je nachdem, ob die Ausführung 1, 2, 3 oder 4 Bytes langen Befehls durch die Betätigung ausgelöst wurde.

4.5.9. Setzen, Löschen und Anzeigen eines Haltepunkts (Breakpoint) (nur im RAM möglich).

a) Das Setzen eines Haltepunkts auf eine bestimmte Programmadresse wird durch folgende Tastenfolge erreicht:

SET BR adresse EX

..adresse" ist dabei durch den gewünschten Wert zu ersetzen. nach SET BR erscheint bei ADDRESS die augenblickliche Haltepunktadresse. Der Wert 0000 bedeutet kein BREAK gesetzt.

NB: Es ist immer nur ein Haltepunkt möglich. Die Veränderung eines bestehenden Haltepunkts erfolgt über einen erneuten Setzvorgang, wobei der alte Breakpoint automatisch gelöscht wird.

Beispiel:
In dem bekannten Anwenderprogramm, das im Speicher von Adresse 3C70 an untergebracht ist, soll zu Prüfzwecken ein Haltepunkt auf Adresse 3C74 gesetzt werden:

SET BR 3/I C 7/X 4/P EX

Wird das Anwenderprogramm gestartet, und erreicht der Befehlszähler diese Adresse, so wird das Programm angehalten. Sichtbar wird dies durch das Wiederaufleuchten der Anzeigeelemente. Bei ADDRESS erscheint der Wert der Breakpointadresse, bei DATA der Code des nächsten Befehls. Durch das Auflaufen auf den Haltepunkt (per START oder STEP) wird dieser gelöscht. Dadurch kann sofort ein neuer Haltepunkt, auch mit dem alten Wert vereinbart werden. Eine weitere Abarbeitung des Programms kann an dieser Stelle mit START oder STEP ausgelöst werden.

b) Das Löschen eines Haltepunktes kann auf zweierlei Arten erfolgen
- Wie oben beschrieben durch Überlaufen mit START oder STEP, wenn das Programm den Haltepunkt erreicht hat.
- Durch Setzen der Haltepunktadresse auf den Wert 0000, wenn Ihr Mikrocomputer nicht mit der Abarbeitung eines Anwenderprogrammes beschäftigt ist.

c) Das Anzeigen der aktuellen Breakpointadresse erfolgt durch das Kommando

DIS BR

4.5.10. Abspeichern auf Cassette

Hierzu ist es notwendig die Anfangs- und Endadresse des abzuspeichernden Programmstücks einzugeben. Diese Adressen werden in insgesamt 4 Speicherzellen des RAM abgelegt, von wo aus sie von der entsprechenden Programmroutine abgeholt werden.
Die Adressen lauten wie folgt:

3C40	Startadresse	low Byte
3C41	Startadresse	high Byte
3C42	Endadresse	low Byte
3C43	Endadresse	high Byte

Beispiel:
Ein Programm, das im Speicher von Adresse 3C60 bis 3CA7 steht, soll auf Cassette übertragen werden. Die Eingabe der Start- bzw. Endadresse erfolgt durch die Kommandofolge:

```
SET   4/P   3/I   C   4/P   0/'   EX

INPUT

6/X   0/'   EX

3/I   C     EX

A     7/X   EX

3/I   C     EX

STORN
```

Um später ein problemfreies Wiedereinlesen zu gewährleisten, muß folgendes beachtet werden:
Der Beginn einer „Sendung" wird auf Grund der Einfachheit des Cassetten-Interface durch ein längerdauerndes high Signal gebildet, dem die fallende Flanke des Startbits des ersten Wortes folgt. Um später das Programm sicher wieder laden zu können, empfiehlt es sich, dieses high-Signal auf ca. ½ Minute auszudehnen (siehe „Load"). Die mit dem Zählwerk des Cassettenrecorders verbundene Ungenauigkeit kann damit ausgeschaltet werden.
Das Ende der „Sendung" wird mit Hilfe eines Breakcharakters markiert, der automatisch dem USER-Programm angeschlossen wird.

Beispiel:
Das obengenannte Programm soll nun auf das Cassettengerät übertragen werden

- Stellen Sie Ihr Cassettengerät so ein, daß ein definierter Bereich des Bandes zur Bespielung zur Verfügung steht u. notieren Sie den Wert des Bandzählwerkes.
- Schalten Sie auf Aufnahme und steuern Sie Ihr Gerät voll aus, bzw. verwenden Sie die automatische Aussteuerung.
- Lassen Sie jetzt das Band laufen (damit wird das längerdauernde high auf das Band geschrieben).
- Nach ca. 30 Sekunden notieren Sie wieder den Stand des Bandzählwerkes und drücken die Taste STORE.
  Auf der linken Anzeige wird, solange auf das Band geschrieben wird, ein S für STORE angezeigt. Nach Beendigung des Abspeichervorgangs erscheint wieder die normale Anzeige.
- Schalten Sie jetzt Ihr Cassettengerät ab.

Damit ist Ihr Programm auf Band geschrieben.

4.5.11. Laden von Cassette in den Speicher

Hierzu ist es notwendig, die Anfangsadresse des Speicherbereiches, in den die Information geladen werden soll, anzugeben.
Die Speicherzellen in denen diese Adresse steht sind:

3C44	Startadresse	low Byte
3C45	Startadresse	high Byte

Beispiel:
Ein auf Band stehendes Programm soll in den Speicher geladen werden. Die Anfangsadresse sei 3C80
Die Eingabe erfolgt durch:

```
SET   4/P   3/I   C   4/P   4/P   EX

INPUT

8/H   0/'   EX
3/I   C     EX

STORN
```

- Stellen Sie Ihr Cassettengerät auf den ersten der beiden notierten Werte des Bandzählwerkes und schalten Sie auf Wiedergabe.
- Lassen Sie das Band laufen
- Sobald das Bandzählwerk in etwa den Mittelwert der beiden beim Abspeichern notierten Werte zeigt, drücken Sie die Taste LOAD
  Das Programm wartet jetzt auf die erste fallende Flanke am Eingang der seriellen Schnittstelle, was gleichbedeutend mit dem Beginn des ersten Wortes ist. Anschließend beginnt der eigentliche Ladevorgang. Während des Ladevorgangs wird auf der linken Anzeige ein L für LOAD angezeigt.
  Nach Beendigung des Ladevorgangs (Erkennen des Breakcharakters) leuchtet wieder die normale Anzeige auf.
- Schalten Sie Ihr Cassettengerät ab.

Damit ist die Information in den Speicher geladen.

NB Sollte während des Ladens das Lämpchen ERROR aufleuchten, so wurde ein Fehler bei der Übertragung erkannt und der gesamte Vorgang ist zu wiederholen.
Die ERROR Lampe erlischt, sobald erneut LOAD gedrückt wird.

5. Zusammenbau

Achtung!
Bei den Mikrocomputer Bausteinen handelt es sich überwiegend um MOS-Bausteine, die trotz ihrer integrierten Schutzstrukturen mit Vorsicht gehandhabt werden sollten!
Bitte nehmen Sie die Bausteine deshalb erst dann aus der leitenden Verpackung heraus, wenn es durch einen Hinweis in der Bauanleitung notwendig ist!

5.1. Druckschriften

Außer der vorliegenden Druckschrift sind im Z80-KIT enthalten:

- ☐ Z80-Assemblersprache-Handbuch
- ☐ Z80-CPU-Manual
- ☐ Z80-PIO-Manual
- ☐ Z80-Programmierkärtchen

Darüberhinaus werden Sie im Anhang der vorliegenden Druckschrift im Detail über die folgenden Eigenschaften des Z80-Systems instruiert:

- ☐ Z80 CPU-Zeitverhalten
- ☐ Befehlssatz der Z80-CPU
- ☐ Interrupt-Verhalten des Systems Z80

5.2. Notwendige Werkzeuge

Für den Zusammenbau sollten nach Möglichkeit folgende Arbeitsmittel zur Verfügung stehen:

- ☐ ebene Arbeitsunterlage
- ☐ Einspannbock oder andere Möglichkeit zum Fixieren der Platinen
- ☐ Lötkolben 25 W
- ☐ Radio-Lötzinn mit Flußmittel-Seele
- ☐ kleiner Seitenschneider
- ☐ Flachzange
- ☐ Schraubenzieher

5.3. Bauteile

Sämtliche Bauteile, mit Ausnahme der Leiterplatten und Teilen der Mechanik, befinden sich übersichtlich angeordnet im oberen Teil der Verpackung. Die Beschriftung entspricht dabei der Benennung der Bauteile im Anwenderhandbuch.

5.4. Stücklisten

Führen Sie zunächst eine Bestandsaufnahme durch. Überprüfen Sie dazu alle Bauteile an Hand der nachfolgenden Aufstellung. Nehmen Sie die Mikrocomputer-Bausteine dazu noch nicht aus dem mit leitendem Anstrich versehenen Fach!

Stückliste zur Grundplatine des Z80-KIT (siehe 5.3.)

Stck.	Bezeichnung	Bauteile Nr.	Bemerkung
Halbleiter			
2	DM 81 LS 97 N od. DM 81 LS 95 N	IC 12, IC 13	
1	SN 75 LS 494	IC 5	
1	SN 74 LS 221 N	IC 9	
1	SN 74 LS 138 N	IC 1	
2	SN 74 LS 74 N	IC 6, IC 8	
1	SN 74 LS 30 N	IC 3	
2	SN 7407 N	IC 10, IC 11	
2	SN 74 LS 04 N	IC 2, IC 7	
1	SN 74 LS 00 N	IC 4	
6	FND 357	A1 . . . A6	LED Anzeigen
1	TIL 209 B	L	Leuchtdiode
Stck.	Bezeichnung	Bauteile Nr.	
Widerstände und Kondensatoren			
8	Widerstand 10 kΩ 1/3 W	R11 . . . R18	braun schwarz orange
8	Widerstand 180Ω 1/3 W	R3 . . . R10	braun grau braun
6	Widerstand 1 kΩ 1/3 W	R19 . . . R24	braun schwarz rot
1	Widerstand 68kΩ 1/3 W	R1	blau grau orange
1	Kondensator 0,01 µ	C1	
2	Kondensator 22µ 16V	C2, C3	
Elektro-Mechanische Bauteile			
1	Buchsenleiste VG	STA	64polig, A und C
29	Tasten	T1 . . . T29	29 Schaltkörper 2 Tasten orange, extra breit 1 Taste orange 10 Tasten weiß 16 Tasten grau, beschriftet
1	Anschlußklemme K		4 polig
2	Anschlußkabel		rot/schwarz

Stückliste zum Einplatinencomputer des Z80 KIT (siehe 5.3.)

Stck.	Bezeichnung	Bauteil Nr.	Bemerkung
Mikrocomputer-Bausteine			
1	Z80 CPU	C5	Z80 Mikroprozessor
1	Z80 PIO	A1	Z80 progr. Parallel-Schnittstelle
1	CIC-I	A2	Festwertspeicher (weiß)
1	CIC-III	B4	Adreßdecoder (hellbl.)
2	2112 A	B2, B3	Schreib/Lesespeicher
IC's, Dioden und Quarz			
2	SN74LS367N	B5, B6,	
1	SN74LS74N	D4	
1	SN74LS04N	D3	
1	SN 74 LS 132 N	C2	
3	1 N 4148	D_1, D_3, D_4	Dioden
1	1 N 4001	D_2	Diode
1	Quarz 9,8304 MHz		

Widerstände und Kondensatoren			
3	Widerstand 330 Ω 1/3 W	R7 . . . R9	orange orange braun
3	Widerstand 470 Ω 1/3 W	R11 . . . R13,	gelb violett braun
7	Widerstand 4,7 kΩ 1/3 W	R1 . . . R6, R24	gelb violett rot
1	Tantalelko 22 µ 16 V	C1	
9	Tantalelko 4,7 µ 16 V	C2 . . . C6 und C9, C8, C11, C12	
1	Scheibenko. 1 n	C7	
Elektro-Mechanische Bauteile			
1	Messerleiste VG	STA	64polig A-C
1	Stiftleiste	STB	26polig
2	IC Sockel 40polig		zu C5 und A1
2	IC Sockel 24polig		zu A2 und A3
3	IC Sockel 16polig		zu B2, B3 und B4

Serienschnittstelle mit Audio Cassetten- u. TTY-Interface			
Stck.	Bezeichnung	Bauteil Nr.	Bemerkung
1	8251 oder 9551	D1	USART
1	SN 74 LS 32	C3	
1	SN 74 LS 193	E5	
2	SN 74 LS 293	E4/D5	
1	IL 74 oder TIL 111	E2	Optokoppler
1	2 N 2222	T1	Transistor
1	2 N 2907	T2	Transistor
2	BC 547 B	T3/T4	Transistoren
2	1 N 4148	D_5, D_6	Dioden
1	Widerstand 10Ω 1/3W	R29	br. schw. schw.
1	Widerstand 150Ω 1/3W	R18	br. grü. br.
1	Widerstand 470Ω 1/3W	R30	ge. viol. br.
3	Widerstand 1K 1/3W	R19, 20, 26	br. schw. rot
3	Widerstand 2,2K 1/3W	R32, 33, 35	rot rot rot
1	Widerstand 4,7K 1/3W	R17	ge. viol. rot
1	Widerstand 6,8K 1/3W	R28	bl. grau rot
2	Widerstand 10K 1/3W	R16, 34	br. schw. rot
1	Widerstand 47K 1/3W	R31	ge. viol. or.
1	Widerstand 220K 1/3W	R27	rot rot ge.
1	Widerstand 1 M 1/3W	R15	br. schw. grü.
3	Elko 0,22 µ16 V	C16, C17, C18	
1	Elko 0,47 µ/16 V	C15	
1	Elko 1 µ/16V	C19	
1	IC Fassung 28-polig		zu D1
1	Stiftleiste 20-polig	STC	

Mechanische Bauteile, Chassis		
Stck.	Bezeichnung	Bemerkung
1	Schutzfolie	
1	Grundplatte	
1	Frontplatte	
2	Haltebügel	
1	Seitenführungsleiste	
4	Schrauben	
2	Schrauben	M3 × 15
2	Schrauben	M2,5 × 8
4	Verlängerungsschrauben	M3 5mm Abst.
6	Muttern	M3
2	Muttern	M2,5
10	Beilagscheiben	
4	Gummifüße	
1	Aufkleber	

5.5. Vorgehensweise beim Aufbau des Z80 KIT

Um Fehler weitgehendst auszuschalten, wurde der Aufbau in mehrere Schritte unterteilt. Für jeden Abschnitt sind Bestückungspläne und eine entsprechende Anleitung vorhanden. Wir bitten Sie, bei jedem Schritt die Bauteile vor dem Einlöten erst in die vorgesehenen Plätze zu stecken und mit dem entsprechenden Bild zu vergleichen.

Bitte beachten Sie!
Obwohl die MOS-Bausteine Eingangsschutzstrukturen eingebaut haben, sollte man sie unbedingt vor statischer Elektrizität schützen. Die folgenden Vorsichtsmaßregeln sind daher unbedingt zu beachten.

Grundsätzliches zur Behandlung von Integrierten Schaltkreisen

- Ausrichten der IC-Anschlüsse
  Falls Anschlüsse eines IC's verbogen sind, sollten Sie unter keinen Umständen versuchen, das Bauteil mit Gewalt in seine Fassung einzusetzen, sondern erst alle Pins ausrichten. Dies erreicht man einfach, indem man das IC-Gehäuse zwischen die Fingerspitzen nimmt und die Beinchen durch leichten Druck auf das gesamte Gehäuse gegen den Arbeitstisch so richtet, daß sie alle senkrecht zum Gehäuse stehen.
- Einsetzen der IC's
  Jedes IC-Gehäuse trägt eine Markierung (Aussparung, Punkt, Pluszeichen oder die Ziffer „1"), die eindeutig die Lage der Anschlüsse definiert. Es ist besonders darauf zu achten, daß die IC's richtig herum eingesetzt werden, weil bei falscher Orientierung eine Zerstörung des IC's bei Inbetriebnahme zu befürchten ist.
  Stecken Sie die Pins auf einer Seite des IC's in ihre hierfür bestimmten Bohrungen bzw. Fassungen. **Drücken Sie die Pins nicht ganz hinein!** Sollten Sie Schwierigkeiten haben, die Pins in die Löcher zu bringen, helfen Sie mit der Klinge eines kleinen Schraubenziehers nach.
  Führen Sie die Pins der anderen IC-Seite genauso in die zugehörigen Löcher ein. Sobald alle Pins an ihrer Stelle sind, wird der IC auf seinen Platz gedrückt, indem er ganz leicht hin- und herbewegt wird, bis er am Anschlag auf der Platine oder in der Fassung sitzt.
- Herausnehmen von IC's aus ihren Fassungen
  Muß ein IC aus seiner Fassung wieder entfernt werden, bewegt man ihn leicht hin und her, bis er sich herauslösen läßt. Sobald eine Lücke zwischen IC und Fassung entstanden ist, kann man mit einem Schraubenzieher nachhelfen. Versuchen Sie, während des ganzen Vorgangs den IC, parallel zur Fassung zu halten.

Zusätzliche Fassungen
Nicht alle Integrierten Schaltkreise des Z80 KIT werden in Fassungen eingesetzt. Sollten Sie im Aufbau von Schaltungen noch geringere Erfahrungen besitzen, empfiehlt es sich, auch diese Bausteine mit entsprechenden Sockeln zu versehen. Das Einkreisen eines beim Aufbau evtl. gemachten Fehlers wird damit wesentlich erleichtert.

Grundsätzliches zur Behandlung von MOS Bausteinen

- Alle Gegenstände, die mit der Schaltung in Berührung kommen, sollten auf gleichem Potential liegen; denken Sie dabei an Lötkolben, Zinn, Werkzeuge usw.
  Daher ist häufiger körperlicher (möglichst elektrischer) Kontakt zwischen Arbeitstisch, Schaltung und den genannten Gegenständen wichtig (z. B. über leitende Schaumstoff-Unterlage).
- Gewöhnen Sie sich an, die IC's nur in leitender Umgebung zu berühren und zu lagern.
- Ebenso sollten Sie auch die Leiterplatte berühren, bevor Sie einen IC einsetzen.
- Fassen Sie den Baustein möglichst nur am Gehäuse und nicht an seinen Anschlüssen an.
- Berühren Sie grundsätzlich einen MOS-Baustein nur mit Gegenständen (z. B. Werkzeugen), die Sie vorher mit dem Arbeitstisch in Berührung gebracht haben.

Löthinweise
Erfahrungsgemäß sind in mehr als 90% aller Fälle, in denen ein KIT „nicht funktioniert", Löt-Kurzschlüsse die Ursache!

- Verwenden Sie nur einen Lötkolben mit sehr dünner Spitze!
- Suchen Sie die günstigste Art, die Lötspitze am Lötpunkt anzusetzen!
- Vermeiden Sie dabei, benachbarte Punkte zu berühren!
- Seien Sie sparsam mit Lötzinn!!!!

5.6. Zusammenbau der Grundplatine

Legen Sie sich alle Teile zurecht, die für den Aufbau der Grundplatine notwendig sind. Eine Aufstellung finden Sie in 5.3 unter der Überschrift: Stückliste zur Grundplatine des Z80 KIT.

Schritt 1 Bild 1

☐ Nehmen Sie nun die Grundplatine (ca. 275 mm × 220 mm7 und legen Sie sie so vor sich wie es in Bild 1 gezeigt wird. Die Zahlen am oberen Rand der Platine, (sie beziffern die Busanschlüsse) müssen dabei von links nach rechts in aufsteigender Reihenfolge stehen. Dies ist die Bestückungsseite für die Tasten, die hexadezimalen Anzeigen und die Leuchtdiode.

Schritt 2 Bild 2

☐ Löten Sie nun alle 29 Schalterkörper in die Platine ein. Die Codierbohrungen lassen eine Montage der Tasten nur in der richtigen Lage zu. Es empfiehlt sich, die Tasten vorerst nur an zwei, diagonal gegenüberliegenden Stiften anzulöten. Dadurch können geringfügige mechanische Korrekturen leichter durchgeführt werden. Erst wenn alle Schaltkörper gleichmäßig „satt" auf der Platine aufliegen, werden die restlichen Stifte verlötet.

Tasten bitte sehr vorsichtig einlöten, sie sind hitzeempfindlich und funktionieren bei zu heißem Einlöten nicht.

☐ Löten Sie die sechs Anzeigeelemente in die vorgesehenen Plätze. Beachten Sie dabei die Lage des Dezimalpunktes!

☐ Löten Sie die Leuchtdiode ein.

Eine Unterscheidung von + und − kann wie folgt getroffen werden:
Betrachtet man die Leuchtdiode gegen das Licht, so erkennt man im Inneren des roten Glaskörpers zwei unterschiedlich geformte Teile, einen kurzen Stift und einen Haken. Der zum kurzen Stift führende Anschluß ist mit + zu verbinden und deshalb im linken Loch einzusetzen.
Die Grundplatine ist nun auf der Vorderseite fertig bestückt. Ihr Ergebnis muß mit Bild 2 übereinstimmen.

Schritt 3 Bild 3

Es folgt nun die Bestückung der restlichen Bauteile. Dazu drehen Sie die Platine um und legen sie wie in Bild 3 gezeigt vor sich.

Schritt 4 Bild 4

☐ Löten Sie nun die Widerstände R1 und R3 bis R24 ein

☐ Löten Sie die Kondensatoren C1 bis C3 ein.
Beachten Sie dabei die Polarität (+) der Tantalelkos C2 und C3! (+ Zeichen ist auf der Platine vorgedruckt).

☐ Löten Sie die Buchsenleiste VG in einen der möglichen Steckplätze ein. Befestigen Sie dazu zuerst die Buchsenleiste mittels den dafür vorgesehenen Schrauben u. Muttern (M2,5!) auf der Platine, um einen einwandfreien „Sitz" zu erhalten.
Da die auf der Lötseite überstehenden Schraubenenden gleichzeitig als Abstützung der Platine gegen die Frontplatte dienen, empfiehlt es sich, den Steckplatz Nr. 2 oder Nr. 3 zu wählen.
Beachten Sie vor dem Löten, daß sowohl die Bezifferung (1 . . . 32) als auch die Beschriftung (A/C) von Platine und Buchsenleiste übereinstimmen!

☐ Löten Sie die vierpolige Anschlußklemme K der Stromversorgung ein
Sind Sie mit diesen Arbeiten fertig muß sich Ihr Ergebnis mit Bild 4 decken.

Schritt 5 Bild 5

☐ Nehmen Sie nun die integrierten Schaltkreise (IC's) zur Hand, und zwar in der, in der Bestückungsliste Kap. 5.3 aufgeführten Reihenfolge. Die dort aufgeführten Bauteilenummern finden Sie auf der Grundplatine wieder. Dadurch ist ein Auffinden der entsprechenden Plätze gesichert. Vergleichen Sie, um sicher zu gehen, nochmals die Lage der Bausteine mit Bild 5! Beachten Sie besonders die Lage von PIN 1!

☐ Wenn Sie sicher sind, den Baustein richtig eingesetzt zu haben, löten Sie ihn ein.
Sind diese Arbeiten beendet, muß sich Ihr Ergebnis mit Bild 5 decken. Die Grundplatine ist damit komplett bestückt.

Schritt 6 Bild 6

☐ Stecken Sie nun die Tasten entsprechend Bild 6 auf die Schaltkörper. Zuvor müssen mit einem kleinen Schraubenzieher die Stifte justiert werden.
Damit sind sämtliche Arbeiten an der Grundplatine abgeschlossen. Es folgt nun die Bestückung des Einplatinencomputers. Legen Sie sich dazu wiederum alle Bauteile zurecht.

5.7. Zusammenbau des Einplatinencomputers

Schritt 7 Bild 7

☐ Legen Sie die Einplatinencomputerplatine in der in Bild 7 gezeigten Weise vor sich. Dies ist die Bestückungsseite für sämtliche Bauteile!

Schritt 8 Bild 8

☐ Löten Sie die Widerstände R1 bis R9, R11, R12, R13 und R24 ein.

☐ Löten Sie die Kondensatoren C1 bis C9 sowie C11 und C12 ein. Beachten Sie dabei unbedingt die Polarität (+) der Tantalelkos!

☐ Löten Sie die Dioden D_1, D_2, D_3, und D_4 und den Quarz ein. Achten Sie dabei auf die richtige Polarität der Dioden. Die Anschlüsse des Quarzes müssen vor dem Einbau rechtwinklig abgebogen werden. Das Quarzgehäuse sollte außerdem noch so an der Masseschiene angelötet werden, daß kein Kontakt mit anderen Leiterbahnen entsteht.

☐ Stecken Sie jetzt die 64polige Messerleiste VG in die vorgesehene Position STA. Spannen Sie die zusammengesteckten Teile fest ein, so daß die Messerleiste fest auf der Platine anliegt und löten Sie die Anschlüsse fest.

☐ Stecken Sie die 26polige Stiftleiste in die vorgesehene Position STB und löten Sie sie fest.

☐ Setzen Sie die Jumper J1, J2, J3 und J4 nach Bild 15 Kap. 5.9.
J1 bis J4 dienen der Auswahl des verwendeten Festwertspeichertyps. Die Bilder 13 bis 15 zeigen die Lage der Jumper für verwendbare Typen. Der mitgelieferte Festwertspeicher CIC I (Monitorprogramm) erfordert eine Anpassung nach Bild 15.

☐ Setzen Sie die Jumper J5, J6, J7, J8, J9 und J10 nach Bild 16 Kap. 5.9.
Die Jumper J5 bis J10 dienen zur Einstellung der Übertragungsrate der seriellen Schnittstelle, und ergeben in der gezeigten Anordnung ein Rate von 110 Baud. Für andere Übertragungsraten siehe 2.2.2.6.

Nach Beendigung dieses Arbeitsganges müssen sich Platinenansicht und Bild 8 decken. Kontrollieren Sie bitte nochmals alle Tantalelkos auf richtige Polarität!

Schritt 9 Bild 9

☐ Wir kommen nun zu den in der Stückliste im Abschnitt 5.3. aufgeführten IC-Fassungen. Hinter den Fassungen stehen die Bauteilenummern der Bausteine für die sie gedacht sind. Nehmen Sie die Fassungen und stecken Sie sie in die in Bild 9 gezeigten Positionen. Achten Sie dabei darauf, daß die Markierung für Pin 1 an der richtigen Stelle liegt und löten Sie die Fassungen ein.

Ihr Ergebnis muß nun Bild 9 entsprechen!

Schritt 10 Bild 10

Hier gilt wieder das bei 5.5. gesagte bezüglich Behandlung der IC's.

☐ Nehmen Sie der Reihe nach die IC's zur Hand die in der Bestückungsliste von Kap. 5.3. aufgeführt sind. Setzen Sie die Bausteine in die in Bild 10 gezeigten Positionen. Überprüfen Sie die Lage an Hand von Bild 10! Überprüfen Sie die Lage von Pin 1! Ist der Baustein richtig eingesetzt, kann er eingelötet werden.
Nachdem Sie alle Bausteine eingesetzt haben, muß eine Übereinstimmung mit Bild 10 gegeben sein.

Schritt 11 Bild 11

Zum Schluß wird die Serienschnittstelle mit Audio-Cassetteninterface und TTY Schnittstelle aufgebaut. Gehen Sie hier auch so vor, daß Sie zuerst die Widerstände und Kondensatoren einlöten und dann zum Einbau der übrigen Elemente mit Ausnahme des 8251 übergehen. Bild 11 gibt Auskunft über die Lage der einzelnen Teile.
Letzter Schritt ist es jetzt, die Mikrocomputerbausteine Z80 CPU, Z80 PIO, 8251, 2112-A, und die Steuerbausteine CIC I u. III in die Fassungen zu stecken.

Schritt 12 Bild 12

Achtung: Die Bausteine CIC-I u. III sind gemäß Bild 12 farbig gekennzeichnet. Tragen Sie Sorge dafür, daß die Bausteine an der richtigen Stelle eingesetzt werden, da sonst Ihr Z80 KIT nicht funktionieren kann.
Wenn Sie die Mikrocomputerbausteine in ihre Fassungen eingesetzt haben, muß sich Ihre Platine in Übereinstimmung mit Bild 12 befinden.
Ihr Z80-KIT ist damit komplett bestückt.

5.8. Mechanischer Aufbau

In der Verpackung sind für den mechanischen Aufbau folgende Teile enthalten:

☐ Schutzfolie
☐ Frontplatte mit Aussparungen
☐ Montageplatte
☐ Seitenführungsleiste mit fünf Führungsschlitzen
☐ 2 Haltebügel
☐ 4 Gummifüße selbstklebend
☐ Schrauben, Muttern, Abstandschrauben und Beilagscheiben

Vor dem Zusammenbau muß der Aufkleber aufgebracht werden.

Der Aufbau ist an Hand der Darstellung in Bild 17 leicht nachzuvollziehen.
Beachten Sie dabei besonders die Lage der Seitenführungsleiste!
Falls die Anzeigeelemente oder die Tasten nicht durch die Ausschnitte passen, oder die Anzeigen klemmen, ist der Ausschnitt mit einer Schlüsselfeile zu vergrößern.
Stecken Sie jetzt den Einplatinencomputer in die auf der Grundplatine vorbereitete Buchsenleiste. Die hintere Ecke der Platine muß dabei in einem der fünf Führungsschlitze zu liegen kommen. Bitte beachten Sie, daß die Schlitze hauptsächlich als Seitenführung und nur sekundär als Auflage gedacht sind, d. h. es ist nicht notwendig, daß die eingesteckten Platinen an diesem Punkt aufliegen.

Ihr Mikrocomputer ist jetzt fertig aufgebaut. Bevor Sie ihn einschalten, ist es jedoch unbedingt notwendig, sich mit der Arbeitsweise des Gerätes und seiner Bedienung vertraut zu machen (siehe hierzu 2.3. und 4.).

5.9. Bestückungspläne zu 5.6. und 5.7.

T1
T28
T29
A1
A4
A5
A6
0V
+5
+12
KONTRON ELEKTRONIK
ECB-M
REV D
114-2B

Bild 2

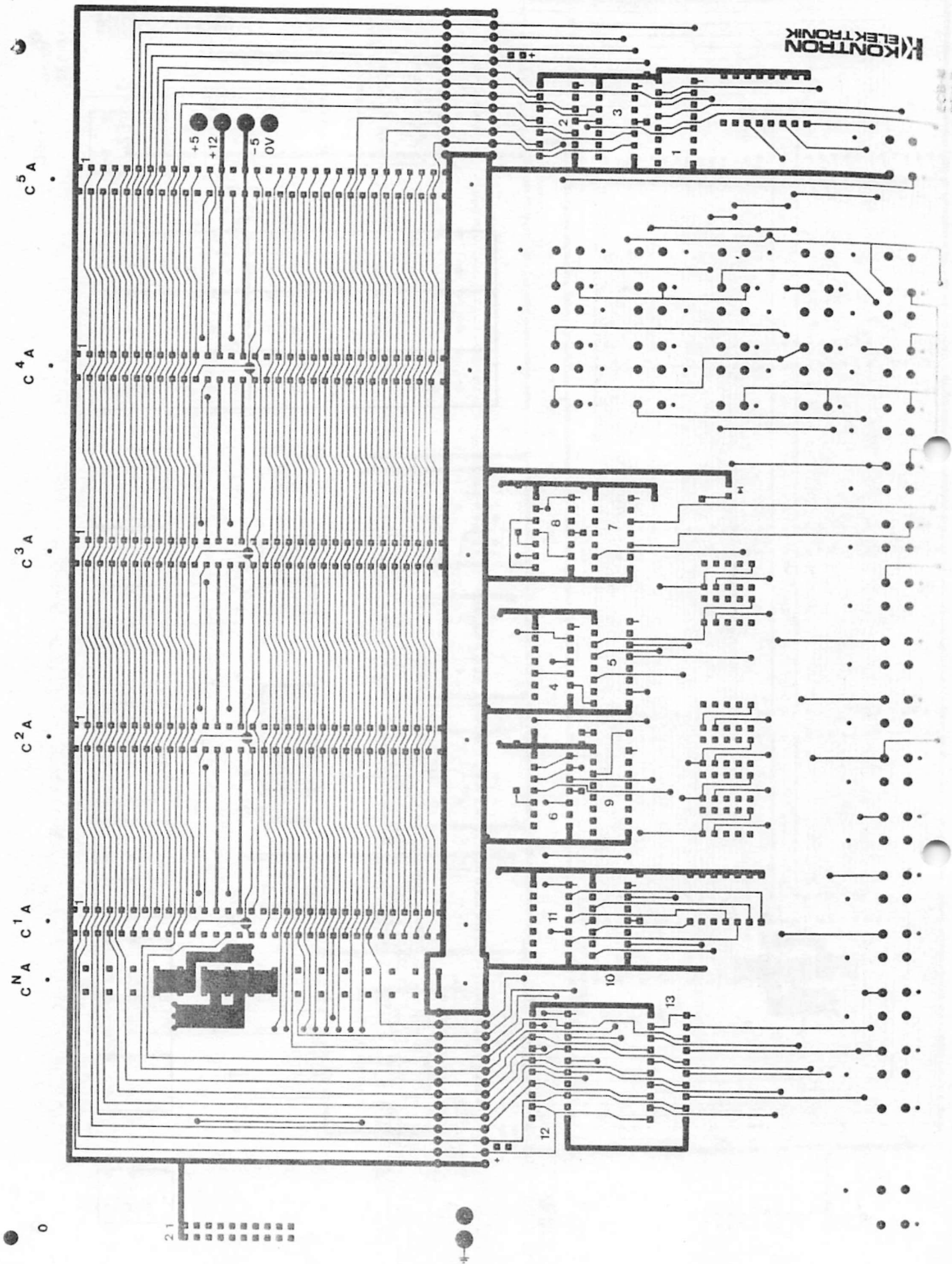

KONTRON
ELEKTRONIK
+5
+12
-5
0V
C^5A
C^4A
C^3A
C^2A
C^1A
C^NA

Bild 3

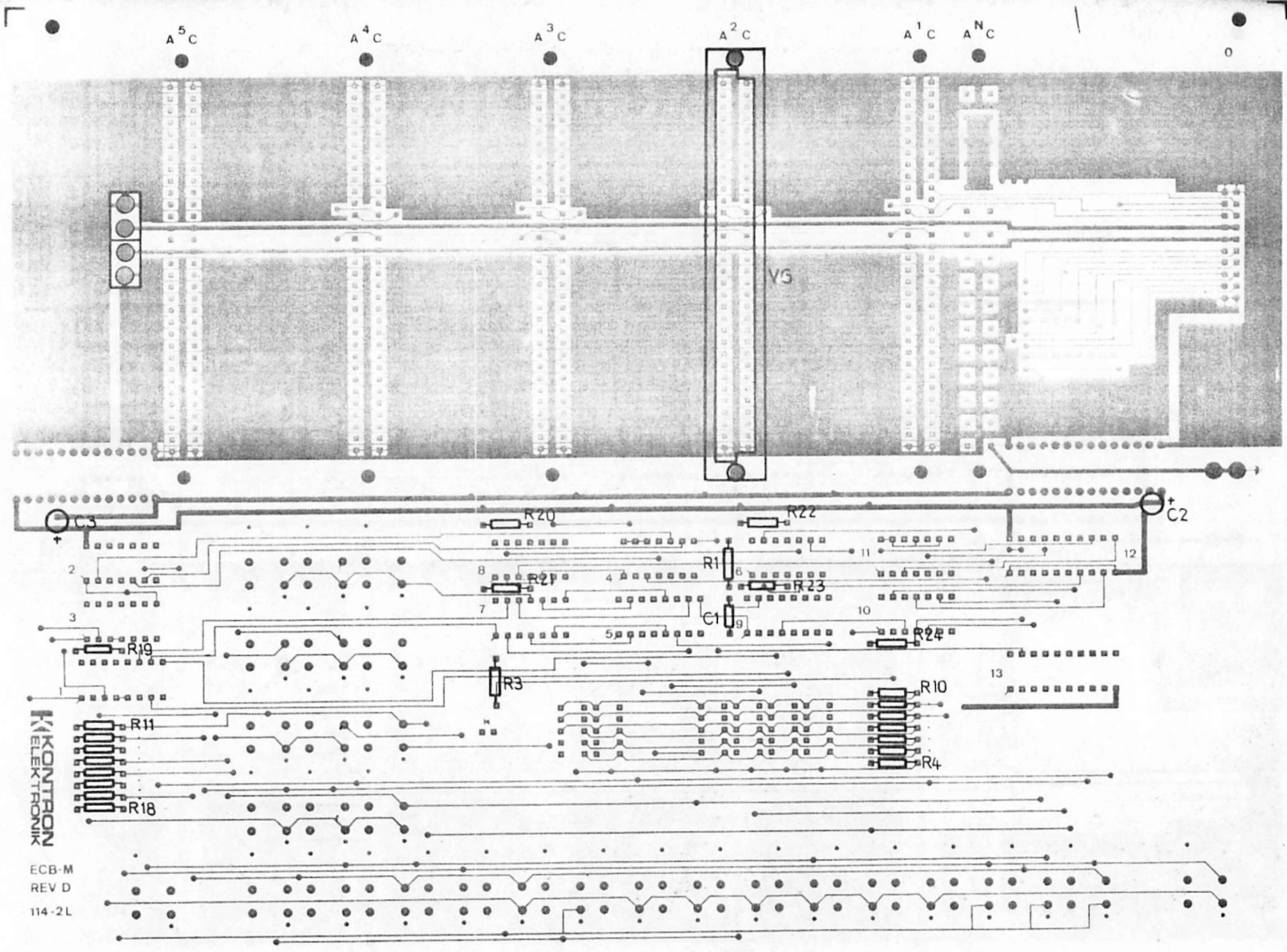

A5C
A4C
A3C
A2C
A1C
ANC
0
VS
C3
C2
R20
R22
R1
R21
R23
C1
R24
R19
R3
R10
R11
R4
R18
KONTRON ELEKTRONIK
ECB-M
REV D
114-2L

Bild 4

271,25 x 217 mm
7 Leiterbahnen
090
130
110
350 restliche Bohrungen
KONTRON ELEKTRONIK
ECB-M
REV D
114-2L
A5C
A4C
A3C
A2C
A1C
ANC
VG
IC1
IC2
IC3
IC4
IC5
IC6
IC7
IC8
IC9
IC10
IC11
IC12
IC13
R1
R3
R4
R10
R11
R18
R19
R20
R21
R22
R23
R24
C1
C2

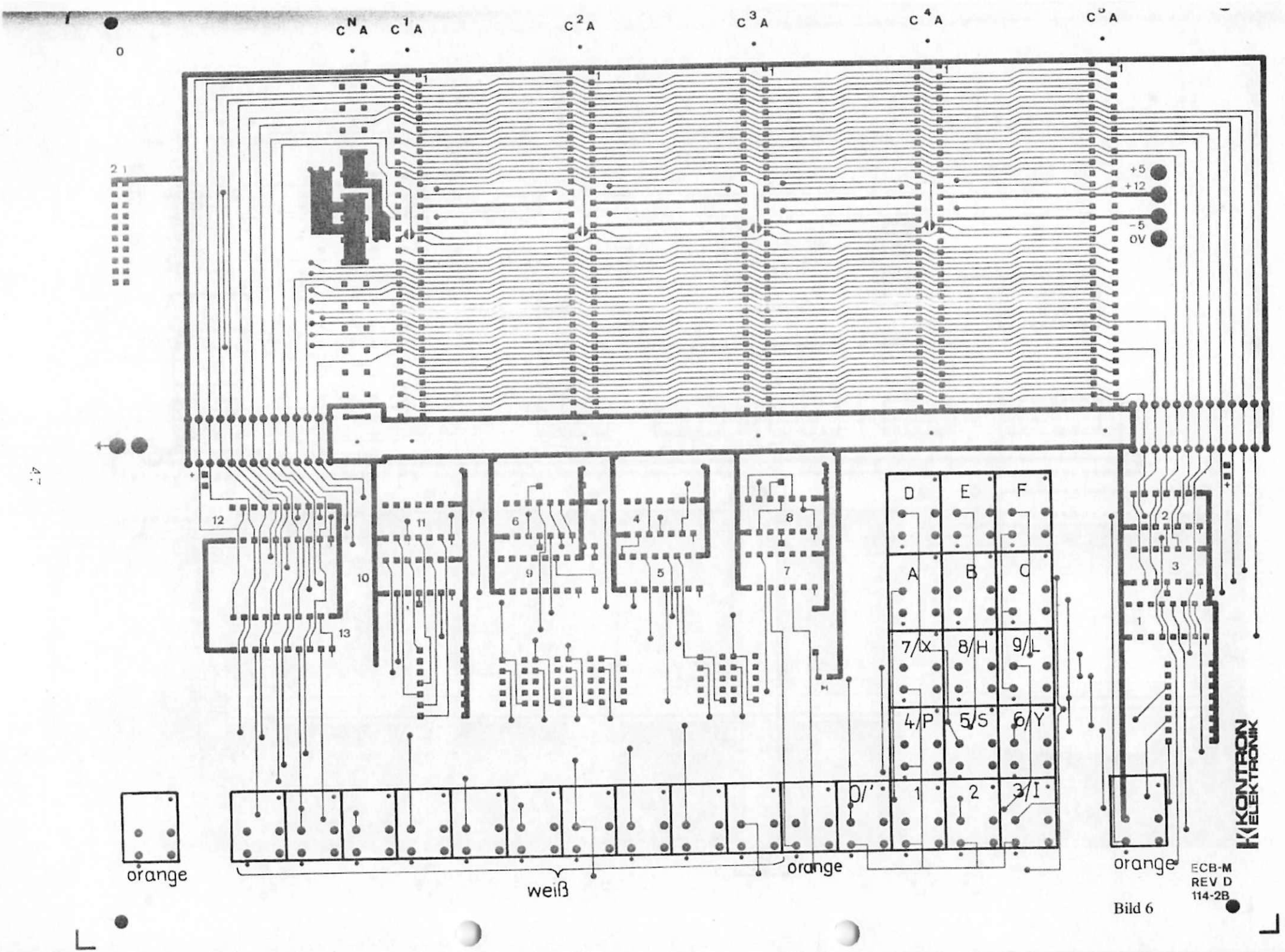

C^N A
C^1 A
C^2 A
C^3 A
C^4 A
C^5 A
+5
+12
-5
0V
D
E
F
A
B
C
7/X
8/H
9/L
4/P
5/S
6/Y
0/'
1
2
3/I
orange
weiß
orange
orange
KONTRON ELEKTRONIK
ECB-M
REV D
114-2B

Bild 6

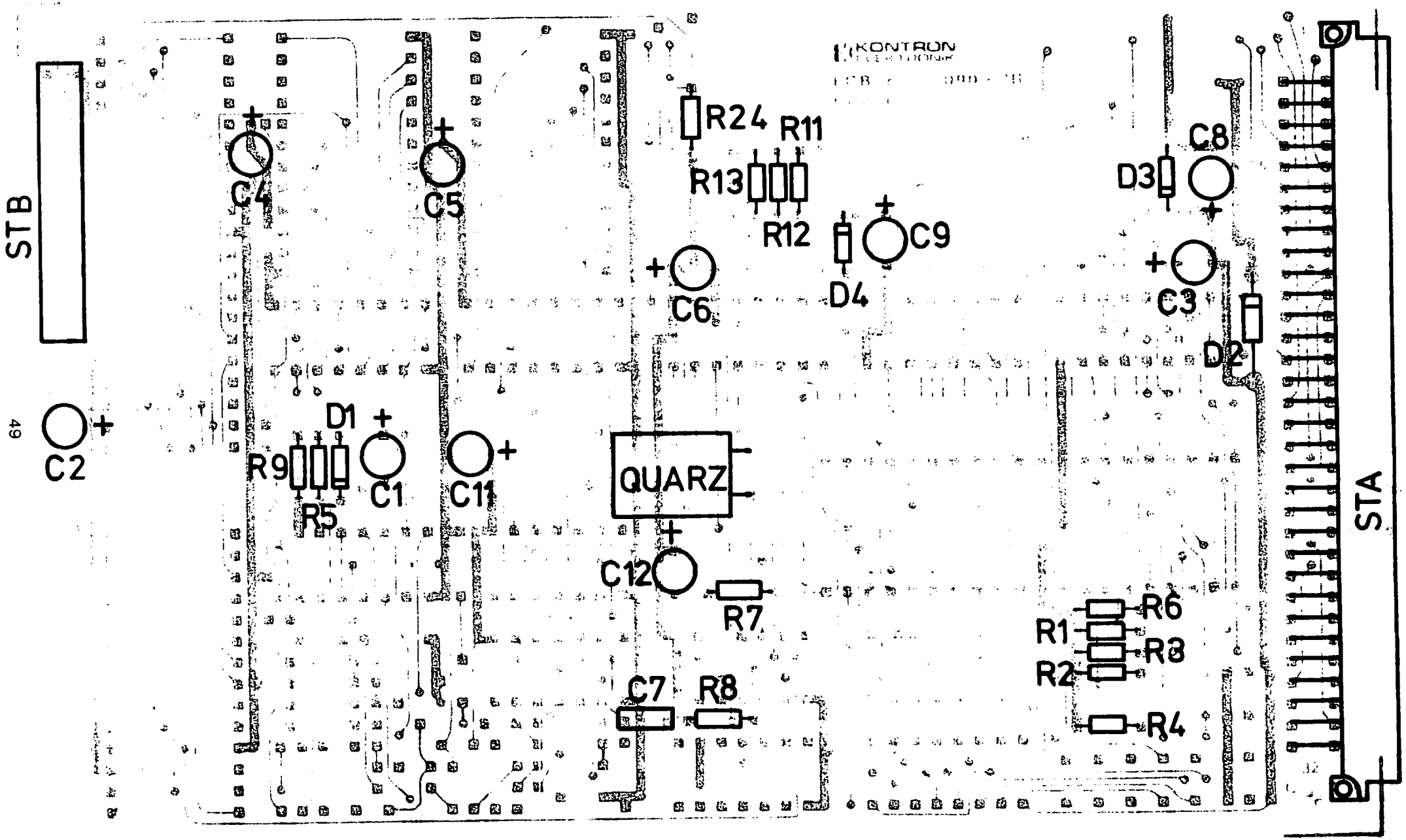

STB
STA
C2
C4
C5
C6
C1
C11
C12
C7
C8
C3
C9
R24
R11
R13
R12
R9
R5
R7
R8
R6
R1
R3
R2
R4
D1
D2
D3
D4
QUARZ
KONTRON

Bild 8

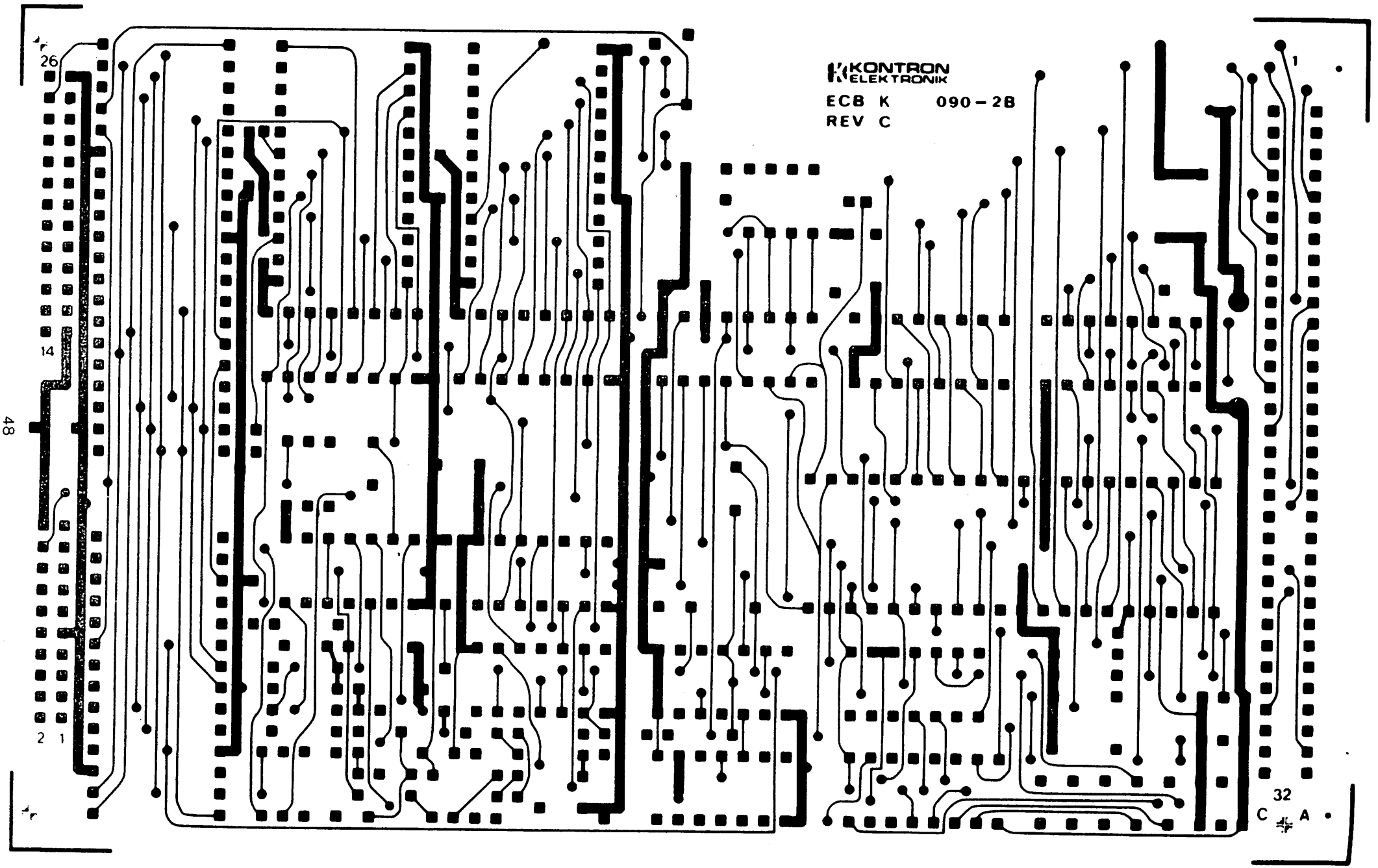


Bild 7

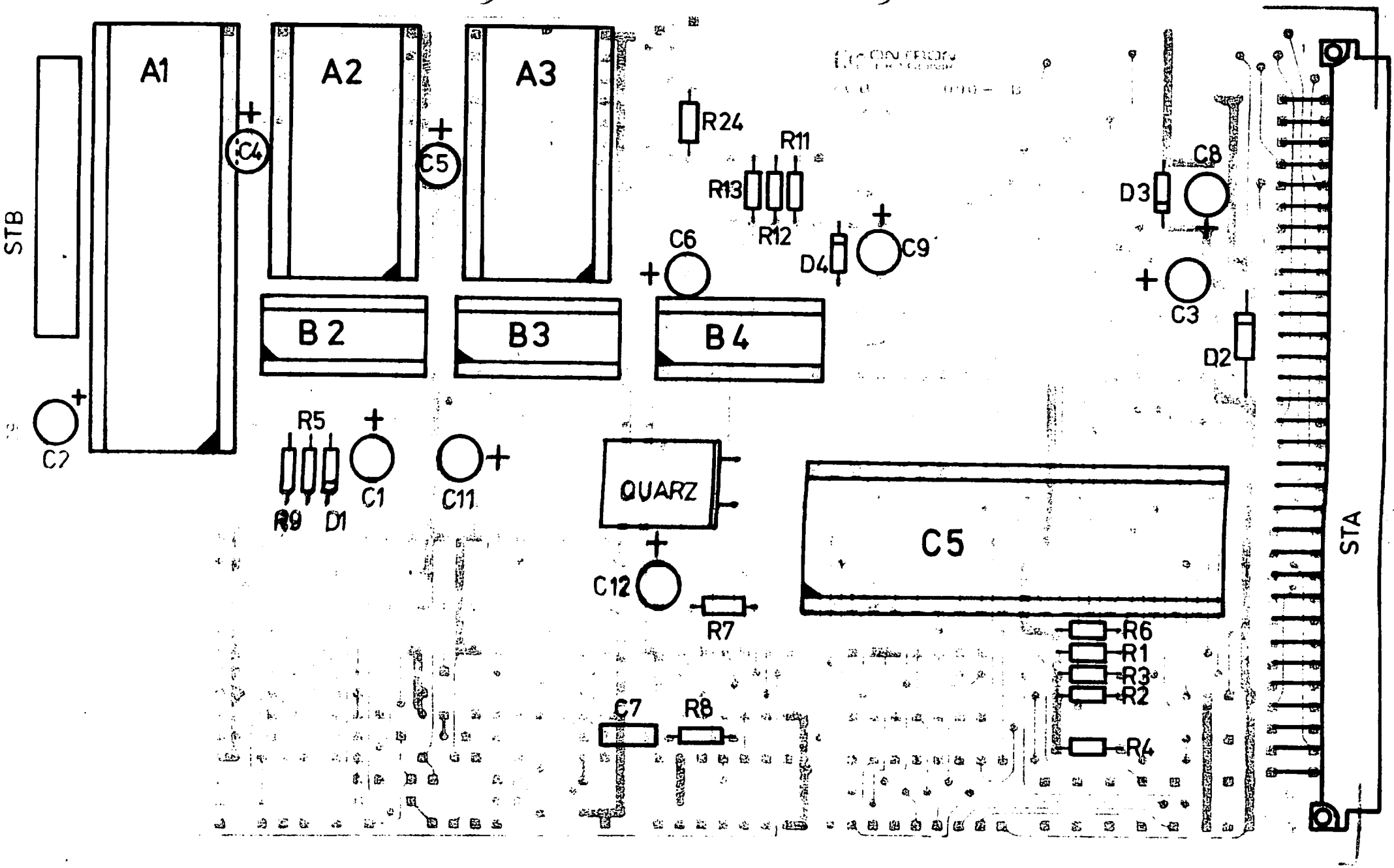

A1
A2
A3
STB
STA
C4
C5
C6
C7
C8
C9
C1
C2
C3
C11
C12
R24
R11
R13
R12
R5
R9
D1
D2
D3
D4
R7
R8
R6
R1
R3
R2
R4
B 2
B3
B 4
QUARZ
C5

Bild 9

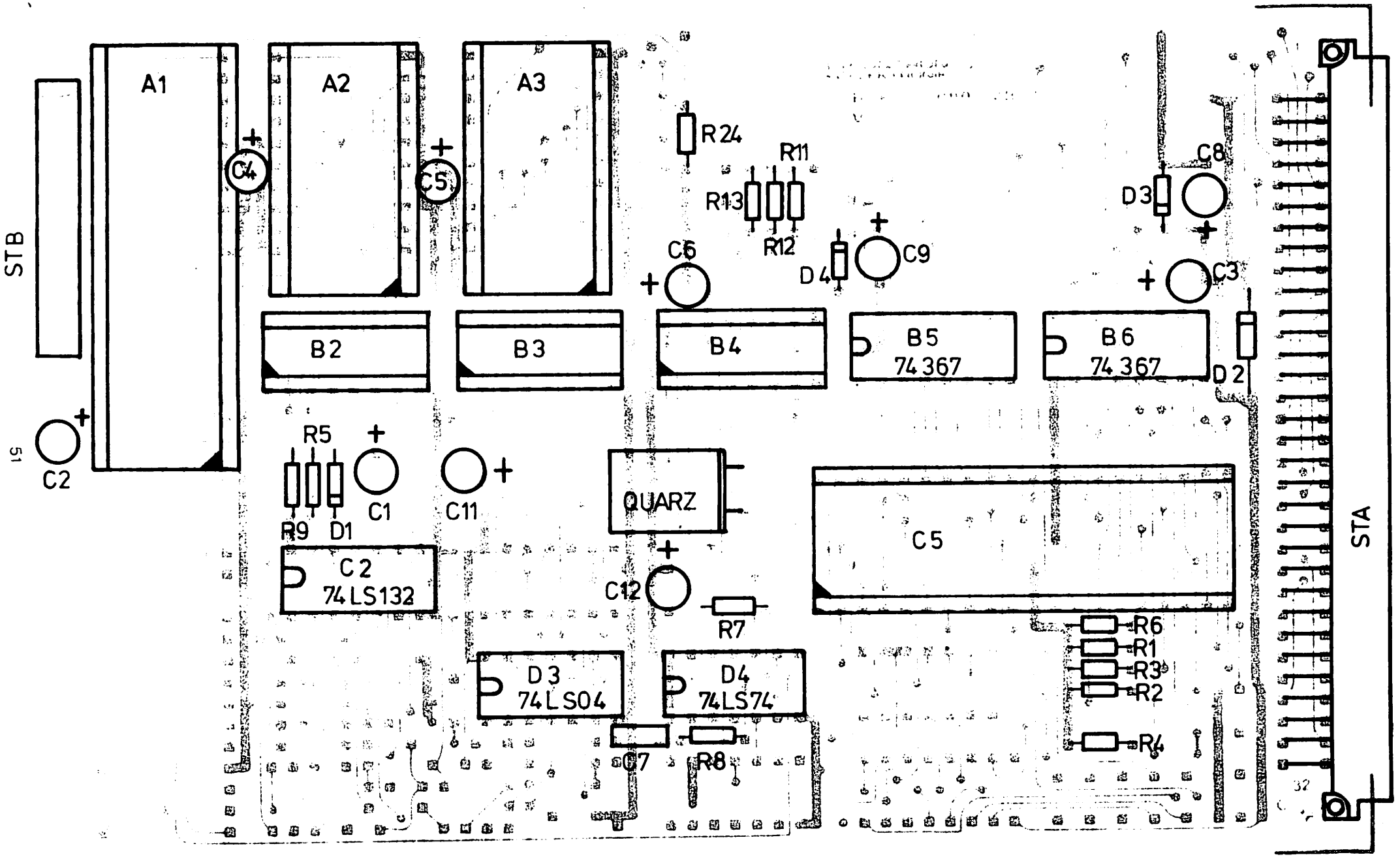

A1
A2
A3
STB
STA
C4
C5
C6
C8
C9
C3
C2
C1
C11
C12
R24
R11
R13
R12
D4
D3
D2
B2
B3
B4
B5
74367
B6
74367
R5
R9
D1
QUARZ
C5
C2
74LS132
R7
R6
R1
R3
R2
R4
D3
74LS04
D4
74LS74
C7
R8
51
32

Bild 10

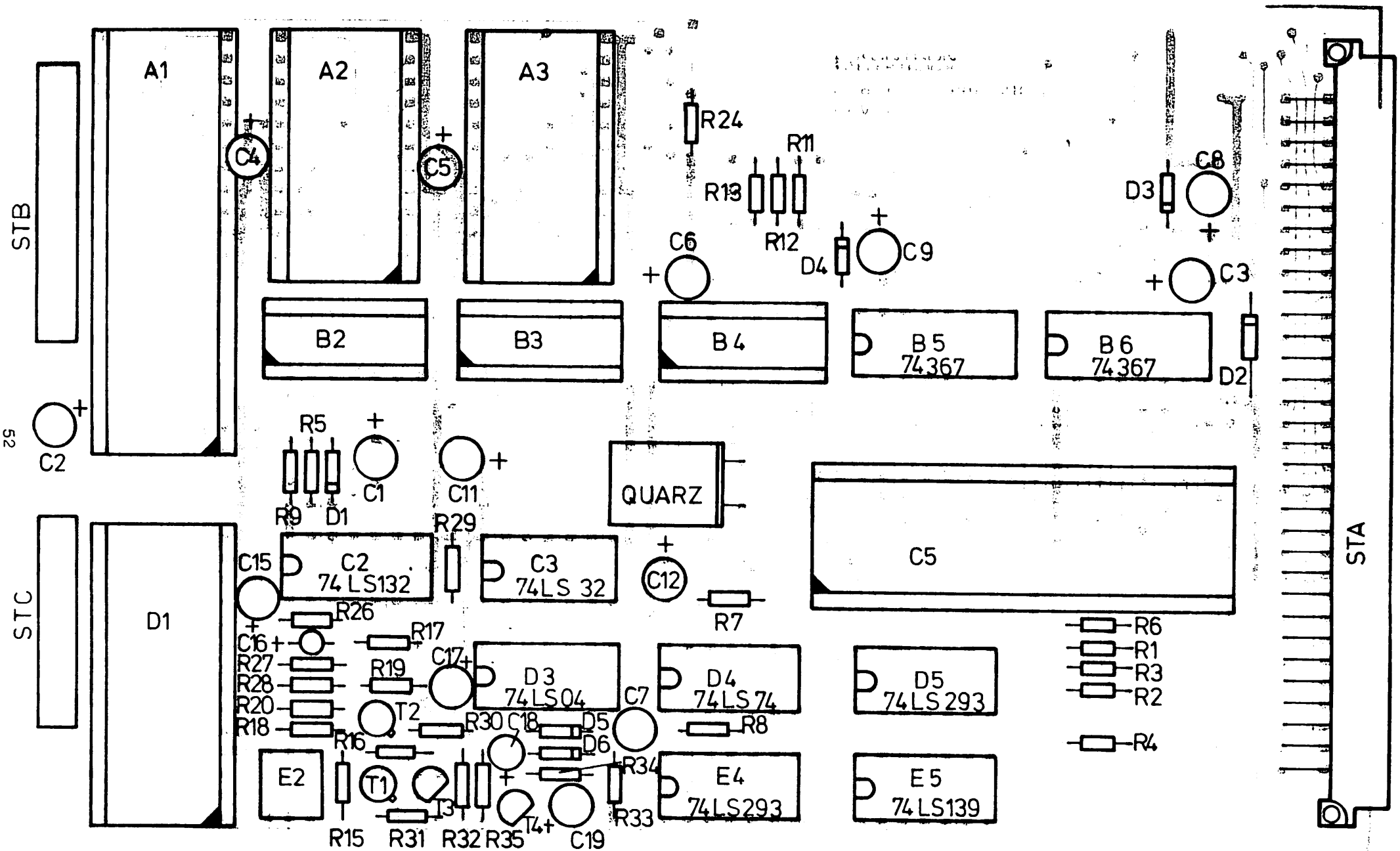

A1
A2
A3
STB
STC
STA
C4
C5
C6
C2
C1
C11
C15
C16
C17
C7
C8
C9
C3
C12
R24
R11
R13
R12
D4
D3
D2
B2
B3
B4
B5
74367
B6
74367
R5
R9
D1
R29
QUARZ
C5
C2
74LS132
C3
74LS 32
R7
R6
R1
R3
R2
R4
R26
R17
R27
R19
R28
R20
R18
T2
R30
C18
D5
D6
R16
R8
D3
74LS04
D4
74LS74
D5
74LS293
R34
R33
E2
T1
T3
T4
R15
R31
R32
R35
C19
E4
74LS293
E5
74LS139

Bild 11

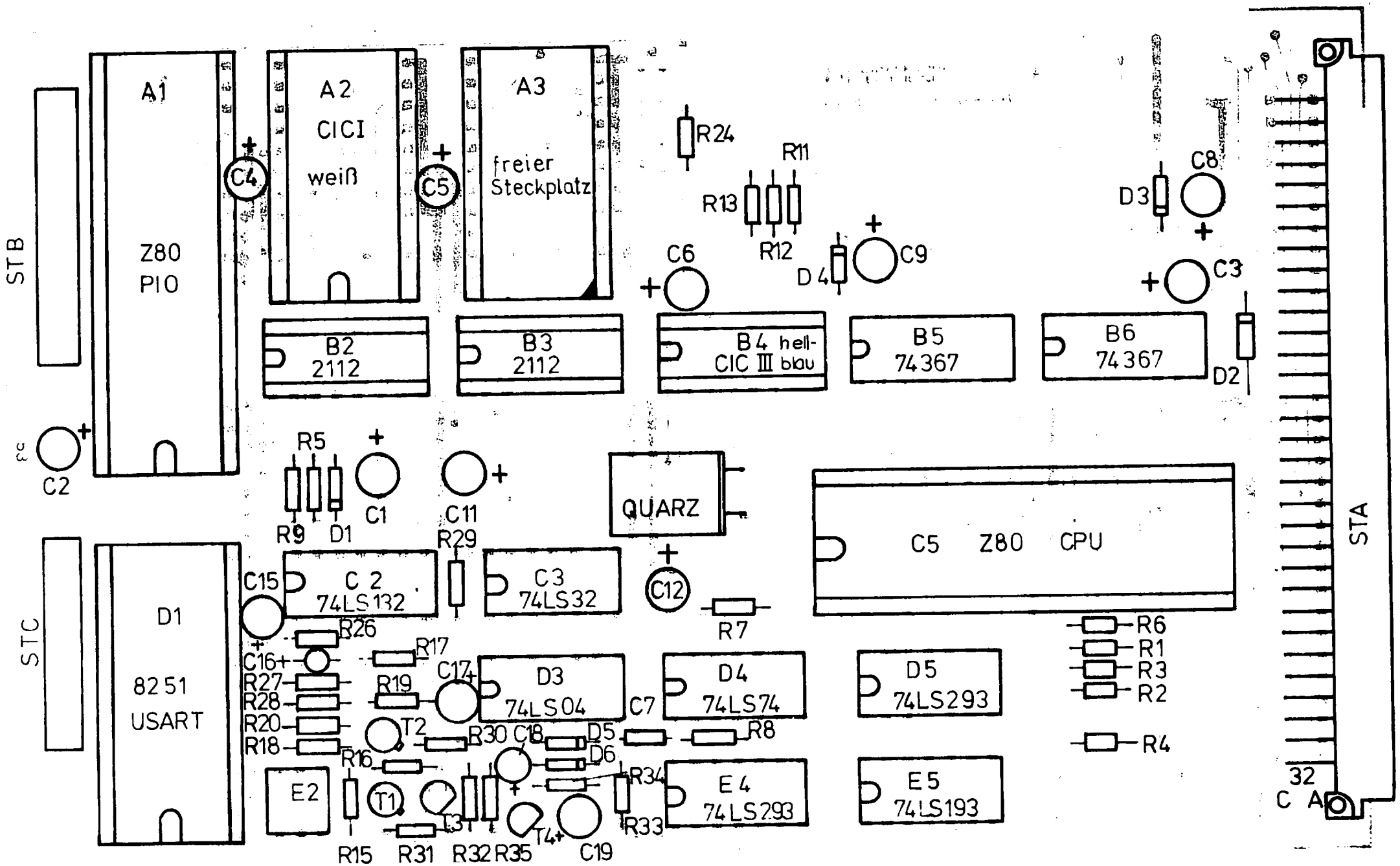

A1
Z80
PIO
A2
CICI
weiß
A3
freier
Steckplatz
STB
STC
STA
B2
2112
B3
2112
B4 hell-
CIC III blau
B5
74367
B6
74367
D1
8251
USART
C 2
74LS132
C3
74LS32
C5 Z80 CPU
QUARZ
D3
74LS04
D4
74LS74
D5
74LS293
E2
E4
74LS293
E5
74LS193
R24
R11
R13
R12
D4
C9
C6
C8
D3
C3
D2
R5
R9
D1
C1
C11
R29
C15
C16
R26
R17
R27
R19
C17
R28
R20
R18
T2
R30
C18
D5
R8
R16
D6
R34
T1
T3
T4
R33
R15
R31
R32
R35
C19
C7
C12
R7
R6
R1
R3
R2
R4
C2
32
C
A

Bild 12

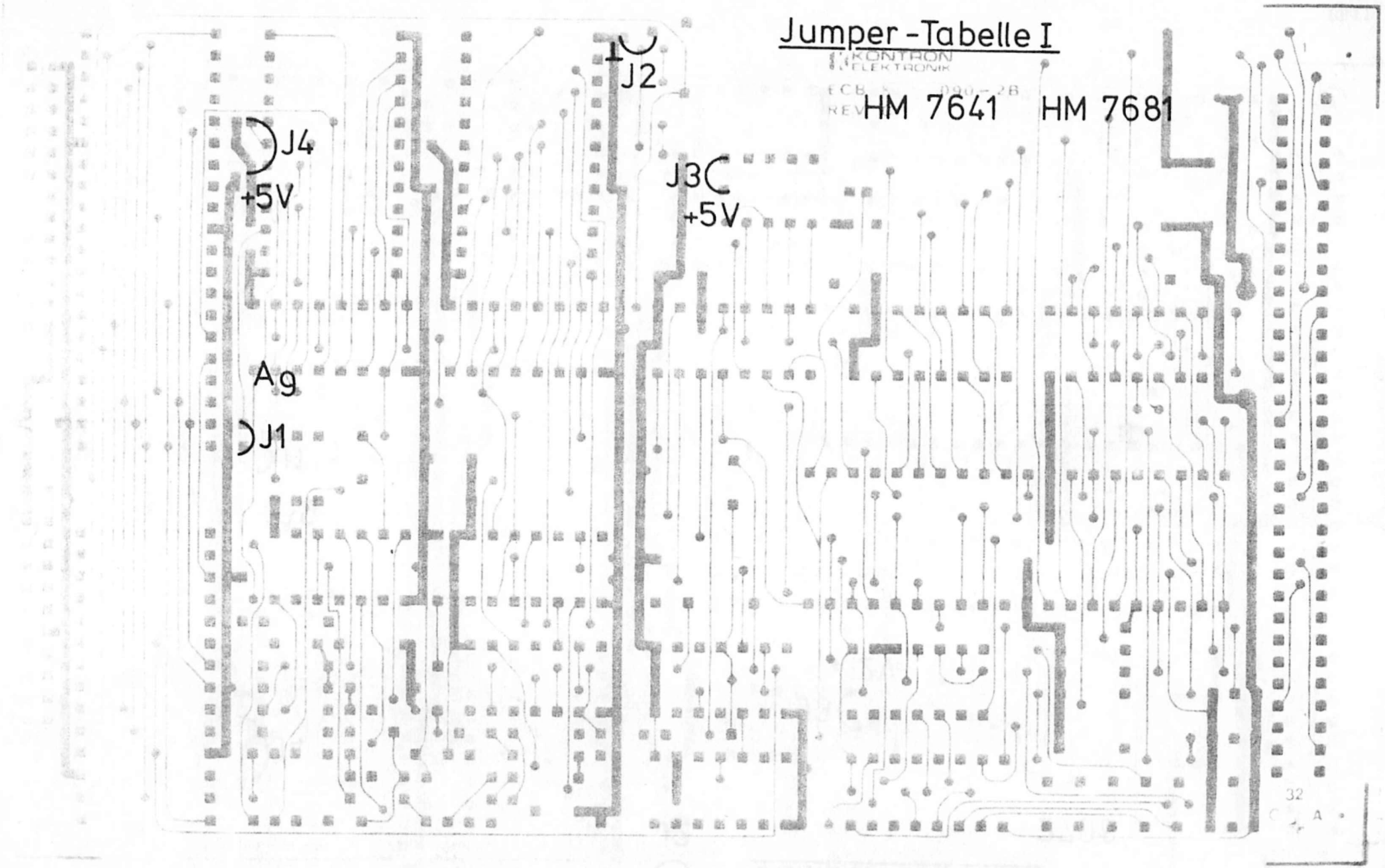


Bild 13

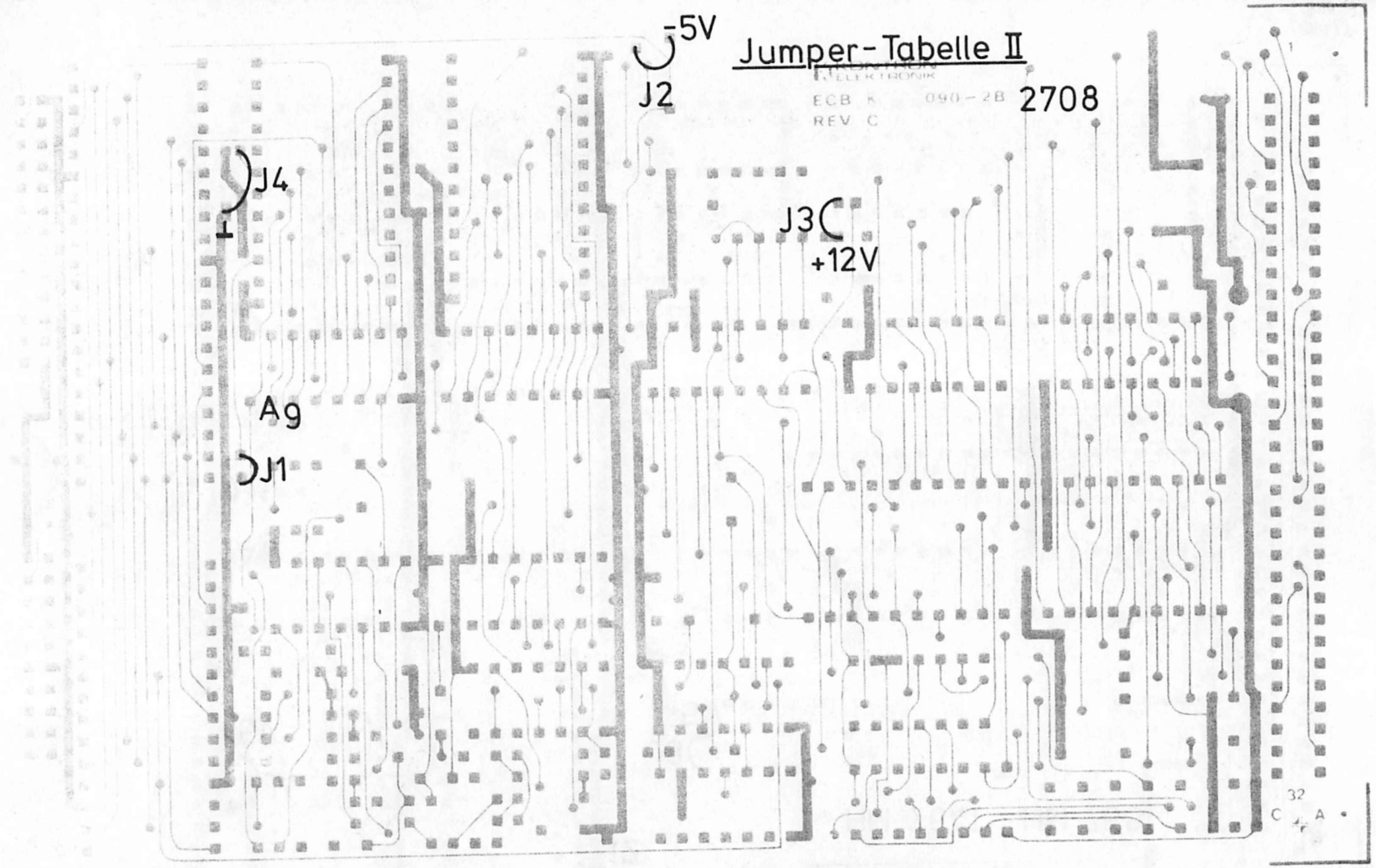

-5V
J2
Jumper-Tabelle II
ECB K 090-2B
REV C
2708
J4
J3
+12V
A9
J1
32
C
A

Bild 14

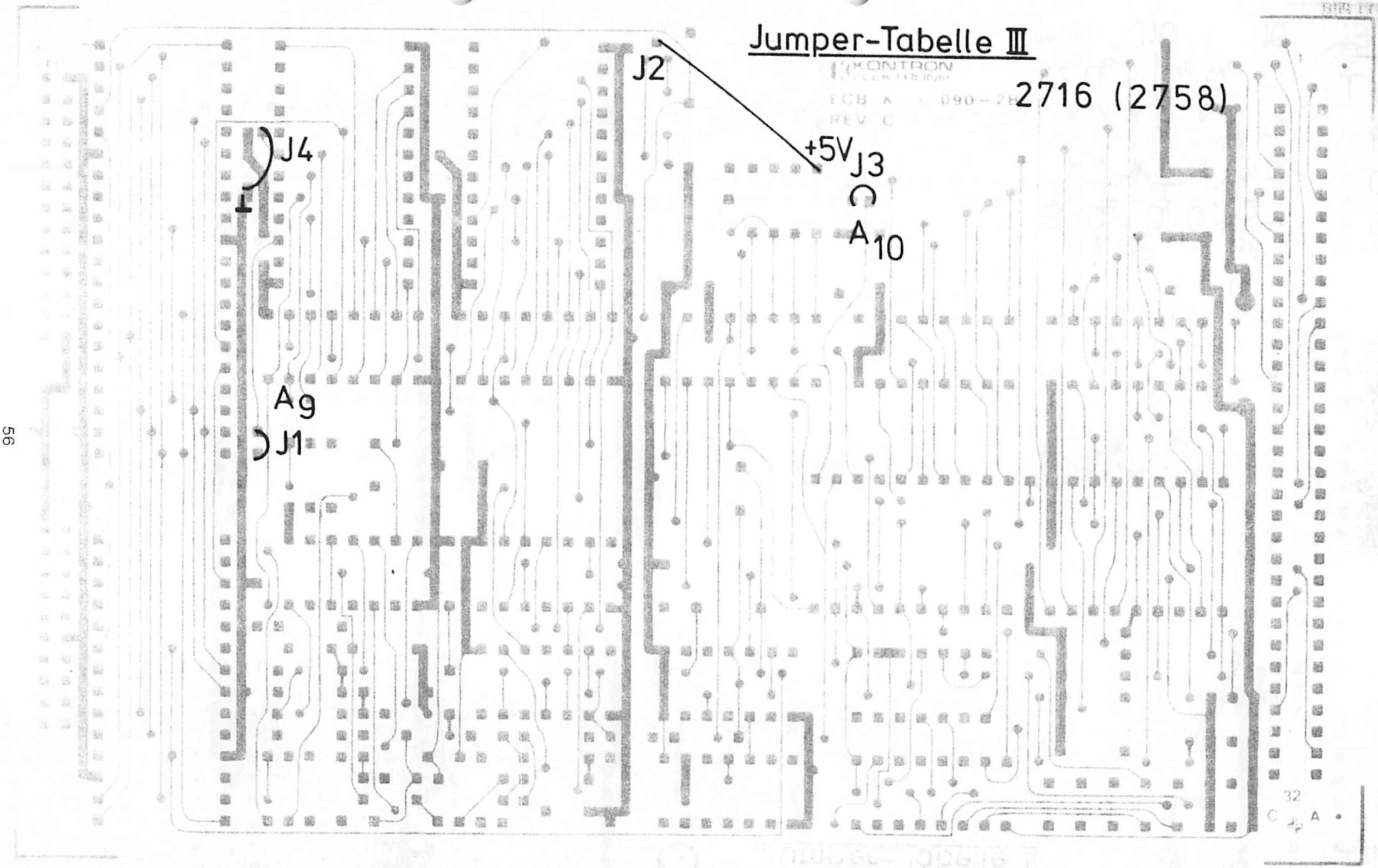


Bild 15

Jumper Tabelle IV

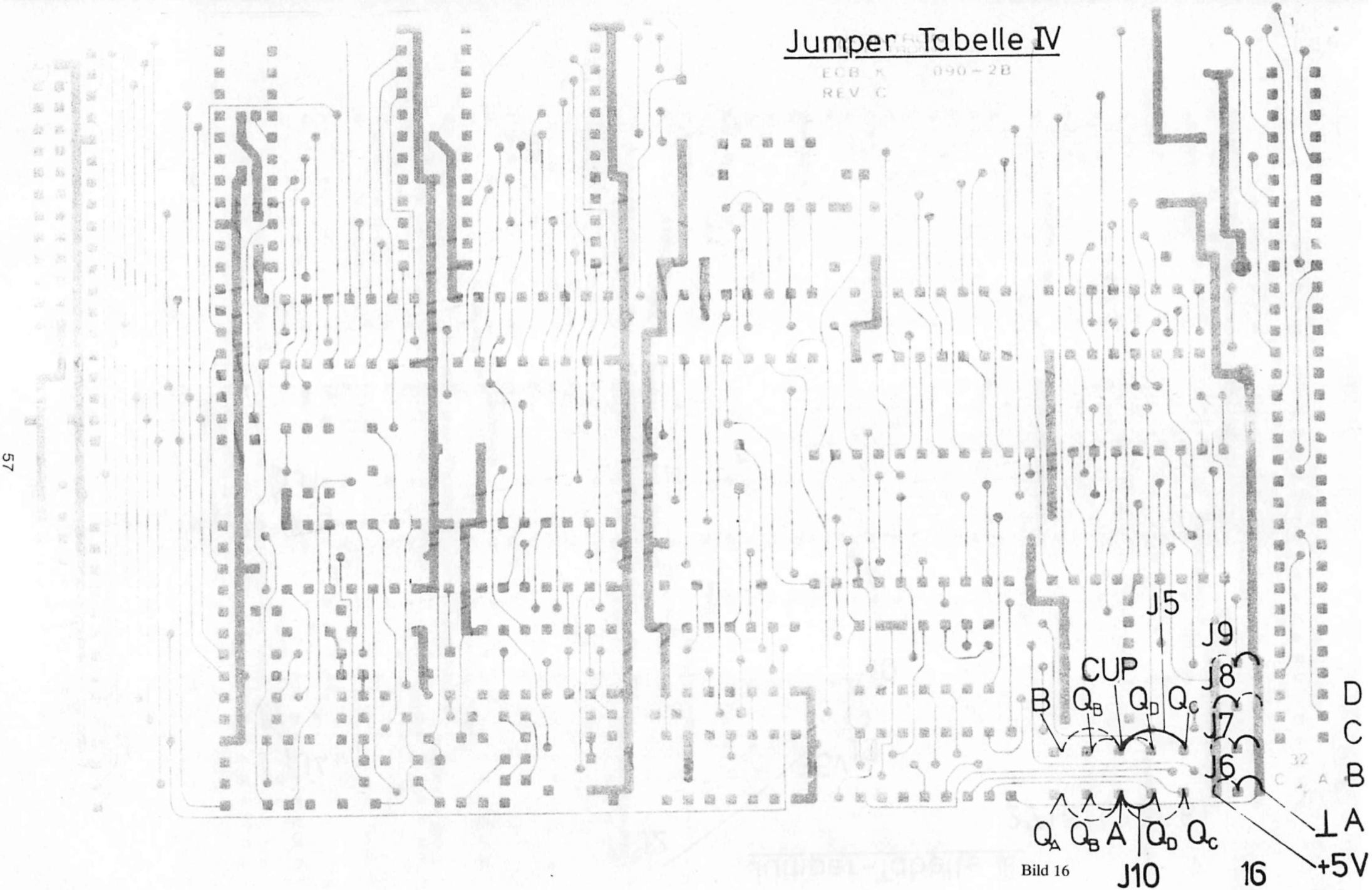


Bild 16

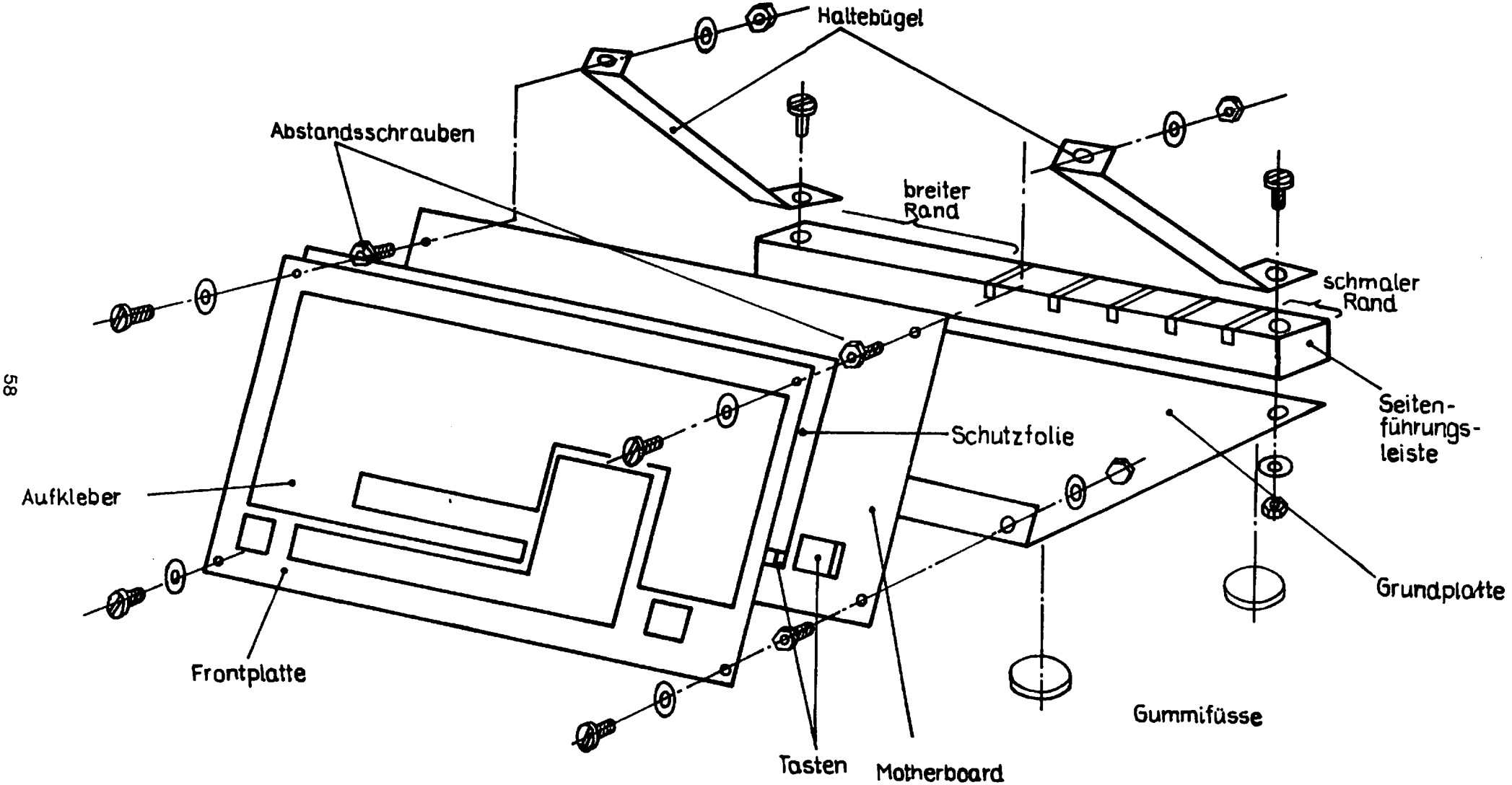

Haltebügel
Abstandsschrauben
breiter Rand
schmaler Rand
Seiten-führungs-leiste
Schutzfolie
Aufkleber
Grundplatte
Frontplatte
Gummifüsse
Tasten
Motherboard

Bild 17

6. Inbetriebnahme

6.1. Einschalten des Z80-KIT

Da Ihr Z80-KIT komplett aufgebaut ist, und Sie sich über die Bedienung klar geworden sind, steht der Inbetriebnahme nichts mehr im Wege.

☐ Schließen Sie jetzt Ihren KIT an die vorbereitete Stromversorgung an

☐ Drücken Sie die RESET Taste (ganz rechts, orange).

Die Anzeigen müssen jetzt alle den Wert Null zeigen.
Ihr KIT ist damit aktionsbereit.

6.2. Fehlersuchhinweise

Sollte Ihr erster Versuch nicht so erfolgreich verlaufen sein, schalten Sie bitte den Strom ab und führen Sie folgende Kontrollen aus:

1. Prüfen Sie, ob alle Widerstände richtig eingebaut sind, an Hand der einzelnen Bilder nach!
2. Stellen Sie fest, ob alle Kondensatoren richtig (Polarität!) eingebaut wurden.
3. Überprüfen Sie die Polarität der Dioden
4. Prüfen Sie ob alle IC's richtig eingebaut sind.
5. Sind die Mikrocomputerbausteine richtig eingesetzt?
6. Sind sämtliche Lötungen einwandfrei?
   (kalte Lötstellen oder Lötbrücken)
7. Suchen Sie nach Kurzschlüssen zwischen benachbarten Leiterbahnen. In mehr als 90 Prozent aller Fälle ist dies die Störungsursache.

7.1.3. Listing des Programms

```
                              TEST   LISTING                    PAGE   1
                                                                ASM 3.0
 LOC   OBJ CODE    STMT SOURCE STATEMENT

3C50                  1  ORG 3C50H
3C50   0E0C          2  LD C,0CH;          ADRESSE DES ANZ.ELEMENTES LADEN
3C52   110F00        3 NEU:LD DE 000FH;    ANZ.REG. MIT ANFANGSWERT LADEN
3C55   21723C        4 START:LD HL,SEGM;   ANFANGSADRESSE TABELLE LADEN
3C58   19            5  ADD HL,DE;        TABELLENADRESSE GENERIERT
3C59   7E            6  LD A,(HL);       INHALT TABELLE GELADEN
3C5A   060F          7  LD B,00FH;       SCHLEIFE 2 LADEN
3C5C   C5            8 S2:PUSH BC;       WERT IN DEN STAPELSPEICHER ABLEGEN
3C5D   06FF          9  LD B,0FFH;       SCHLEIFE 1 LADEN
3C5F   00           10  S1:NOP
3C60   ED79         11  OUT (C),A;       AUSGABE
3C62   10FB         12  DJNZ S1;         SCHL.1 DECR.,WENN NICHT 0,SPR.NACH S1
3C64   00           13  NOP
3C65   C1           14  POP BC;          WERT VON SCHL.2 AUS DEM STPL.SP.HOLEN
3C66   10F4         15  DJNZ S2;         SCHL.2 DECR.,WENN NICHT 0,SPR.NACH S2
3C68   3E00         16  LD A,0
3C6A   83           17  ADD A,E
3C6B   CA523C       18  JP Z,NEU;        WENN ANZEIGE 0,DANN SPRUNG NACHRT IN D
3C6E   1D           19  DEC E;           ANZ.REG. DECREMENTIEREN
3C6F   C3553C       20  JP START;        DAS GANZE VON VORNE MIT NEUER ZAHL
3C72   5F           21 SEGM:DEFB 05FH;   ANFANGSWERT TABELLE
3C73   06           22  DEFB 06H
3C74   3B           23  DEFB 3BH
3C75   2F           24  DEFB 2FH
3C76   66           25  DEFB 66H
3C77   6D           26  DEFB 6DH
3C78   7D           27  DEFB 7DH
3C79   07           28  DEFB 07H
3C7A   7F           29  DEFB 7FH
3C7B   6F           30  DEFB 6FH
3C7C   77           31  DEFB 77H
3C7D   7C           32  DEFB 7CH
3C7E   59           33  DEFB 59H
3C7F   3E           34  DEFB 3EH
3C80   79           35  DEFB 79H
3C81   71           36  DEFB 71H
                    37 END

    0 ASSEMBLY ERRORS
```

7.1.4. Eingabe und Starten des Programms

Setzen Sie den Befehlszähler (PC) auf den Wert 3C50 (damit haben Sie ausreichenden Speicherplatz für den Stapelspeicher und das nun folgende Programm) und gehen Sie in den Input-Mode. Tippen Sie nun das Programm ein. Die Zahlenfolge finden Sie im Listing unter OBJ CODE.

Sie tippen demnach:

0 E
0 C
1 1
0 F

usw. bis Sie bei 71 angelangt sind. Verlassen Sie den Input-Mode und setzen Sie den Befehlszähler wieder auf die Startadresse (=3C50).
Wenn jetzt die Taste Start gedrückt wird, läuft Ihr Programm ab, was sich durch ein entsprechendes Verhalten der Anzeige bemerkbar macht.
Ein Verlassen des Usersprogramms ist (in diesem Fall der unendl. Schleife) nur über einen Reset möglich.

7.2. Multiplikationsprogramm

Hier handelt es sich um eine arithmetische Rechenfunktion, die es dem Anwender ermöglicht, zwei hexadezimale Zahlen miteinander zu multiplizieren.

Im Gegensatz zum in 5.1 gezeigten Prüfprogramm, das in einer undendlichen Userprogrammschleife verbleibt, wird hier der Rücksprung ins Monitorprogramm ausgeführt. Dies hat zur Folge, daß ohne Reset sofort eine neue Aktivität ausgelöst werden kann.

7.2.1. Programmerläuterung

Bekanntlich verfügen heute handelsübliche 8bit-Mikroprozessoren lediglich über festverdrahtete 8bit-Additionsbefehle, etwas weiterentwickelte Maschinen wie Z80 auch über festverdrahtete 8- und 16bit-Additions- und Subtraktionsbefehle. Benötigt man in einem solchen Mikrocomputeranwendersystem andere Rechnungsarten, so muß man diese mit Hilfe von Software implementieren. Da die Vorgehensweise dabei für den Anfänger interessant ist, soll hier eine 8bit-Multiplikationsroutine angegeben und erläutert werden.
Als Register für den 8bit-Multiplikator wurde C gewählt; das Ergebnis kann 16bit lang sein, da die Multiplikation der zwei höchstmöglichen 8stelligen Zahlen $255 \times 255 = 65025$ groß werden kann. Aus diesem Grund wurde für Zwischenergebnis und Ergebnis ein 16bit-Registerpaar verwendet (HL). Ebenso mußte für den Multiplikanden ein Registerpaar (DE) reserviert werden, da das Verfahren ja darin besteht, den um eine Potenzstelle nach links verschobenen Multiplikanden sukzessive auf das Zwischenergebnisfeld aufzuaddieren.

7. Programmbeispiele

Um einen ersten Einstieg für den Gebrauch von USER-Programmen auf dem KIT zu geben, seien an dieser Stelle einige Programmbeispiele aufgeführt.

NB.:
Die Programme sind lediglich zum Erlernen der Programmierung, nicht als Selbstzweck oder Einsatz-System-Beispiele für den Z80-KIT gedacht.

Für jedes Anwenderprogramm ist vor dem Starten ein definierter Befehlszählerstand (= Startadresse) angegeben. Wird der Stapelspeicher (Stack) benutzt, kann die Voreinstellung 3CE0 oder ein beliebiger anderer Wert gewählt werden. Das Setzen des Stackpointers kann dabei auch vom Userprogramm softwaremäßig erfolgen. Der vom Anwenderprogramm aufgebaute Stack darf sich dabei nicht mit dem vom Monitorprogramm benutzten Stack überdecken. Dies wird z.B. durch die automatische Voreinstellung auf 3CE0H erreicht.

7.1. Prüfprogramm für den Z80-KIT

Um eine Möglichkeit zu schaffen, den KIT ohne eigenes und evtl. fehlerbchaftetes Userprogramm zu testen, sei an dieser Stelle ein kleines Programmbeispiel gegeben.
Das Programm besteht aus zwei ineinander verschachtelten Zählschleifen, an deren Enden eine Ausgaberoutine für eines der Hexadezimalen Anzeigeelemente steht. Ausgegeben wird der Inhalt von Registerpaar DE, der nach jedem Durchgang decrementiert wird.
Beginnend mit F werden so alle hexadezimalen Elemente in absteigender Folge nacheinander angezeigt. Nach 0 folgt wieder F, so daß sich eine unendliche Folge ergibt.

Dieses Programm macht keinen Gebrauch von der Möglichkeit, Monitorroutinen zu verwenden. Die Ansteuerung der LED Anzeige ist mit im Userprogramm enthalten.

7.1.1. Programmerläuterung

Überprüfen Sie das hier Gesagte anhand des Listings!
Wichtigster Teil des Testprogramms ist die Ausgaberoutine, die es ermöglicht, bestimmte Werte auf der Anzeige darzustellen. Diese Routine, die auch vom Monitorprogramm benutzt wird, basiert auf folgendem Gedanken:

Ein anzuzeigender Wert (hexadezimal!) wird in das DE Registerpaar eingelesen, wobei das D Register stets nur 0 enthalten darf. Das DE Registerpaar wird nun zum HL Registerpaar addiert (deshalb war es notwendig, das DE Registerpaar und nicht nur das E Register zu verwenden), welches die Anfangsadresse einer Tabelle (SEGM) enthält. Nach der Addition enthält das HL Register eine neue Adresse und zwar den Wert:

Anfangsadresse + Inhalt DE Registerpaar

Unter dieser Adresse ist der Wert abgespeichert, der bei Ausgabe auf die Anzeigeeinheit den im DE Registerpaar abgespeicherten Wert darstellt.
z.B.: Ist in DE der Wert 8 gespeichert, so ergibt sich:

	D	E
DE =	00	08

Im HL Register steht die Anfangsadresse der Tabelle SEGM:

	H	L
HL =	3C	72

Nach der Addition (hexadezimal) ergibt sich:

	H	L
HL =	3C	7A

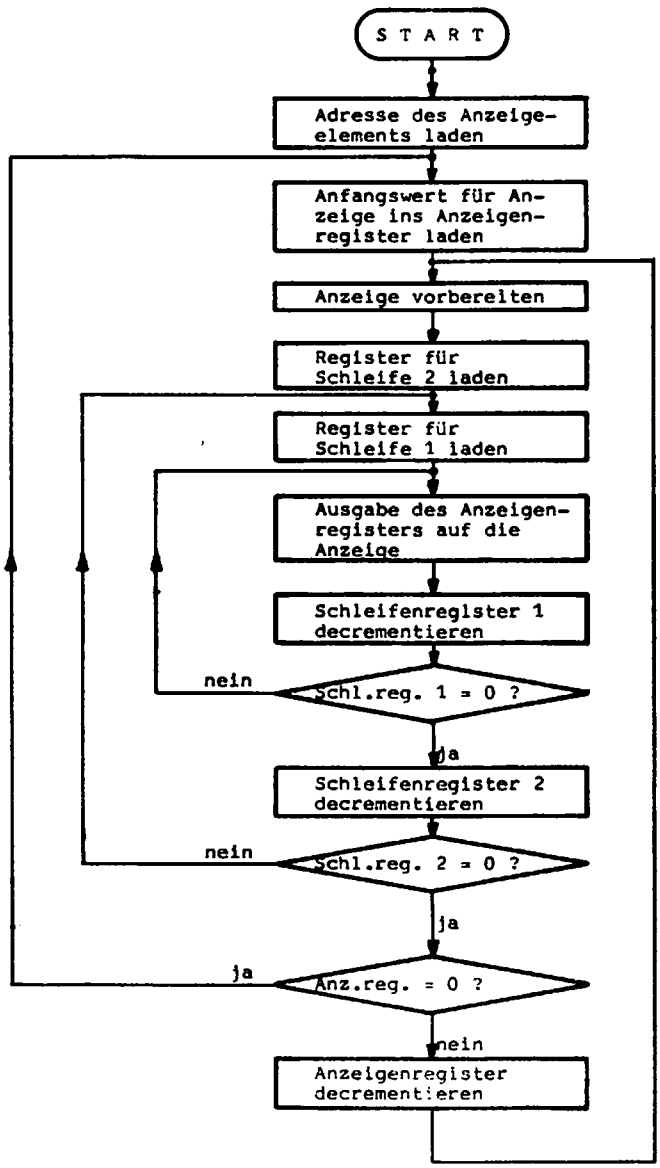
Unter dieser Adresse findet man den Wert

(3C 7A) = 7F

Bei Ausgabe dieses Wertes zeigt das Anzeigeelement den Wert 8.

Die Ansteuerung der einzelnen Anzeigeelemente geschieht mit Hilfe des Port-Registers C und einer zugehörigen Port-Tabelle. Soll die Ausgabe z.B. auf das obere Datenanzeigeelement erfolgen, ist C nicht mit 0C sondern mit 0D zu laden.
Die Zeitkonstanten sind jeweils im B Register untergebracht. Nach dem erstmaligen Laden mit dem Wert für Schleife 2 (0F), wird der Inhalt des B Registers in den Stack ausgelagert (und damit „konserviert"). B wird erneut geladen und zwar mit dem Wert für Schleife 1 (FF). Wird Wert I wieder gebraucht, so wird er aus dem Stack geholt.
Durch Veränderung der beiden Werte kann ein verschieden schneller Durchlauf der Anzeige erreicht werden.
Z.B.: Lädt man statt FF den Wert 0F, so ergibt sich ein wesentlich schnellerer Durchlauf.

7.1.2. Programmablaufplan



Selbstverständlich muß bei der Datenübergabe der Inhalt von D = 0 sein. Nach den erforderlichen Initialisierungen verschiebt man den 8bit-Multiplikator nach rechts. War das zu diesem Zeitpunkt im Register C stehende niederwertigste bit = 1, so wird das Überlaufbedingungsbit gesetzt und aufgrund einer darauffolgenden Abfrage „Carry = 1?" der Inhalt des Registerpaares DE zum Inhalt des (Zwischen-)Ergebnisregisterpaares HL addiert.

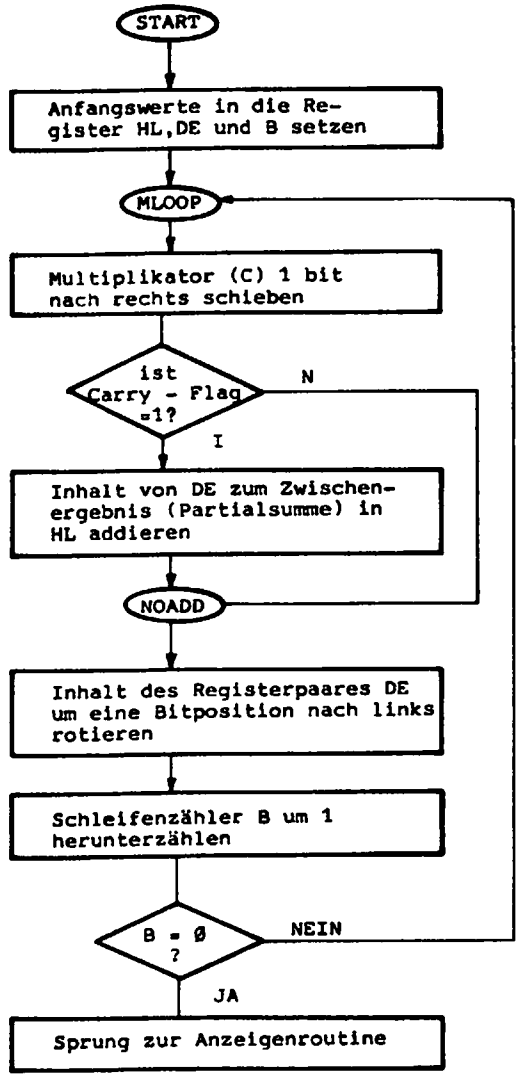
War das Carry-Flag = 0, so entfällt die Addition; in beiden Fällen wird jedoch der Inhalt von DE um eine Binärstelle erhöht, d.h. um ein bit nach links geschoben.

Am Ende des Durchlaufs wird geprüft, ob bereits der ganze Multiplikator abgearbeitet ist, d.h. ob das Durchschieben des Multiplikators 8mal stattgefunden hat. Ist der Zähler = 0, so steht im Registerpaar HL das fertige Multiplikationsergebnis, sonst ist der Programmablauf ab der Marke MLOOP zu wiederholen.

Die Codierung (s. Listing) ist recht einfach. Bemerkenswert die einfache Realisierung eines 16bit-Schiebevorganges mit Hilfe der zwei Befehle SLA r und RL r. Dabei wird das höchstwertige Bit des Registers E durch SLA E ins Carry-Flag geschoben und die unterste Stelle mit einer Null aufgefüllt.

RL D „holt" dann den Übertrag aus dem E-Register über das Carry-Flag ab.

Das ganze Programmstück kann selbstverständlich auch als Unterprogramm geschrieben werden.



7.2.2. Programmablaufplan

7.2.3. Listing des Programms

```
MULTIPLIKATIONSPROGRAMM        MULT                              PAGE   1
 LOC   OBJ CODE M STMT SOURCE STATEMENT                           ASM 4.5

                      1  #H MULTIPLIKATIONSPROGRAMM
                      2
                      3
3C50                  4          ORG 3C50H
3C50  210000          5          LD HL,0          ;ERGEBNISREGISTER LOESCHEN
3C53  1600            6          LD  D,0          ;D ERGISTER LOESCHEN
3C55  3A403C          7          LD  A,(3C40H)    ;MULTIPLIKAND INS E REG.
3C58  5F              8          LD E,A           ;LADEN
3C59  3A413C          9          LD  A,(3C41H)    ;MULTIPLIKATOR INS C REG.
3C5C  4F             10          LD C,A           ;LADEN
3C5D  0608           11          LD  B,8          ;ACHT DURCHLAEUFE
                     12
                     13
3C5F  CB39           14  MLOOP:  SRL C            ;UNTERSTES BIT VON C INS CARRY
3C61  3001           15          JR NC,NOADD      ;WENN 0,KEIN AUFADDIEREN
3C63  19             16          ADD HL,DE
                     17
                     18
3C64  CB23           19  NOADD:  SLA E
3C66  CB12           20          RL  D            ;16 BIT SCHIEBEN ENDE
3C68  10F5           21          DJNZ MLOOP       ;WENN NOCH KEINE ACHT
                     22                           ;DURCHGAENGE ZURUECK!
3C6A  22183C         23          LD (3C18H),HL    ;HL REGISTERPAAR RETTEN
3C6D  210B00         24          LD HL,000BH
3C70  22FE3C         25          LD (3CFEH),HL    ;RUECKKEHRADR. FUER JP 0148
                     26                           ;ANGEBEN
3C73  C34801         27          JP 0148H         ;RUECKSPRUNG IN DEN MONITOR
```

7.2.4. Eingabe und Starten des Programms

Setzen Sie wie in Programm 1 den Befehlszähler auf 3C50. Geben Sie in den Input-Mode und tippen Sie das Programm ein. Die Folge der einzugebenden Werte steht im Listing unter OBJ CODE.
Sie tippen demnach:

21
00
00
16
usw.

bis Sie bei 0 1 angelangt sind.
Sie haben damit das Multiplikationsprogramm geladen.
Die Eingabe von Multiplikator und Multiplikand erfolgt nun auf die folgenden Speicheradressen:

3C40 Multiplikand
3C41 Multiplikator.

Setzen Sie jetzt den Befehlszähler auf die Startadresse 3C50 und drücken Sie die Start-Taste. Das Ergebnis im HL Registerpaar kann sofort über das DISPLAY Kommando sichtbar gemacht werden. Durch den im Programm MULT eingebauten Rücksprung ins Monitorprogramm können sofort zwei neue Werte eingegeben und miteinander multipliziert, oder irgend eine andere Funktion ausgeführt werden.

Beispiel 1

Gelöst werden soll die Aufgabe $4 \times 13 = ?$
Zuerst ist eine Umwandlung in das hexadezimale Format notwendig.

Zuordnungstabelle

dezimal	hexadezimal
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	A
11	B
12	C
13	D
14	E
15	F

Umwandlung zwischen Dezimalzahlen und Hexadezimalzahlen

Achtung: Diese kleine Anleitung soll lediglich eine kleine Hilfe bei der Umwandlung zwischen den beiden Formaten sein. Sollten noch keine Erfahrungen auf diesem Gebiet vorhanden sein, wird zusätzliche Literatur dringend empfohlen!

Während es sich bei einer Dezimalzahl um eine Summe von Potenzen zur Basis 10 handelt, ist eine hexadezimale Zahl als Summe der Potenzen zur Basis 16 zu verstehen.

Beispiel:

dez.: $4123 = 4 \times 10^3 + 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0$
hex.: $12D4 = 1 \times 16^3 + 2 \times 16^2 + 13 \times 16^1 + 4 \times 16^0$

Beispiel für Umwandlung hex → dezimal:

21 A (hex) $= 2 \times 16^2 + 1 \times 16^1 + 10 \times 16^0$
$= 512 + 16 + 10 = 538$ (dez)

Beispiel für Umwandlung dez. → hex.

421 (dez) : $16^3 = 0$
421 : $16^2 = 1$
256
Rest 165 : $16^1 = 10$
160
Rest 5 : $16^0 = 5$
421 (dez) = 1A5 (hex)

Für unser Beispiel ergibt sich:

dezimal		hexadezimal
4	→	4
13	→	D

Die Multiplikation lautet in hexadezimaler Form demnach

$4 \times D$

Da sowohl Multiplikator wie auch Multiplikand in 8 bit Register eingelesen werden, ist ein Auffüllen der restlichen bits mit dem Wert 0 erforderlich:

$04 \times 0D$

Diese beiden Werte werden in die Speicheradressen 3C40 und 3C41 eingegeben.
Das Ergebnis der Operation lautet

0034

und ist natürlich als Hexzahl zu verstehen.
Während die willkürlich gewählte dezimale Zahl 7D36 als

6 Einer = 6×10^0
3 Zehner = 3×10^1
0 Hunderter = 0×10^2
7 Tausender = 7×10^3
Summe 7036

dargestellt werden kann (Potenzen zur Basis 10), wird die **hexadezimale** Zahl 0034 wie folgt ausgewertet

$4 \times 16^0 = 4 \times 1 = 4$
$3 \times 16^1 = 3 \times 16 = 48$
$0 \times 16^2 = 0 \times 256 = 0$
$0 \times 16^3 = 0 \times 4096 = 0$

(Potenzen zur Basis 16) Summe 52 (dez)
Als Summe ergibt sich der erwartete Wert $48 + 4 = 52$ (dez)

Beispiel 2

dez: 123 50 =
↓ $7 \times 16^1 + 11 \times 16^0$ × $3 \times 16^1 + 2 \times 16^0$
hex: 7 B × 3 2 =
hex: 1806
=
↓ $1 \times 16^1 + 8 \times 16^2 + 0 \times 16^4 + 6 \times 16^0$ =

dez: 6150

7.2.5. Erweiterung des Multiplikationsprogramms

Das Multiplikationsprogramm soll nun derart erweitert werden, daß nach erfolgter Berechnung das Ergebnis bei ADDRESS (16 bit Wert!) angezeigt wird, und daß durch das kurzzeitige Austasten der Anzeige die Beendigung der Operation angezeigt wird.
Dazu ist es notwendig, das Ergebnis in die LED-Puffer zu überführen. Die Monitor-Routine RESPE 2 bringt den Inhalt von zwei Speicherzellen (= 2×8 bit) in die LED-Puffer der ADDRESS-Anzeige (= 4×4 bit). Es ist also lediglich notwendig, das HL-Registerpaar im Speicher abzulegen und anschließend RESPE 2 aufzurufen. Ein Rücksprung in den Monitor erfolgt diesmal ohne Aktualisierung der Userwerte bei 000BH.

7.2.6. Listing des erweiterten Multiplikationsprogramms

```
MULTIPLIKATIONSPROGRAMM MIT   MULT1                                  PAGE   1
 LOC   OBJ CODE M STMT SOURCE STATEMENT                               ASM 4.5

                       1  *H MULTIPLIKATIONSPROGRAMM MIT AUSGABE
                       2
                       3
3C50                   4          ORG 3C50H
3C50   210000          5          LD HL,0          ;ERGEBNISREGISTER LOESCHEN
3C53   1600            6          LD  D,0          ;D ERGISTER LOESCHEN
3C55   3A403C          7          LD  A,(3C40H)    ;MULTIPLIKAND INS E REG.
3C58   5F              8          LD E,A           ;LADEN
3C59   3A413C          9          LD  A,(3C41H)    ;MULTIPLIKATOR INS C REG.
3C5C   4F              10         LD C,A           ;LADEN
3C5D   0608            11         LD  B,8          ;ACHT DURCHLAEUFE
                       12
                       13
3C5F   CB39            14 MLOOP:  SRL C            ;UNTERSTES BIT VON C INS CARRY
3C61   3001            15         JR NC,NOADD      ;WENN 0,KEIN AUFADDIEREN
3C63   19              16         ADD HL,DE
                       17
                       18
3C64   CB23            19 NOADD:  SLA E
3C66   CB12            20         RL  D            ;16 BIT SCHIEBEN ENDE
3C68   10F5            21         DJNZ MLOOP       ;WENN NOCH KEINE ACHT
                       22                          ;DURCHGAENGE ZURUECK!
                       23
                       24
3C6A   22423C          25         LD (3C42H),HL    ;HL REGISTERPAAR IM SPEICHER
                       26                          ;ABLEGEN
                       27
3C6D   F5              28         PUSH AF
3C6E   C5              29         PUSH BC
3C6F   E5              30         PUSH HL
3C70   DDE5            31         PUSH IX
3C72   FDE5            32         PUSH IY          ;VON RESPE2 UND ZEITKO BENUTZTE
                       33                          ;REGISTER IN DEN STACK GERETTET
                       34
3C74   FD21423C        35         LD IY,3C42H      ;UEBERGABEPARAMETER = HERKUNFTS-
                       36                          ;ADRESSE DER ANZUZEIGENDEN DATEN
                       37
3C78   CD8E03          38         CALL 038EH       ;AUFRUF VON RESPE2
                       39
3C7B   CDCA03          40         CALL 03CAH       ;AUFRUF VON ZEITKO
                       41
3C7E   FDE1            42         POP IY
3C80   DDE1            43         POP IX
3C82   E1              44         POP HL
3C83   C1              45         POP BC
3C84   F1              46         POP AF           ;GERETTETE REGISTER ZURUECK
                       47
3C85   C30B00          48         JP 000BH         ;RUECKSPRUNG IN DEN MONITOR
```

7.3.Reaktionstestprogramm

Hierbei handelt es sich um ein Spielprogramm zum Testen der Reaktionszeit. Nach Drücken einer beliebigen Taste erzeugt ein Zufallsgenerator softwaremäßig eine bestimmte Zeitdauer, nach deren Ablauf die ERROR-Lampe aufleuchtet. Die Zeit vom Aufleuchten bis zum Drücken einer beliebigen Taste wird gemessen, und direkt in Hundertstel Sekunden dezimal angezeigt.

7.3.1. Programmbeschreibung

Das Programm besteht aus einer Initialisierung, einer Warteschleife, dem Zufallszeitgenerator, einer Zeitschleife und einer Ausgaberoutine.

In fast allen Teilen werden Monitorroutinen verwendet. So benützt z.B. die Warteschleife (Listing STMT 17 bis 22) die Anzeigenroutine LED 2 und die Tastenabfrage die Routine OBTAST, wodurch der Programmieraufwand erheblich verringert wird.

Interessant ist die Verwirklichung der Zufallszeit (Listing STMT 25 bis 36). Hauptgedanke dabei ist der Zugriff auf das Refresh-Register der CPU. Es enthält die jeweils aktuelle Refreshadresse (8 bit!) und wird bei jedem M1-Zyklus incrementiert. Der augenblickliche Wert ist bezogen auf den Zeitpunkt irgendeinen Ereignisses wie das Drücken einer Taste rein zufällig.

Der Inhalt des R-Registers wird bei Start eines Durchlaufes entnommen und in die Register B und H geladen. Daraufhin wird eine zweifach verschachtelte Zählschleife durchlaufen, nach deren Ende die ERROR-Lampe zum Leuchten gebracht wird.

Die Zeitmessung erfolgt durch wiederholtes Abarbeiten eines, auf eine Hundertstel Sekunde abgestimmten Programmstückes, an dessen Ende das Ergebnisregister incrementiert wird (Listing STMT 47 bis 53). Die Abstimmung erfolgt dabei durch einen Programmteil (STMT 71 bis 82), der mit Hilfe einer Zählschleife auf die gewünschte Länge eingestellt ist.

Die Ausgabe des Ergebnisses wird dezimal im BCD Format durchgeführt. Eine Umrechnung von hexadezimalen Werten (mit denen die CPU arbeitet) in dezimale wurde mit Hilfe des Befehls DAA (Decimal Adjust Accumulator) durchgeführt. Dieser Befehl führt nach Addition oder Subtraktion zweier Werte eine Korrektur im Akkumulator aus, die einen hexadezimalen Wert in einen dezimalen (im BCD Format) umrechnet.

z.B.

Wert 1 = 08 dezimal
Wert 2 = 13 dezimal

Nach Addition der Werte ergibt sich, da die Maschine **nicht** dezimal addiert:

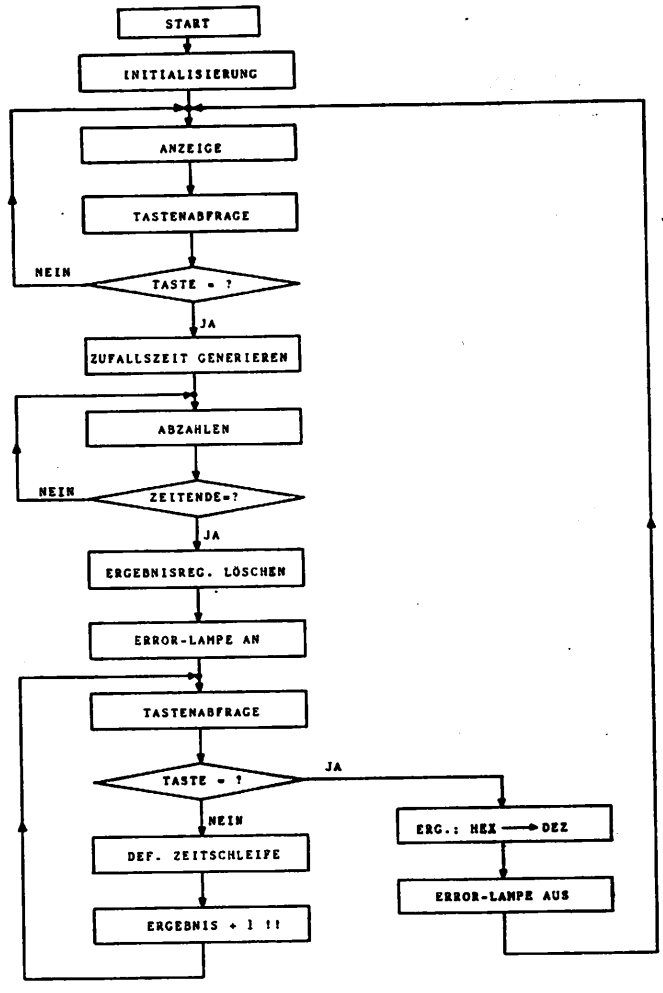
$$08 + 13 = 1B$$

Gibt man nun den Befehl DAA, wird der eigentlich erwartete, dezimale Wert generiert.

```
1B    DAA    21
```

Soll, wie im vorliegenden Fall, ein Ergebnis, das in hexadezimaler Form vorliegt in dezimale Form umgewandelt werden, so kann ebenfalls der DAA verwendet werden. Da der DAA jedoch nur nach Addition bzw. Subtraktionen angewandt werden darf, muß zuvor eine Pseudoverknüpfung durchgeführt werden. Durch Addition des Wertes 0 kann diese Forderung z.B. erfüllt werden.

7.3.2. Programmablaufplan



7.3.3. Listing des Reaktionstestprogramms

```
REAKTIONSTEST                    REACT                          PAGE   1
 LOC   OBJ CODE M STMT SOURCE STATEMENT                          ASM 4.5

                      1  *H REAKTIONSTEST
                      2
3C50                  3          ORG 3C50H
                      4
3C50  CDCA03          5          CALL 03CAH        ;AUFRUF ZEITKO
3C53  CDCA03          6          CALL 03CAH        ;WEGEN TASTENPRELLEN
3C56  210000          7          LD HL,0
3C59  22403C          8          LD (3C40H),HL     ;00 IM SPEICHER
3C5C  FD21403C        9          LD IY,3C40H       ;UEBERGABEPARAMETER
3C60  CD8E03         10          CALL 038EH        ;AURUF VON RESPE2
                     11                            ; = ADDRESS ANZEIGE GELOESCHT
3C63  21423C         12          LD HL,3C42H       ;ADRESSE DER ANZEIGEMARKE
3C66  CBC6           13          SET 0,(HL)        ;ANZEIGE MIT KONVERSION !
3C68  E5             14          PUSH HL           ;HL (MARKE) RETTEN
                     15
                     16
                     17  WARTEN:
3C69  E1             18          POP HL            ;HL ZURUECK
3C6A  CD1703         19          CALL 0317H        ;AUFRUF LEDS2
3C6D  E5             20          PUSH HL           ;HL (MARKE) RETTEN
3C6E  CD5102         21          CALL 0251H        ;AUFRUF OBTAST
3C71  28F6           22          JR Z,WARTEN       ;WENN KEINE TASTE
                     23
                     24
                     25  ZEIT1:
3C73  ED5F           26          LD A,R            ;INHALT R = ZUFAELLIGE ZAHL !
3C75  47             27          LD B,A
3C76  67             28          LD H,A
3C77  110100         29          LD DE,1
                     30
                     31
3C7A  78             32  MARK1:  LD A,B
3C7B  D601           33  MARK2:  SUB 1             ;A UM 1 ERNIEDRIGEN
3C7D  20FC           34          JR NZ,MARK2       ;WENN NOCH NICHT 0 ZURUECK !
3C7F  ED52           35          SBC HL,DE
3C81  20F7           36          JR NZ,MARK1       ;WENN NOCH NICHT 0 ZURUECK !
                     37
                     38
3C83  CDCA03         39          CALL 03CAH        ;AUFRUF ZEITKO
3C86  CDCA03         40          CALL 03CAH        ;WEGEN TASTENPRELLEN
                     41
                     42
3C89  01FFFF         43          LD BC,0FFFFH
3C8C  CDFF02         44          CALL 02FFH        ;AUFRUF ERROR (LAMPE GEHT AN)
                     45
                     46
                     47  REAK:
3C8F  CDB83C         48          CALL DELAY        ;DEFINIERTE ZEITKONSTANTE
3C92  04             49          INC B             ;ERGEBNISREGISTER HOCHZAEHLEN
3C93  C5             50          PUSH BC           ;BC RETTEN DA ANSCHLIESSEND OBTAST
3C94  CD5102         51          CALL 0251H        ;AUFRUF OBTAST
3C97  C1             52          POP BC            ;BC ZURUECK
3C98  28F5           53          JR Z,REAK         ;WENN NOCH KEINE TASTE,ZURUECK !
                     54
                     55
3C9A  78             56          LD A,B            ;ERGEBNIS NACH A
3C9B  C600           57          ADD A,0           ;PSEUDO OPERATION FUER DAA
3C9D  27             58          DAA               ;ERGEBNIS JETZT IN BCD FORMAT
```

```
3C9E   6F          59          LD L,A          ;INS L REGISTER
3C9F   2600        60          LD H,0          ;H LOESCHEN
3CA1   CB14        61          RL H            ;EVTL.UEBERTRAG VON DAA INS H
3CA3   22403C      62          LD (3C40H),HL   ;ERGEBNIS IM SPEICHER
3CA6   FD21403C    63          LD IY,3C40H     ;UEBERGABEPARAMETER
3CAA   CD8E03      64          CALL 038EH      ;AUFRUF VON RESPE2
3CAD   CD0303      65          CALL 0303H      ;AUFRUF FEHLEX (LAMPE AUS)
3CB0   CDCA03      66          CALL 03CAH      ;AUFRUF ZEITKO
3CB3   CDCA03      67          CALL 03CAH      ;WEGEN TASTENPRELLEN
3CB6   18B1        68          JR WARTEN       ;DAS GANZE VON NEUEM !
                   69
                   70
                   71  DELAY:                  ;DEFINIERTE ZEITKONSTANTE
3CB8   F5          72          PUSH AF
3CB9   C5          73          PUSH BC
3CBA   018C04      74          LD BC,1164
                   75  LO1:
3CBD   0B          76          DEC BC
3CBE   78          77          LD A,B
3CBF   B1          78          OR C
3CC0   20FB        79          JR NZ,LO1
3CC2   C1          80          POP BC
3CC3   F1          81          POP AF
3CC4   C9          82          RET
```

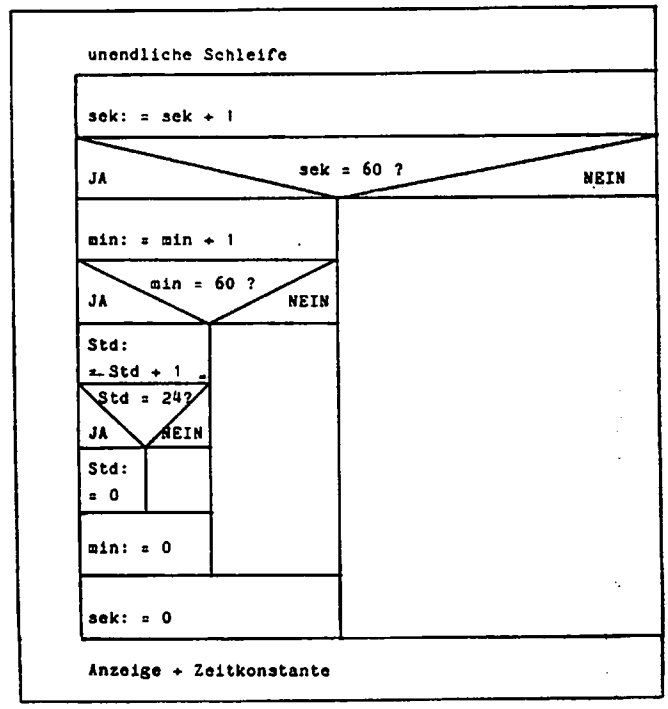
7.4. Uhrenprogramm

Hierbei handelt es sich um ein Programm, das auf dem KIT als 24 Stunden Uhr ablaufen kann. Das Programm wurde ohne Zugriff auf die Monitorroutinen aufgebaut.
Die Ausgabe auf die Anzeige wurde mitprogrammiert.

7.4.1. Programmebeschreibung

Wichtigster Teil des Programmes ist der eigentliche „Uhrenteil" UHR (STMT 9 bis 33), während die übrigen Teile lediglich Unterroutinen darstellen. Dieser Hauptteil wurde zur Einführung des Lesers in dieses Programmierverfahren in strukturierter Programmierung ausgeführt. Siehe hierzu die Applikationsschrift der KONTRON ELEKTRONIK GmbH „Wie spart man Mikrocomputer-Software-Kosten?"
Hervorstechendster Vorteil dieser Programmierweise ist u. a. die hohe Übersichtlichkeit der aufgebauten Programme. Das nachfolgende Ablaufdiagramm verdeutlicht dies.
Um die Uhr zu starten, muß zuerst die aktuelle Zeit eingegeben werden. Dazu wird das B Register mit dem Stundenwert, das C Register mit dem Minutenwert und A mit dem Sekundenwert geladen.
Da beim Uhrenprogramm der Zeittakt durch Abarbeiten eines bestimmten Programmstückes softwaremäßig erzeugt wird, ist Folgendes zu beachten:
Die besagte Programmschleife enthält u.a. die Ausgabe auf die Anzeigeeinheiten. Da bei der Anzeige mit dem WAIT-Zyklus gearbeitet wird, geht der Wert dieser WAIT-Zyklen in die Berechnung der Zeitkonstante mit ein. Durch Bauteiletoleranzen treten jedoch von Gerät zu Gerät geringfügig verschieden lange WAIT-Signale auf. Der im Programm angegebene Wert XX = 150 H (STMT 95) kann deshalb nur als Anhalt dienen. Eine genaue Abstimmung der Uhr ist experimentell, durch Veränderung dieses Wertes vorzunehmen.

7.4.2. Ablaufdiagramm



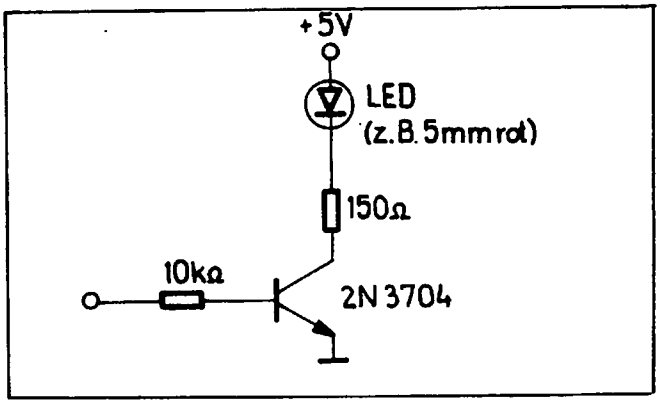
```
UHREN PROGRAMM FUER Z-80 KIT  UHR2          10.7.78_VERS.3     PAGE   1
 LOC   OBJ CODE M STMT SOURCE STATEMENT                          ASM 4.5

                      1  #H UHREN PROGRAMM FUER Z-80 KIT
                      2
                      3
                      4
                      5  ;KONTRON MUENCHEN
                      6
3C50                  7          ORG 3C50H
                      8
                      9  UHR:
3C50   C601          10          ADD A,1          ;SEK := SEK+1
3C52   27            11          DAA              ;HEX --> BCD FORMAT
3C53   FE60          12          CP 60H           ;60 SEK ?
3C55   201A          13          JR NZ,ANZ        ;SEK < 60
3C57   79            14          LD A,C
3C58   C601          15          ADD A,1          ;MIN := MIN+1
3C5A   27            16          DAA              ;HEX --> BCD FORMAT
3C5B   4F            17          LD C,A
3C5C   FE60          18          CP 60H           ;60 MIN ?
3C5E   3E00          19          LD A,0           ;RESET SEK
3C60   200F          20          JR NZ,ANZ        ;MIN < 60
3C62   0E00          21          LD C,0           ;RESET MIN
3C64   78            22          LD A,B
3C65   C601          23          ADD A,1          ;STD := STD+1
3C67   27            24          DAA              ;HEX --> BCD FORMAT
3C68   47            25          LD B,A
3C69   FE24          26          CP 24H           ;24 STD ?
3C6B   3E00          27          LD A,0           ;RESET SEK
3C6D   2002          28          JR NZ,ANZ        ;STD < 24
3C6F   0600          29          LD B,0           ;RESET STD
                     30  ANZ:
3C71   CD763C        31          CALL ANZEIG      ;ANZEIGE UND ZEITKONSTANTE
3C74   18DA          32          JR UHR           ;SCHLEIFE AUSFUEHREN
```

```
ANZEIGE UND ZEITKONSTANTE     UHR2          10.7.78_VERS.3     PAGE   2
 LOC   OBJ CODE M STMT SOURCE STATEMENT                          ASM 4.5

                      33  #H ANZEIGE UND ZEITKONSTANTE
                      34
                      35  ANZEIG:                  ;PROGRAMMTEIL ZUR ANZEIGE DER
                      36                           ;ZEIT UND FUER DIE ZEITKONSTANTE
3C76   C5            37          PUSH BC          ;STD U. MIN RETTEN
3C77   F5            38          PUSH AF          ;SEK RETTEN
3C78   21303C        39          LD HL,LEDBUF     ;LEDBUF ADRESSIEREN
3C7B   CD973C        40          CALL ABSPEI      ;SEK IN LEDBUF
3C7E   79            41          LD A,C
3C7F   CD973C        42          CALL ABSPEI      ;MIN IN LEDBUF
3C82   78            43          LD A,B
3C83   CD973C        44          CALL ABSPEI      ;STD IN LEDBUF
                     45
3C86   015001        46          LD BC,XX         ;ZEITKONSTANTE LADEN
                     47  ZEIT:
3C89   C5            48          PUSH BC          ;ZEITKONSTANTE RETTEN
3C8A   CDA63C        49          CALL DIGIT       ;AUSGABE DER 6 ZIFFERN
3C8D   C1            50          POP BC           ;ZEITKONSTANTE ZURUECK
3C8E   AF            51          XOR A
3C8F   0B            52          DEC BC           ;ZEITKO. := ZEITKO.-1
3C90   B0            53          OR B             ;AUF 0 ABFRAGEN
3C91   B1            54          OR C
3C92   20F5          55          JR NZ,ZEIT
3C94   F1            56          POP AF           ;SEK ZURUECKHOLEN
3C95   C1            57          POP BC           ;STD U. MIN ZURUECKH.
3C96   C9            58          RET
```

```
                        59  #H SUBROUTINES
                        60
                        61
                        62  ABSPEI:
3C97    F5              63          PUSH AF         ;AKKU RETTEN
3C98    E60F            64          AND 0FH         ;DIE 4 UNTEREN BITS
3C9A    77              65          LD (HL),A       ;IN LEDBUF ABSPEICHERN
3C9B    23              66          INC HL          ;LEDBUF := LEDBUF+1
3C9C    F1              67          POP AF          ;AKKU ZURUECKHOLEN
3C9D    0F              68          RRCA            ;DIE OBEREN 4 BITS MIT
3C9E    0F              69          RRCA            ;DEN UNTEREN 4 TAUSCHEN
3C9F    0F              70          RRCA
3CA0    0F              71          RRCA
3CA1    E60F            72          AND 0FH         ;DIE 4 OBEREN BIT
3CA3    77              73          LD (HL),A       ;IN LEDBUF ABSPEICHERN
3CA4    23              74          INC HL          ;LEDBUF := LEDBUF+1
3CA5    C9              75          RET
                        76
                        77  DIGIT:
3CA6    0606            78          LD B,6          ;FUER 6 ZIFFERN
3CA8    DD21303C        79          LD IX,LEDBUF    ;LEDBUF ADRESSIEREN
3CAC    FD21D503        80          LD IY,PORTS     ;PORTS ADRESSIEREN
                        81  AUS:
3CB0    FD4E00          82          LD C,(IY)       ;PORTADRESSE IN C-REG
3CB3    21DB03          83          LD HL,SEGM      ;HL ZEIGT AUF UMRECHNUNGSTAB.
3CB6    DD5E00          84          LD E,(IX)       ;ZIFFER IN E-REG
3CB9    1600            85          LD D,0
3CBB    19              86          ADD HL,DE       ;HEX IN 7-SEGMENT UMRECHNEN
3CBC    7E              87          LD A,(HL)
3CBD    ED79            88          OUT (C),A       ;ZIFFER AUSGEBEN
3CBF    DD23            89          INC IX          ;LEDBUF := LEDBUF+1
3CC1    FD23            90          INC IY          ;PORT := PORT+1
3CC3    10EB            91          DJNZ AUS        ;6 MAL ASUGABE
3CC5    C9              92          RET
                        93
                        94  XX:     EQU 150H        ;WERT FUER ZEITKONSTANTE
                        95  LEDBUF: EQU 3C30H
                        96  PORTS:  EQU 3D5H
                        97  SEGM:   EQU 3DBH
                        98
                        99
                       100
                       101  ;DIE UHR KANN DURCH EINGABE VON WERTEN IN DIE REGISTER
                       102  ;A,B UND C GESTELLT UND DURCH START BEI 3C50 IN
                       103  ;GANG GESETZT WERDEN.
                       104  ;DIE WERTE SIND WIE FOLGT:
                       105  ;A = SEKUNDEN
                       106  ;B = STUNDEN
                       107  ;C = MINUTEN
                       108  ;DA DIE ZEITKONSTANTE DURCH DIE TOLERANZEN DER
                       109  ;BAUTEILE R1,C1 UND IC9 DER GRUNDPLATINE UND
                       110  ;AUSSERDEM DURCH DIE SPEISESPANNUNG BEEINFLUSST
                       111  ;WIRD, IST DER WERT FUER DIE ZEITKONSTANTE VON
                       112  ;KIT ZU KIT UNTERSCHIEDLICH UND MUSS
                       113  ;EXPRIMENTELL ERMITTELT WERDEN.
```

7.5. Lampensteuerung 1

Hierbei handelt es sich um ein Programm, das bestimmte Bitmuster über die Parallele Schnittstelle (PIO) der ECB/K nach außen abgeben kann. Zur Verdeutlichung empfiehlt es sich, die Zustände der Ausgangsleitungen der PIO z.B. mittels folgender Schaltung sichtbar zu machen:



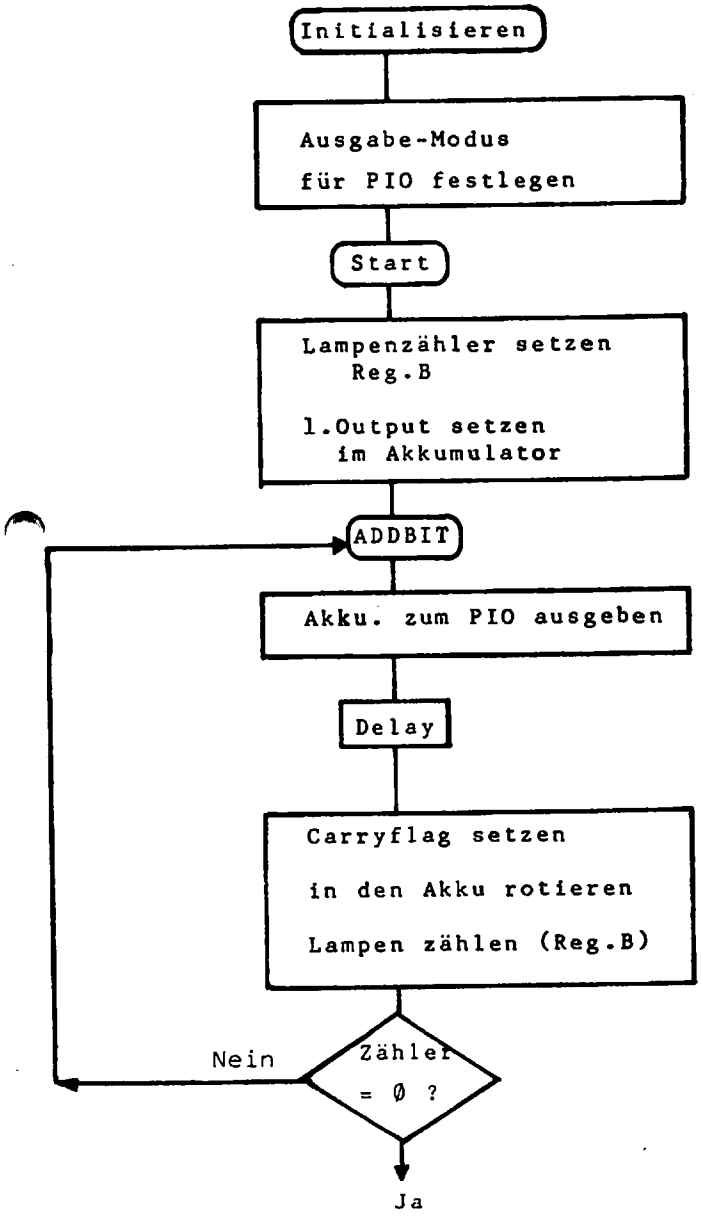
(siehe auch 7.9.)

7.5.1. Programmbeschreibung

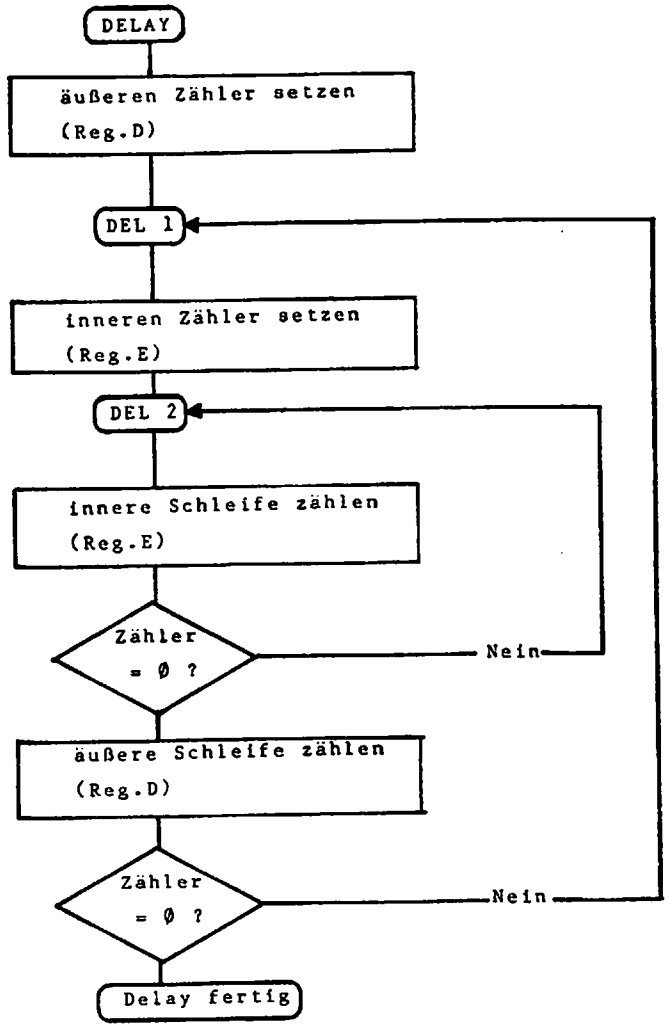
Das Programm läuft derart ab, daß die acht Ausgangsleitungen des PORT A der PIO auf der ECB/K nacheinander von LO auf HI geschaltet werden. Beginnend mit dem niederwertigsten BIT (Do), wird das jeweils nächste BIT geschaltet, bis alle acht Leitungen den Zustand HI angenommen haben. Danach kehrt das Programm zum Anfang zurück, wodurch alle Ausgänge wieder auf LO gesetzt werden. Das Programm läuft in einer unendlichen Schleife und kann nur über RESET unterbrochen werden.

Die aufeinanderfolgenden Bitkombinationen werden im Akkumulator erzeugt und anschließend dem Port übergeben. Der Aufbau der Kombinationen erfolgt über den Befehl RL A (Rotate left through carry), der den Akku Inhalt nach links rotieren läßt und dabei das Carry Flag in die unterste Akkustelle übernimmt. Das Carry Flag wird vor jedem Rotiervorgang auf 1 gesetzt. Um die einzelnen Zustände auch optisch erfassen zu können, muß jeweils eine softwaremäßig erzeugte Zeitkonstante durchlaufen werden.

7.5.2. Flußdiagramm



Flußdiagramm für DELAY



7.5.3. Listing des Lampensteuerprogramms 1

```
                                LAMPB                                PAGE    1
LOC   OBJ CODE M STMT SOURCE STATEMENT                               ASM 5.0

                      1  ;VORFUEHRPROGRAMM FUER Z-80 KIT
                      2  ;ANSTEUERUNG VON LEDS  MODE: A
                      3  ;
                      4  ;KONTRON MUENCHEN AUG.77
                      5  ;
                      6  PBC EQU 0AH     ;I/O PORT A CONTROL INFO-ADR
                      7  PBD EQU 08H     ;I/O PORT A DATEN - ADR
                      8  ;
0000  3E0F            9   LD    A,0FH     ;PORT A IN
0002  D30A           10   OUT   (PBC),A   ;OUTPUT MODE
                     11
                     12  START:
0004  0609           13   LD    B,9       ;LAMPENZAEHLER INITIALISIEREN
0006  3E00           14   LD    A,0       ;AUSZUGEB. BYTE = 00
                     15
                     16  ADDBIT:
0008  D308           17   OUT   (PBD),A   ;NAECHSTES BYTE AUSGEBEN
                     18  ;
000A  16FF           19   LD    D,255     ;AEUSS. ZAEHLSCHLEIFE INITIAL
                     20  DEL1:
000C  1E32           21   LD    E,50      ;INNERE ZAEHLSCHLEIFE INITIAL
                     22  ;
                     23  DEL2:
000E  1D             24   DEC   E         ;INNERE SCHLEIFE RUNTERZ.
000F  20FD           25   JR    NZ,DEL2
                     26  ;
0011  15             27   DEC   D         ;AEUSSERE SCHL. RUNTERZ.
0012  20F8           28   JR    NZ,DEL1
                     29  ;
0014  37             30   SCF             ;CARRY FLAG SETZEN
0015  CB17           31   RL    A         ;ACCU NACH LINKS SCHIEBEN UND
                     32  ;                CARRY BIT AUF BIT 0 VON ACCU
                     33  ;
0017  10EF           34   DJNZ  ADDBIT    ;LAMPENZAEHLER RUNTERZAEHLEN
                     35  ;
0019  C30400    R    36   JP    START
                     37  ;
                     38  END
```

7.6. Lampensteuerung 2

Bei diesem Programm handelt es sich ebenfalls um eine Ansteuerung der acht PORT-Leitungen.

7.6.1. Programmbeschreibung

Entgegen dem ersten Beispiel existiert in diesem Fall die Zeitschleife als Unterprogramm und wird über CALL aufgerufen. Der Neubeginn eines Durchlaufs ist nicht mehr von einer bestimmten Anzahl von Durchläufen abhängig, sondern erfolgt durch einen unbedingten Sprung. Die auszugebende Kombination wird wieder im Akkumulator aufgebaut. Diesmal jedoch durch einen direkten Ladevorgang, d.h. im Gegensatz zum vorigen Programm sind willkürliche Kombinationsfolgen möglich.

7.6.2. Listing des Lampensteuerprogramms 2

```
                                LAMPE                                PAGE   1
LOC   OBJ CODE M STMT SOURCE STATEMENT                               ASM 4.5

                      1  ;VORFUEHRPROGRAMM FUER Z-80 KIT
                      2  ;ANSTEUERUNG VON LEDS  MODE: 0
                      3  ;
                      4  ;KONTRON MUENCHEN AUG.77
                      5  ;
                      6  PBC EQU 0AH     ;I/O PORT A CONTROL INFO-ADR
                      7  PBD EQU 08H     ;I/O PORT A DATEN - ADR
                      8  ;
3C50                  9          ORG 3C50H
                     10
3C50  3E0F           11   LD     A,0FH     ;PORT A IN
3C52  D30A           12   OUT    (PBC),A   ;OUTPUT MODE
                     13  ;
                     14  BITMTR:
3C54  3E18           15   LD A,18H
3C56  CD743C         16   CALL OUTBIT
3C59  3E24           17   LD A,24H
3C5B  CD743C         18   CALL OUTBIT
3C5E  3E42           19   LD A,42H
3C60  CD743C         20   CALL OUTBIT
3C63  3E81           21   LD A,81H
3C65  CD743C         22   CALL OUTBIT
3C68  3E42           23   LD A,42H
3C6A  CD743C         24   CALL OUTBIT
3C6D  3E24           25   LD A,24H
3C6F  CD743C         26   CALL OUTBIT
3C72  18E0           27   JR BITMTR
                     28
                     29  OUTBIT:
3C74  D308           30   OUT    (PBD),A   ;NAECHSTES BYTE AUSGEBEN
                     31  ;
3C76  16FF           32   LD     D,255     ;AEUSS. ZAEHLSCHLEIFE INITIAL
                     33  DELI:
3C78  1E32           34   LD     E,50      ;INNERE ZAEHLSCHLEIFE INITIAL
                     35  ;
                     36  DEL2:
3C7A  1D             37   DEC    E         ;INNERE SCHLEIFE RUNTERZ.
3C7B  20FD           38   JR    NZ,DEL2
                     39  ;
3C7D  15             40   DEC   D         ;AEUSSERE SCHL. RUNTERZ.
3C7E  20F8           41   JR    NZ,DEL1
3C80  C9             42   RET
```

7.7. Lampensteuerung 3

Auch hier handelt es sich um eine Ansteuerung der acht PORT Leitungen.

7.7.1. Programmbeschreibung

Um die Möglichkeiten des Z80 Befehlssatzes zu demonstrieren sei an dieser Stelle das vorige Programm mit Hilfe von BIT SET Befehlen aufgebaut. Dabei ergibt sich der gleiche Ablaufplan; lediglich der Aufbau der Kombinationen im Akku ist unterschiedlich. Außerdem arbeitet der PIO PORT nicht im MODE 0 (OUTPUT) sondern im MODE 3 (BIT MODE).

7.7.2. Listing

```
                                LAMPE1                                  PAGE    1
LOC   OBJ CODE M STMT SOURCE STATEMENT                                   ASM 4.5

                      1  ;VORFUEHRPROGRAMM FUER Z-80 KIT
                      2  ;ANSTEUERUNG VON LEDS  MODE: 0
                      3  ;
                      4  ;KONTRON MUENCHEN AUG.77
                      5  ;
                      6  PBC EQU 0AH     ;I/O PORT A CONTROL INFO-ADR
                      7  PBD EQU 0BH     ;I/O PORT A DATEN - ADR
                      8  ;
3C50                  9          ORG 3C50H
                     10
3C50  3E0F           11   LD     A,0FH     ;PORT A IN
3C52  D30A           12   OUT    (PBC),A   ;OUTPUT MODE
                     13  ;
                     14  BITMTR:
3C54  97             15          SUB A              ;AKKU LOESCHEN
3C55  CBC7           16          SET 0,A            ;BIT 0 UND
3C57  CBFF           17          SET 7,A            ;BIT 7 DES AKKUS SETZEN
3C59  CD953C         18   CALL OUTBIT
3C5C  CB87           19          RES 0,A            ;BIT 0 UND
3C5E  CBBF           20          RES 7,A            ;BIT 7 RUECKSETZEN
3C60  CBCF           21          SET 1,A            ;DAFUER BIT 1 UND
3C62  CBF7           22          SET 6,A            ;BIT 6 DES AKKUS SETZEN
3C64  CD953C         23   CALL OUTBIT
3C67  CB8F           24          RES 1,A
3C69  CBB7           25          RES 6,A
3C6B  CBD7           26          SET 2,A
3C6D  CBEF           27          SET 5,A
3C6F  CD953C         28   CALL OUTBIT
3C72  CB97           29          RES 2,A
3C74  CBAF           30          RES 5,A
3C76  CBDF           31          SET 3,A
3C78  CBE7           32          SET 4,A
3C7A  CD953C         33   CALL OUTBIT
3C7D  CB9F           34          RES 3,A
3C7F  CBA7           35          RES 4,A
3C81  CBD7           36          SET 2,A
3C83  CBEF           37          SET 5,A
3C85  CD953C         38   CALL OUTBIT
3C88  CB97           39          RES 2,A
3C8A  CBAF           40          RES 5,A
3C8C  CBCF           41          SET 1,A
3C8E  CBF7           42          SET 6,A
3C90  CD953C         43   CALL OUTBIT
3C93  18BF           44   JR BITMTR
                     45
                     46  OUTBIT:
3C95  D308           47   OUT    (PBD),A   ;NAECHSTES BYTE AUSGEBEN
                     48  ;
3C97  16FF           49   LD     D,255     ;AEUSS. ZAEHLSCHLEIFE INITIAL
                     50  DEL1:
3C99  1E32           51   LD     E,50      ;INNERE ZAEHLSCHLEIFE INITIAL
                     52  ;
                     53  DEL2:
3C9B  1D             54   DEC    E         ;INNERE SCHLEIFE RUNTERZ.
3C9C  20FD           55   JR     NZ,DEL2
                     56  ;
3C9E  15             57   DEC    D         ;AEUSSERE SCHL. RUNTERZ.
3C9F  20F8           58   JR     NZ,DEL1
3CA1  C9             59   RET
```

7.8. Lampensteuerung mit Interrupt

Hier handelt es sich ebenfalls um eine Ansteuerung der parallelen Schnittstelle. Während ein PORT wieder als Ausgabe dient, arbeitet der zweite PORT im Eingabebetrieb. Auf eine Eingabe reagiert das Steuerprogramm mit einem Wechsel der Durchlaufgeschwindigkeit.

7.8.1. Programmbeschreibung

PORT A wird wieder in den OUTPUT-MODE versetzt. PORT B wird auf MODE 3 (BIT MODE) programmiert, wobei D_0 bis D_3 als INPUT Leitungen, D_4 bis D_7 als OUTPUTs definiert werden. Mit Hilfe des Interrupt-Steuerwortes und der nachfolgenden Maske wird Leitung D_0 als einzig interruptfähig festgelegt.
Wird die an Leitung D_0 angeschlossene **entprellte** Taste betätigt, wird das laufende Programm unterbrochen (INTERRUPT) und in der Interruptserviceroutine ein Auswechseln des Zeitkonstantenwertes vorgenommen. Das Programm läuft anschließend mit der neuen Geschwindigkeit weiter.

7.8.2. Listing

```
INTERRUPTPROGRAMM                INT                              PAGE   1
 LOC    OBJ CODE M STMT SOURCE STATEMENT                           ASM 4.5

                    1  *H INTERRUPTPROGRAMM
                    2
                    3
3C50                4          ORG 3C50H
                    5
                    6
                    7  INIT:
3C50    31F03C      8          LD SP,SPOINT      ;STACKPOINTER SETZEN
3C53    21863C      9          LD HL,LISTE       ;ADR.DER LISTE DER PIO ANWEIS.
3C56    0E0B       10          LD C,PORTB        ;CONTROLADR.PORT B INS C REG.
3C58    0604       11          LD B,4            ;4 DURCHGAENGE FUER OTIR
3C5A    EDB3       12          OTIR              ;BLOCKAUSGABE AN PORT B
                   13
3C5C    3E0F       14          LD A,0FH          ;OUTPUT MODE FUER PORT A
3C5E    D30A       15          OUT (PORTA+2),A   ;AN PORT A CONTROLADR.
                   16
3C60    218A3C     17          LD HL,TABLE       ;ADR.INTERRUPTTABELLE INS HL
3C63    ED69       18          OUT (C),L         ;LOW BYTE = INT.VEKTOR FUER PIO
3C65    7C         19          LD A,H            ;HIGH BYTE = INT.VEKTOR CPU
3C66    ED47       20          LD I,A
3C68    21FFFF     21          LD HL,0FFFFH      ;ZEITKONSTANTE 1 (LANGSAM)
3C6B    11000F     22          LD DE,0F00H       ;ZEITKNSTANTE 2 (SCHNELL)
3C6E    ED5E       23          IM 2              ;INT.MODE 2!
3C70    FB         24          EI
                   25
                   26
                   27
                   28
3C71    3E01       29          LD A,1            ;ERSTES AUSZUGEBENDES MUSTER
                   30
                   31
                   32  SCHL1:
3C73    D308       33          OUT (PORTA),A     ;DATEN AN PORT A DATAADR.
3C75    07         34          RLCA              ;BIT LINKS ROTIEREN
3C76    CD7B3C     35          CALL ZEIT         ;ZEITKONSTANTE AUFRUFEN
3C79    18F8       36          JR SCHL1          ;NAECHSTER DURCHGANG
                   37
                   38
                   39
                   40
                   41  ZEIT:
3C7B    F5         42          PUSH AF           ;AF RETTEN
3C7C    E5         43          PUSH HL           ;AKTUELLEN WERT VON HL
3C7D    C1         44          POP BC            ;INS BC BRINGEN
                   45
                   46
                   47  SCHL2:
3C7E    0B         48          DEC BC            ;BC RUNTERZAEHLEN
3C7F    AF         49          XOR A             ;AKKU LOESCHEN
3C80    B0         50          OR B              ;MIT B UND
3C81    B1         51          OR C              ;C "VERODERN"
3C82    20FA       52          JR NZ,SCHL2       ;WENN NICHT 0,ZURUECK!
3C84    F1         53          POP AF            ;SONST AF HOLEN
3C85    C9         54          RET
                   55
                   56
                   57
                   58
```

```
INTERRUPTPROGRAMM                INT                          PAGE   2
 LOC   OBJ CODE M STMT SOURCE STATEMENT                        ASM 4.5

                     59  LISTE:                  ;ANWEISUNGEN FUER PORT B
3C86   CF            60          DEFB OCFH       ;MODE 3 = BIT MODE
3C87   OF            61          DEFB  OFH       ;MASKE,BIT 0,1,2 U 3 = INPUT
3C88   D7            62          DEFB OD7H       ;INT.STEUERWORT MIT EI PIO
3C89   FE            63          DEFB OFEH       ;INT.MASKE,NUR BIT O INT.FAEHIG!
                     64
                     65
3C8A                 66          ORG ($+1)/2*2   ;INT.TABELLE MUSS AUF OERADER
                     67                          ;ADRESSE BEGINNEN!!!
                     68
                     69  TABLE:
3C8A   8C3C          70          DEFW ISR        ;ISR = INT.SERVICE ROUTINE
                     71
                     72
                     73  ISR:
3C8C   EB            74          EX HL,DE        ;AUSTAUSCH DER ZEITKONSTANTEN
3C8D   FB            75          EI
3C8E   ED4D          76          RETI
                     77
                     78
                     79
                     80
                     81  PORTA EQU 08H           ;PORT A DATA ADR.
                     82  PORTB EQU 0BH           ;PORT B CONTROL ADR.
                     83  SPOINT EQU 3CF0H        ;STACKPOINTER
```

```
TTY TREIBER ROUTINEN FUER EC  USART          180778          PAGE   2
 LOC   OBJ CODE M STMT SOURCE STATEMENT                        ASM 4.5

                     59
                     60  USDAT:  EQU 04H         ;04H = ADRESSE FUER ECB/K
                     61                          ;UND ECB/C
```

In Bezug auf Interruptbehandlung sei auf die, von der KONTRON ELEKTRONIK GmbH herausgegebene Applikationsschrift

„Eine detaillierte Interrupt-Behandlung – hier in Z 80 Systemen“
verwiesen.

7.9. TTY Treiberprogramm

Hier handelt es sich um ein Programm das für die auf dem KIT bereits hardwaremäßig vorhandene TTY Schnittstelle geeignet ist.

7.9.1. Programmbeschreibung

Das Programm umfaßt drei Teile: die Initialisierung des USART, eine Schreib- und eine Lese-Routine.
Die Initialisierung ist so ausgelegt, daß der USART aus jedem beliebigen Zustand heraus angesprochen werden kann. Die eingestellten Parameter können dem Listing entnommen werden. Für eine andere Einstellung ist die Applicationsschrift

Using the 8251 Universal
Synchronous/Asyncronous
Receiver/Transmitter

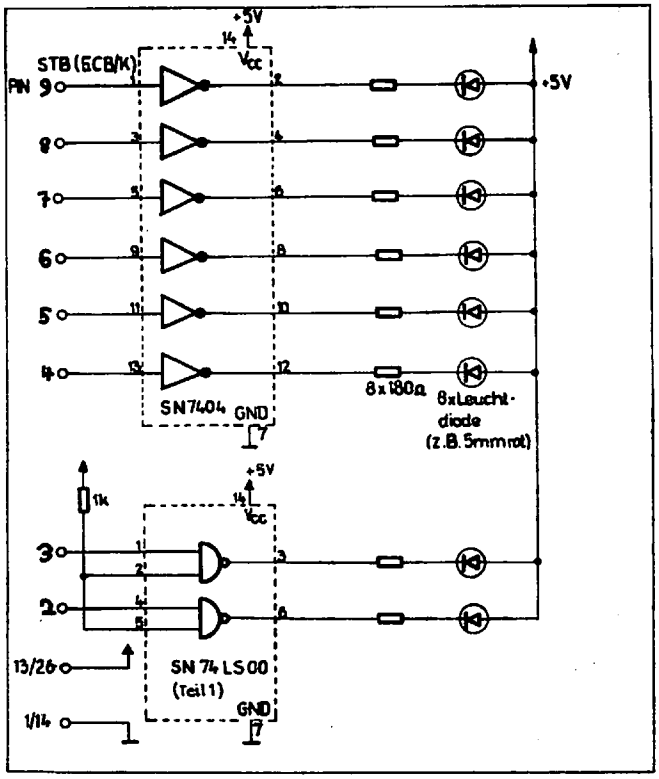
heranzuziehen.
Die Lese- bzw. Schreib-Routine bezieht sich jeweils auf den Akkumulator. In beiden Fällen wird nach dem Aufruf eine Warteschleife durchlaufen, bis ein Datenwort angekommen bzw. der Transmitter wieder zur Aufnahme eines neuen Datenwortes bereit ist.
Alle drei Teile können über CALL aus beliebigen Programmteilen aufgerufen werden.

7.9.2. Listing

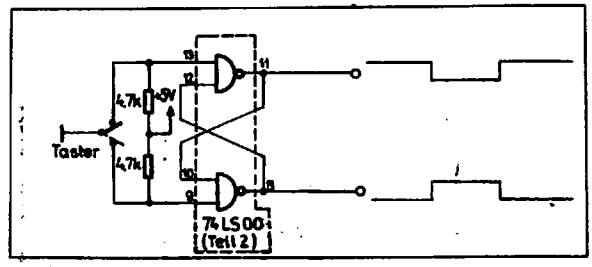
```
TTY TREIBER ROUTINEN FUER EC  USART          180778              PAGE   1
 LOC   OBJ CODE M STMT SOURCE STATEMENT                            ASM 4.5

                      1  *H TTY TREIBER ROUTINEN FUER ECB/K
                      2
OF6B                  3          ORG OF6BH
                      4
                      5  SETUP:                  ;ROUTINE ZUR USART-
                      6                          ;INITIALISIERUNO
                      7
                      8
OF6B  97              9          SUB A           ;AKKU LOESCHEN
OF6C  D306           10          OUT (USDAT+2),A ;DREIMALIGE ANWEISUNG,
OF6E  D306           11          OUT (USDAT+2),A ;UM EINEN DEFINIERTEN
OF70  D306           12          OUT (USDAT+2),A ;ZUSTAND ZU ERREICHEN
                     13
OF72  3E40           14          LD A,40H        ;RESETANWEISUNG
OF74  D306           15          OUT (USDAT+2),A
                     16
OF76  00             17          NOP
OF77  00             18          NOP             ;8 T-ZYKLEN ZEIT
                     19                          ;FUER RESET
                     20
OF78  3ECE           21          LD A,0CEH       ;STEUERWORT:
                     22                          ;  ASYN MODE (X16)
                     23                          ;  8 DATA BITS
                     24                          ;  NO PARITY
                     25                          ;  2 STOP BITS
OF7A  D306           26          OUT (USDAT+2),A
                     27
OF7C  3E27           28          LD A,27H        ;STEUERWORT:
                     29                          ;  RTS = '1'
                     30                          ;  ENABLE RXRDY
                     31                          ;  ENABLE TXRDY
                     32                          ;  DTR = '0'
OF7E  D306           33          OUT (USDAT+2),A
OF80  C9             34          RET
                     35
                     36
                     37
                     38  TTYIN:                  ;EIN ASCII CHARACTER IN DEN AKKU
OF81  DB06           39          IN A,(USDAT+2)
OF83  CB4F           40          BIT 1,A         ;RECEIVER READY ?
OF85  28FA           41          JR Z,TTYIN      ;WENN NICHT
OF87  DB04           42          IN A,(USDAT)    ;SONST DATA IN DEN AKKU
OF89  CBBF           43          RES 7,A         ;RESET PARITY
OF8B  C9             44          RET
                     45
                     46
                     47
                     48  TTYOUT:                 ;EIN ASCII CHARACTER AUS DEM AK
OF8C  F5             49          PUSH AF         ;AF RETTEN
                     50  READY:
OF8D  DB06           51          IN A,(USDAT+2)
OF8F  CB47           52          BIT 0,A         ;TRANSMITTER READY ?
OF91  28FA           53          JR Z,READY      ;WENN NICHT
OF93  F1             54          POP AF          ;SONST AF ZURUECK UND
OF94  D304           55          OUT (USDAT),A   ;DATA SENDEN
OF96  C9             56          RET
                     57
                     58
```

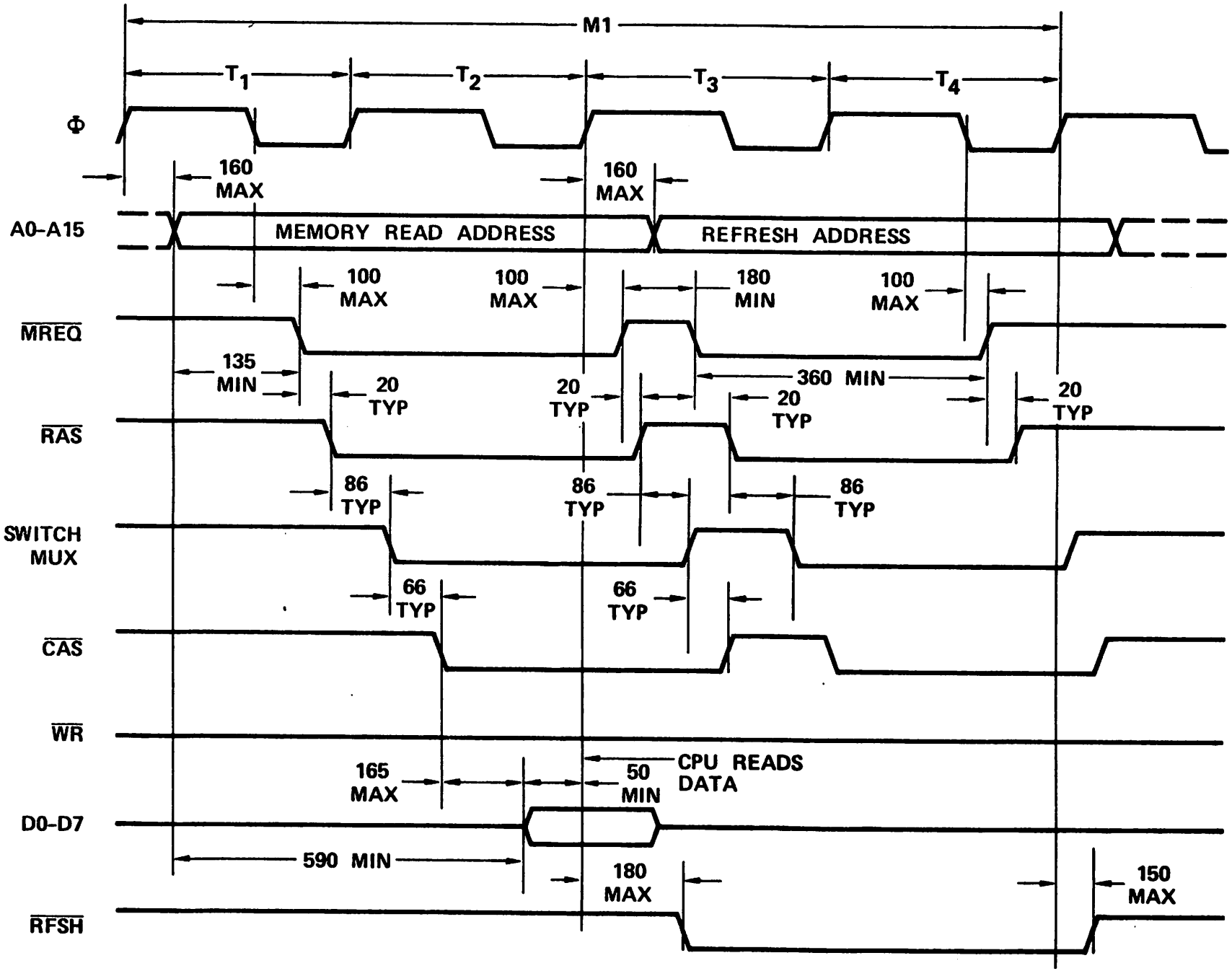
7.10. Schaltungsentwurf für eine achtstellige LED Anzeige und einer entprellten Taste

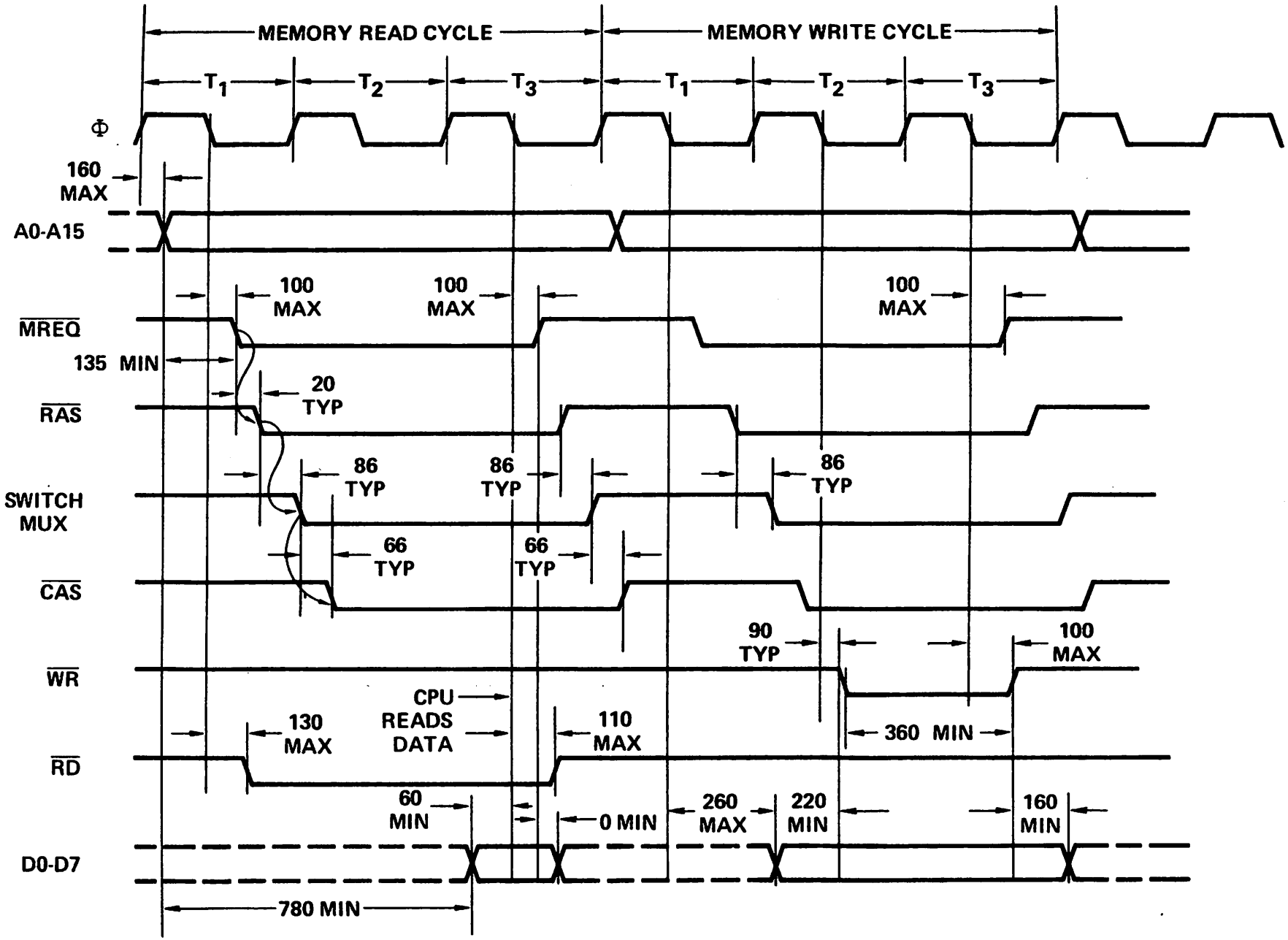
Zum Anzeigen der Ausgangszustände der acht PORT Leitungen von PORT A der PIO auf der ECB K kann folgende Schaltung verwendet werden:



Die beiden restlichen Gatter des SN 7400 können zum Aufbau einer Tastenentprellung verwendet werden:

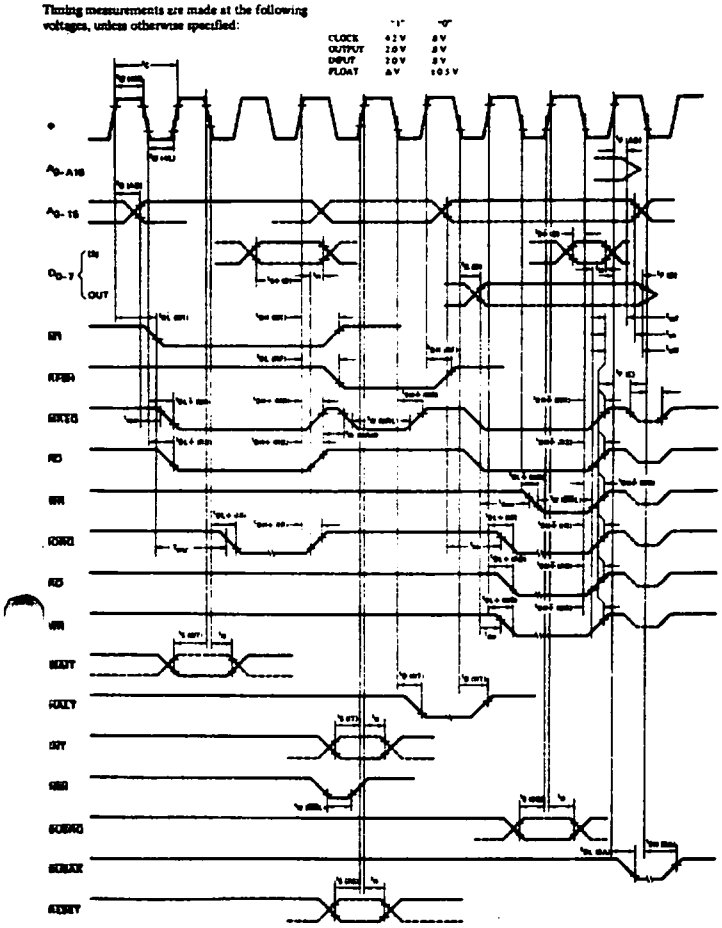



M1
T1
T2
T3
T4
Φ
160 MAX
160 MAX
A0-A15
MEMORY READ ADDRESS
REFRESH ADDRESS
100 MAX
100 MAX
180 MIN
100 MAX
MREQ
135 MIN
20 TYP
20 TYP
360 MIN
20 TYP
20 TYP
RAS
86 TYP
86 TYP
86 TYP
SWITCH MUX
66 TYP
66 TYP
CAS
WR
165 MAX
50 MIN
CPU READS DATA
D0-D7
590 MIN
180 MAX
150 MAX
RFSH

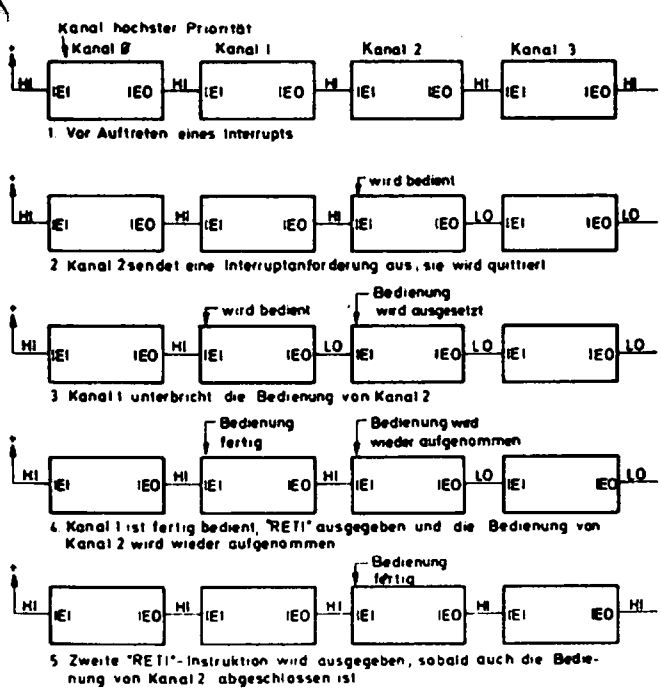

MEMORY READ CYCLE
MEMORY WRITE CYCLE
T1
T2
T3
T1
T2
T3
Φ
160 MAX
A0-A15
100 MAX
100 MAX
100 MAX
MREQ
135 MIN
20 TYP
RAS
86 TYP
86 TYP
86 TYP
SWITCH MUX
66 TYP
66 TYP
CAS
90 TYP
100 MAX
WR
CPU READS DATA
130 MAX
110 MAX
360 MIN
RD
60 MIN
0 MIN
260 MAX
220 MIN
160 MIN
D0-D7
780 MIN

Zu Anhang 1:

☐ Zeitverhalten des Z 80-CPU



Anhang 2: Interrupt-Architektur des Systems Z80



1. Vor Auftreten eines Interrupts
2. Kanal 2 sendet eine Interruptanforderung aus, sie wird quittiert
3. Kanal 1 unterbricht die Bedienung von Kanal 2
4. Kanal 1 ist fertig bedient, "RETI" ausgegeben und die Bedienung von Kanal 2 wird wieder aufgenommen
5. Zweite "RETI"-Instruktion wird ausgegeben, sobald auch die Bedienung von Kanal 2 abgeschlossen ist

Anhang 3: Z80-Befehlssatz

☐ Z 80-Befehlssatz

Im folgenden wird lediglich ein großer Überblick über den Z 80-Befehlssatz gegeben, Einzelheiten finden Sie im Z-80-Manual. Zu unterscheiden sind folgende Befehlsgruppen:

- 8 Bit-Ladebefehle (= "8 bit loads")
- 16 Bit-Ladebefehle (= "16 bit loads")
- Austauschbefehle (= "Exchanges")
- Blocktransfers im Speicher ("Memory Block Moves")
- Blocksuchbefehle ("Memory Block Searches")
- 8 Bit arthmetische und logische Befehle (= "8 bit arithmetic and logic")
- 16 Bit arithmetische Befehle (= 16 bit arithmetic)
- Allgemeine Akkumulator- und Status-Anweisungen (= "General purpose Accu and Flag Operations")
- Akku-Rotieren und -Schieben (= "Rotate and Shift")
- Bit Setzen, Rücksetzen und Testen (= "Bit Set, Reset and Test")
- Ein/Ausgabe (= "Input and Output")
- Sprünge (= "Jumps")
- Unterprogrammaufrufe ("Calls")
- Restarts ("Restarts")
- Rücksprünge (= "Returns")
- Sonstige Befehle ("Miscellaneous Group")

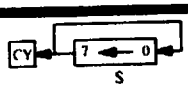
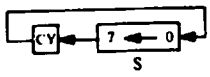
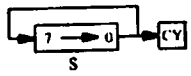
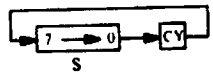
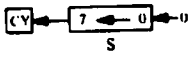
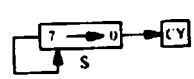
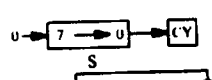
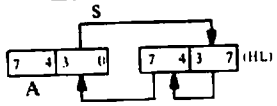
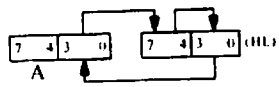
Zeichenerklärung

b	bezeichnet eine Bit-Position in einem Register oder einer Speicherstelle
cc	Statusbedingungscode ("Flag condition") Erlaubte Bedingungen: NZ: ungleich Null (= "Nonzero") Z: gleich Null (= "Zero") NC: Kein Übertrag (= "Non carry") C: Übertrag (= "Carry") PO: Ungerade oder kein Überlauf ("Parity Odd") PE: Gerade oder Überlauf ("Parity Even") P: Positiv N: Negativ
d	Zielregister (8 bit)
dd	16 bit-Zielregister oder Zieladresse im Speicher
e	8 bit vorzeichenbehaftetes Zweierkomplement der Distanz bei relativen Sprüngen oder indizierter Adressierung
L	bezeichnet die 8 speziellen Zieladressen in Seite Ø (dezimal Ø, 8, 16, 32, 4Ø, 48 und 56).
n	8 bit-Binärzahl.
nn	16 bit-Binärzahl.
r	allgemeines 8 bit-Register (A, B, C, D, E, H oder L)
s	8 bit Senderregister oder Speicherstelle.
s_b	ein Bit in einem bestimmten 8 bit-Register oder Speicherstelle.
ss	16 bit Senderegister oder Speicherstelle.
Index „L"	Niederwertige (= "Low order") 8 bits eines 16-bit-Registers
Index „H"	Höherwertige (= "High order") 8 bits eines 16-bit-Registers
( )	„Inhalt von . . . ." Zeichen zwischen den Klammern stellen einen Zeiger auf eine Speicherstelle oder ein I/O Port dar.

8 bit Register sind: A, B, C, D, E, H, L, I und R.
16 bit „Paare": AF, BC, DE und HL.
16 bit Register: SP, PC, IX und IY.

Als Adressierweisen kommen (auch Kombinationen) in Frage:
Direkt ("immediate")
Erweitert direkt ("immediate extended")
Modifizierte Seite Null ("Modified Page Zero")
Relativ ("relative")
Erweitert ("extended")
Indiziert ("indexed")
über Register:
impliziert ("implied")
indirekt über Register ("register indirect")
Adressierung eines Bits.

	Mnemo	Durchgeführte Operation	Bemerkungen
8 bit-Ladebefehle	LD r, s	r ← s	s ≡ r, n, (HL), (IX+e), (IY+e)
	LD d, r	d ← r	d ≡ (HL), r (IX+e), (IY+e)
	LD d, n	d ← n	d ≡ (HL), (IX+e), (IY+e)
	LD A, s	A ← s	s ≡ (BC), (DE), (nn), I, R
	LD d, A	d ← A	d ≡ (BC), (DE), (nn), I, R
16 bit-Ladebefehle	LD dd, nn	dd ← nn	dd ≡ BC, DE, HL, SP, IX, IY
	LD dd, (nn)	dd ← (nn)	dd ≡ BC, DE, HL, SP, IX, IY
	LD (nn), ss	(nn) ← ss	ss ≡ BC, DE, HL, SP, IX, IY
	LD SP, ss	SP ← ss	ss = HL, IX, IY
	PUSH ss	$(SP-1) \leftarrow ss_H$; $(SP-2) \leftarrow ss_L$	ss = BC, DE, HL, AF, IX, IY
	POP dd	$dd_L \leftarrow (SP)$; $dd_H \leftarrow (SP+1)$	dd = BC, DE, HL, AF, IX, IY
Registeraustausch	EX DE, HL	DE ↔ HL	
	EX AF, AF'	AF ↔ AF'	
	EXX	$\begin{pmatrix} BC \\ DE \\ HL \end{pmatrix} \leftrightarrow \begin{pmatrix} BC' \\ DE' \\ HL' \end{pmatrix}$	
	EX (SP), ss	$(SP) \leftrightarrow ss_L$, $(SP+1) \leftrightarrow ss_H$	ss ≡ HL, IX, IY
Block-Suchbefehle	LDI	(DE) ← (HL), DE ← DE+1 HL ← HL+1, BC ← BC-1	
	LDIR	(DE) ← (HL), DE ← DE+1 HL ← HL+1, BC ← BC-1 Repeat until BC = 0	
	LDD	(DE) ← (HL), DE ← DE-1 HL ← HL-1, BC ← BC-1	
	LDDR	(DE) ← (HL), DE ← DE-1 HL ← HL-1, BC ← BC-1 Repeat until BC = 0	
	CPI	A-(HL), HL ← HL+1 BC ← BC-1	
	CPIR	A-(HL), HL ← HL+1 BC ← BC-1, Repeat until BC = 0 or A = (HL)	A-(HL) sets the flags only. A is not affected
	CPD	A-(HL), HL ← HL-1 BC ← BC-1	
	CPDR	A-(HL), HL ← HL-1 BC ← BC-1, Repeat until BC= 0 or A = (HL)	
8 bit-arithmetische und logische Operationen	ADD s	A ← A + s	CY is the carry flag s ≡ r, n, (HL) (IX+e), (IY+e)
	ADC s	A ← A + s + CY	
	SUB s	A ← A - s	
	SBC s	A ← A - s - CY	
	AND s	A ← A ∧ s	
	OR s	A ← A ∨ s	
	XOR s	A ← A ⊕ s	

Mnemo	Durchgeführte Operation	Bemerkungen
16 bit-arithmetische Operationen		
CP s	A - s	s = r, n (HL) (IX+e), (IY+e)
INC d	d ← d + 1	d = r, (HL) (IX+e), (IY+e)
DEC d	d ← d - 1	
ADD HL, ss	HL ← HL + ss	ss ≡ BC, DE HL, SP
ADC HL, ss	HL ← HL + ss + CY	
SBC HL, ss	HL ← HL - ss - CY	
ADD IX, ss	IX ← IX + ss	ss ≡ BC, DE, IX, SP
ADD IY, ss	IY ← IY + ss	ss ≡ BC, DE, IY, SP
INC dd	dd ← dd + 1	dd ≡ BC, DE, HL, SP, IX, IY
DEC dd	dd ← dd - 1	dd ≡ BC, DE, HL, SP, IX, IY
BCD-, Akku- und Flag-Operationen		
DAA	Converts A contents into packed BCD following add or subtract.	Operands must be in packed BCD format
CPL	$A \leftarrow \overline{A}$	
NEG	A ← 00 – A	
CCF	$CY \leftarrow \overline{CY}$	
SCF	CY ← 1	
Verschiedene		
NOP	No operation	
HALT	Halt CPU	
DI	Disable Interrupts	
EI	Enable Interrupts	
IM 0	Set interrupt mode 0	8080A mode
IM 1	Set interrupt mode 1	Call to 0038_H
IM 2	Set interrupt mode 2	Indirect Call
RLC s		s ≡ r, (HL) (IX+e), (IY+e)
RL s		
RRC s		
RR s		
SLA s		
SRA s		
SRL s		
RLD		
RRD		

Mnemo	Durchgeführte Operation	Bemerkungen
Bit Setzen, Rücksetzen, Testen		
BIT b, s	$Z \leftarrow \overline{s_b}$	Z is zero flag
SET b, s	$s_b \leftarrow 1$	s ≡ r, (HL) (IX+e), (IY+e)
RES b, s	$s_b \leftarrow 0$	
Ein/Ausgabe		
IN A, (n)	A ← (n)	Set flags
IN r, (C)	r ← (C)	
INI	(HL) ← (C), HL ← HL + 1 B ← B - 1	
INIR	(HL) ← (C), HL ← HL + 1 B ← B - 1 Repeat until B = 0	
IND	(HL) ← (C), HL ← HL - 1 B ← B - 1	
INDR	(HL) ← (C), HL ← HL - 1 B ← B - 1 Repeat until B = 0	
OUT(n), A	(n) ← A	
OUT(C), r	(C) ← r	
OUTI	(C) ← (HL), HL ← HL + 1 B ← B - 1	
OTIR	(C) ← (HL), HL ← HL + 1 B ← B - 1 Repeat until B = 0	
OUTD	(C) ← (HL), HL ← HL - 1 B ← B - 1	
OTDR	(C) ← (HL), HL ← HL - 1 B ← B - 1 Repeat until B = 0	
Sprungbefehle		
JP nn	PC ← nn	cc { NZ PO; Z PE; NC P; C M
JP cc, nn	If condition cc is true PC ← nn, else continue	
JR e	PC ← PC + e	kk { NZ NC; Z C
JR kk, e	If condition kk is true PC ← PC + e, else continue	
JP (ss)	PC ← ss	ss = HL, IX, IY
DJNZ e	B ← B - 1, if B = 0 continue, else PC ← PC + e	
Unterprogramm-aufruf		
CALL nn	$(SP-1) \leftarrow PC_H$ $(SP-2) \leftarrow PC_L$, PC ← nn	cc { NZ PO; Z PE; NC P; C M
CALL cc, nn	If condition cc is false continue, else same as CALL nn	
Restarts		
RST L	$(SP-1) \leftarrow PC_H$ $(SP-2) \leftarrow PC_L$, $PC_H \leftarrow 0$ $PC_L \leftarrow L$	
Rücksprünge		
RET	$PC_L \leftarrow (SP)$, $PC_H \leftarrow (SP+1)$	cc { NZ PO; Z PE; NC P; C M
RET cc	If condition cc is false continue, else same as RET	
RETI	Return from interrupt, same as RET	
RETN	Return from non-maskable interrupt	

Z80-CPU INSTRUCTION SET

ALPHABETICAL ASSEMBLY MNEMONIC	OPERATION
ADC HL,ss	Add with Carry Reg. pair ss to HL
ADC A,s	Add with carry operand s to Acc.
ADD A,n	Add value n to Acc.
ADD A,r	Add Reg. r to Acc.
ADD A,(HL)	Add location (HL) to Acc.
ADD A,(IX+d)	Add location (IX+d) to Acc.
ADD A,(IY+d)	Add location (IY+d) to Acc.
ADD HL,ss	Add Reg. pair ss to HL
ADD IX,pp	Add Reg. pair pp to IX
ADD IY,rr	Add Reg. pair rr to IY
AND s	Logical 'AND' of operand s and Acc
BIT b,(HL)	Test BIT b of location (HL)
BIT b,(IX+d)	Test BIT b of location (IX+d)
BIT b,(IY+d)	Test BIT b of location (IY+d)
BIT b,r	Test BIT b of Reg. r
CALL cc,nn	Call subroutine at location nn if condition cc is true
CALL nn	Unconditional call subroutine at location nn
CCF	Complement carry flag
CP s	Compare operand s with Acc.
CPD	Compare location (HL) and Acc. decrement HL and BC
CPDR	Compare location (HL) and Acc. decrement HL and BC, repeat until BC=0
CPI	Compare location (HL) and Acc. increment HL and decrement BC
CPIR	Compare location (HL) and Acc. increment HL, decrement BC repeat until BC=0
CPL	Complement Acc. (1's comp)
DAA	Decimal adjust Acc.
DEC m	Decrement operand m
DEC IX	Decrement IX
DEC IY	Decrement IY
DEC ss	Decrement Reg. pair ss
DI	Disable interrupts
DJNZ e	Decrement B and Jump relative if B≠0
EI	Enable interrupts
EX (SP),HL	Exchange the location (SP) and HL
EX (SP),IX	Exchange the location (SP) and IX
EX (SP),IY	Exchange the location (SP) and IY
EX AF,AF'	Exchange the contents of AF and AF'
EX DE,HL	Exchange the contents of DE and HL
EXX	Exchange the contents of BC,DE,HL with contents of BC',DE',HL' respectively
HALT	HALT (wait for interrupt or reset)
IM 0	Set interrupt mode 0
IM 1	Set interrupt mode 1
IM 2	Set interrupt mode 2
IN A,(n)	Load the Acc. with input from device n
IN r,(C)	Load the Reg. r with input from device (C)
INC (HL)	Increment location (HL)
INC IX	Increment IX
INC (IX+d)	Increment location (IX+d)
INC IY	Increment IY
INC (IY+d)	Increment location (IY+d)
INC r	Increment Reg. r
INC ss	Increment Reg. pair ss
IND	Load location (HL) with input from port (C), decrement HL and B
INDR	Load location (HL) with input from port (C), decrement HL and decrement B, repeat until B=0
INI	Load location (HL) with input from port (C); and increment HL and decrement B
INIR	Load location (HL) with input from port (C), increment HL and decrement B, repeat until B=0
JP (HL)	Unconditional Jump to (HL)
JP (IX)	Unconditional Jump to (IX)
JP (IY)	Unconditional Jump to (IY)
JP cc,nn	Jump to location nn if condition cc is true
JP nn	Unconditional jump to location nn
JR C,e	Jump relative to PC+e if carry=1
JR e	Unconditional Jump relative to PC+e
JR NC,e	Jump relative to PC+e if carry=0
JR NZ,e	Jump relative to PC+e if non zero (Z=0)
JR Z,e	Jump relative to PC+e if zero (Z=1)
LD A,(BC)	Load Acc. with location (BC)
LD A,(DE)	Load Acc. with location (DE)
LD A,I	Load Acc. with I
LD A,(nn)	Load Acc. with location nn
LD A,R	Load Acc. with Reg. R
LD (BC),A	Load location (BC) with Acc.
LD (DE),A	Load location (DE) with Acc.
LD (HL),n	Load location (HL) with value n
LD dd,nn	Load Reg. pair dd with value nn
LD dd,(nn)	Load Reg. pair dd with location (nn)
LD HL,(nn)	Load HL with location (nn)
LD (HL),r	Load location (HL) with Reg. r
LD I,A	Load I with Acc.
LD IX,nn	Load IX with value nn
LD IX,(nn)	Load IX with location (nn)
LD (IX+d),n	Load location (IX+d) with value n
LD (IX+d),r	Load location (IX+d) with Reg. r
LD IY,nn	Load IY with value nn
LD IY,(nn)	Load IY with location (nn)
LD (IY+d),n	Load location (IY+d) with value n
LD (IY+d),r	Load location (IY+d) with Reg. r
LD (nn),A	Load location (nn) with Acc.
LD (nn),dd	Load location (nn) with Reg. pair dd
LD (nn),HL	Load location (nn) with HL
LD (nn),IX	Load location (nn) with IX
LD (nn),IY	Load location (nn) with IY
LD R,A	Load R with Acc.
LD r,(HL)	Load Reg. r with location (HL)
LD r,(IX+d)	Load Reg. r with location (IX+d)
LD r,(IY+d)	Load Reg. r with location (IY+d)
LD r,n	Load Reg. r with value n
LD r,r'	Load Reg. r with Reg. r'
LD SP,HL	Load SP with HL
LD SP,IX	Load SP with IX
LD SP,IY	Load SP with IY
LDD	Load location (DE) with location (HL) decrement DE,HL and BC
LDDR	Load location (DE) with location (HL), decrement DE,HL and BC; repeat until BC=0
LDI	Load location (DE) with location (HL), increment DE,HL, decrement BC
LDIR	Load location (DE) with location (HL), increment DE,HL, decrement BC and repeat until BC=0
NEG	Negate Acc. (2's complement)
NOP	No operation
OR s	Logical 'OR' of operand s and Acc.
OTDR	Load output port (C) with location (HL) decrement HL and B, repeat until B=0
OTIR	Load output port (C) with location (HL), increment HL, decrement B, repeat until B=0
OUT (C),r	Load output port (C) with Reg. r
OUT (n),A	Load output port (n) with Acc.
OUTD	Load output port (C) with location (HL), decrement HL and B
OUTI	Load output port (C) with location (HL), increment HL and decrement B
POP IX	Load IX with top of stack
POP IY	Load IY with top of stack
POP qq	Load Reg. pair qq with top of stack
PUSH IX	Load IX onto stack
PUSH IY	Load IY onto stack
PUSH qq	Load Reg. pair qq onto stack
RES b,m	Reset Bit b of operand m
RET	Return from subroutine
RET cc	Return from subroutine if condition cc is true
RETI	Return from interrupt
RETN	Return from non maskable interrupt
RL m	Rotate left through carry operand m
RLA	Rotate left Acc. through carry
RLC (HL)	Rotate location (HL) left circular
RLC (IX+d)	Rotate location (IX+d) left circclar
RLC (IY+d)	Rotate location (IY+d) left circular
RLC r	Rotate Reg. r left circular
RLCA	Rotate left circular Acc.
RLD	Rotate digit left and right between Acc. and location (HL)
RR m	Rotate right through carry operand m
RRA	Rotate right Acc. through carry
RRC m	Rotate operand m right circular
RRCA	Rotate right circular Acc.
RRD	Rotate digit right and left between Acc. and location (HL)
RST p	Restart to location p
SBC A,s	Subtract operand s from Acc. with carry
SBC HL,ss	Subtract Reg. pair ss from HL with carry
SCF	Set carry flag (C=1)
SET b,(HL)	Set Bit b of location (HL)
SET b,(IX+d)	Set Bit b of location (IX+d)
SET b,(IY+d)	Set Bit b of location (IY+d)
SET b,r	Set Bit b of Reg. r
SLA m	Shift operand m left arithmetic
SRA m	Shift operand m right arithmetic
SRL m	Shift operand m right logical
SUB s	Subtract operand s from Acc.
XOR s	Exclusive 'OR' operand s and Acc.

Anhang 4: Befehlsvergleichsliste der Systeme Z80 und 8080A

Opcode	8080A		Z80		Opcode	8080A		Z80	
00	NOP		NOP		30	———		JR	NC,disp
01	LXI	B,dddd	LD	BC,dddd	31	LXI	SP,dddd	LD	SP,dddd
02	STAX	B	LD	(BC),A	32	STA	adr	LD	(adr),A
03	INX	B	INC	BC	33	INX	SP	INC	SP
04	INR	B	INC	B	34	INR	M	INC	(HL)
05	DCR	B	DEC	B	35	DCR	M	DEC	(HL)
06	MVI	B,dd	LD	B,dd	36	MVI	M,dd	LD	(HL),dd
07	RLC		RLCA		37	STC		SCF	
08			EX	AF,AF'	38	———		JR	C,disp
09	DAD	B	ADD	HL,BC	39	DAD	SP	ADD	HL,SP
0A	LDAX	B	LD	A,(BC)	3A	LDA	adr	LD	A,(adr)
0B	DCX	B	DEC	BC	3B	DCX	SP	DEC	SP
0C	INR	C	INC	C	3C	INR	A	INC	A
0D	DCR	C	DEC	C	3D	DCR	A	DEC	A
0E	MVI	C,dd	LD	C,dd	3E	MVI	A,dd	LD	A,dd
0F	RRC		RRCA		3F	CMC		CCF	
10	———		DJNZ	disp	40	MOV	B,B	LD	B,B
11	LXI	D,dddd	LD	DE,dddd	41	MOV	B,C	LD	B,C
12	STAX	D	LD	(DE),A	42	MOV	B,D	LD	B,D
13	INX	D	INC	DE	43	MOV	B,E	LD	B,E
14	INR	D	INC	D	44	MOV	B,H	LD	B,H
15	DCR	D	DEC	D	45	MOV	B,L	LD	B,L
16	MVI	D,dd	LD	D,dd	46	MOV	B,M	LD	B,(HL)
17	RAL		RLA		47	MOV	B,A	LD	B,A
18	———		JR	disp	48	MOV	C,B	LD	C,B
19	DAD	D	ADD	HL,DE	49	MOV	C,C	LD	C,C
1A	LDAX	D	LD	A,(DE)	4A	MOV	C,D	LD	C,D
1B	DCX	D	DEC	DE	4B	MOV	C,E	LD	C,E
1C	INR	E	INC	E	4C	MOV	C,H	LD	C,H
1D	DCR	E	DEC	E	4D	MOV	C,L	LD	C,L
1E	MVI	E,dd	LD	E,dd	4E	MOV	C,M	LD	C,(HL)
1F	RAR		RRA		4F	MOV	C,A	LD	C,A
20	———		JR	NZ,disp	50	MOV	D,B	LD	D,B
21	LXI	H,dddd	LD	HL,dddd	51	MOV	D,C	LD	D,C
22	SHLD	adr	LD	(adr),HL	52	MOV	D,D	LD	D,D
23	INX	H	INC	HL	53	MOV	D,E	LD	D,E
24	INR	H	INC	H	54	MOV	D,H	LD	D,H
25	DCR	H	DEC	H	55	MOV	D,L	LD	D,L
26	MVI	H,dd	LD	H,dd	56	MOV	D,M	LD	D,(HL)
27	DAA		DAA		57	MOV	D,A	LD	D,A
28	———		JR	Z,disp	58	MOV	E,B	LD	E,B
29	DAD	H	ADD	HL,HL	59	MOV	E,C	LD	E,C
2A	LHLD	adr	LD	HL,(adr)	5A	MOV	E,D	LD	E,D
2B	DCX	H	DEC	HL	5B	MOV	E,E	LD	E,E
2C	INR	L	INC	L	5C	MOV	E,H	LD	E,H
2D	DCR	L	DEC	L	5D	MOV	E,L	LD	E,L
2E	MVI	L,dd	LD	L,dd	5E	MOV	E,M	LD	E,(HL)
2F	CMA		CPL		5F	MOV	E,A	LD	E,A

Anhang 4:

Opcode	8080A		Z80		Opcode	8080A		Z80	
60	MOV	H,B	LD	H,B	90	SUB	B	SUB	B
61	MOV	H,C	LD	H,C	91	SUB	C	SUB	C
62	MOV	H,D	LD	H,D	92	SUB	D	SUB	D
63	MOV	H,E	LD	H,E	93	SUB	E	SUB	E
64	MOV	H,H	LD	H,H	94	SUB	H	SUB	H
65	MOV	H,L	LD	H,L	95	SUB	L	SUB	L
66	MOV	H,M	LD	H,(HL)	96	SUB	M	SUB	(HL)
67	MOV	H,A	LD	H,A	97	SUB	A	SUB	A
68	MOV	L,B	LD	L,B	98	SBB	B	SBC	A,B
69	MOV	L,C	LD	L,C	99	SBB	C	SBC	A,C
6A	MOV	L,D	LD	L,D	9A	SBB	D	SBC	A,D
6B	MOV	L,E	LD	L,E	9B	SBB	E	SBC	A,E
6C	MOV	L,H	LD	L,H	9C	SBB	H	SBC	A,H
6D	MOV	L,L	LD	L,L	9D	SBB	L	SBC	A,L
6E	MOV	L,M	LD	L,(HL)	9E	SBB	M	SBC	A,(HL)
6F	MOV	L,A	LD	L,A	9F	SBB	A	SBC	A,A
70	MOV	M,B	LD	(HL),B	A0	ANA	B	AND	B
71	MOV	M,C	LD	(HL),C	A1	ANA	C	AND	C
72	MOV	M,D	LD	(HL),D	A2	ANA	D	AND	D
73	MOV	M,E	LD	(HL),E	A3	ANA	E	AND	E
74	MOV	M,H	LD	(HL),H	A4	ANA	H	AND	H
75	MOV	M,L	LD	(HL),L	A5	ANA	L	AND	L
76	HLT		HALT		A6	ANA	M	AND	(HL)
77	MOV	M,A	LD	(HL),A	A7	ANA	A	AND	A
78	MOV	A,B	LD	A,B	A8	XRA	B	XOR	B
79	MOV	A,C	LD	A,C	A9	XRA	C	XOR	C
7A	MOV	A,D	LD	A,D	AA	XRA	D	XOR	D
7B	MOV	A,E	LD	A,E	AB	XRA	E	XOR	E
7C	MOV	A,H	LD	A,H	AC	XRA	H	XOR	H
7D	MOV	A,L	LD	A,L	AD	XRA	L	XOR	L
7E	MOV	A,M	LD	A,(HL)	AE	XRA	M	XOR	(HL)
7F	MOV	A,A	LD	A,A	AF	XRA	A	XOR	A
80	ADD	B	ADD	A,B	B0	ORA	B	OR	B
81	ADD	C	ADD	A,C	B1	ORA	C	OR	C
82	ADD	D	ADD	A,D	B2	ORA	D	OR	D
83	ADD	E	ADD	A,E	B3	ORA	E	OR	E
84	ADD	H	ADD	A,H	B4	ORA	H	OR	H
85	ADD	L	ADD	A,L	B5	ORA	L	OR	L
86	ADD	M	ADD	A,(HL)	B6	ORA	M	OR	(HL)
87	ADD	A	ADD	A,A	B7	ORA	A	OR	A
88	ADC	B	ADC	A,B	B8	CMP	B	CP	B
89	ADC	C	ADC	A,C	B9	CMP	C	CP	C
8A	ADC	D	ADC	A,D	BA	CMP	D	CP	D
8B	ADC	E	ADC	A,E	BB	CMP	E	CP	E
8C	ADC	H	ADC	A,H	BC	CMP	H	CP	H
8D	ADC	L	ADC	A,L	BD	CMP	L	CP	L
8E	ADC	M	ADC	A,(HL)	BE	CMP	M	CP	(HL)
8F	ADC	A	ADC	A,A	BF	CMP	A	CP	A

Anhang 4:

Opcode	8080A		Z80	
C0	RNZ		RET	NZ
C1	POP	B	POP	BC
C2	JNZ	adr	JP	NZ,adr
C3	JMP	adr	JP	adr
C4	CNZ	adr	CALL	NZ,adr
C5	PUSH	B	PUSH	BC
C6	ADI	dd	ADD	A,dd
C7	RST	0	RST	0
C8	RZ		RET	Z
C9	RET		RET	
CA	JZ	adr	JP	Z,adr
CB	———		see below	
CC	CZ	adr	CALL	Z,adr
CD	CALL	adr	CALL	adr
CE	ACI	dd	ADC	A,dd
CF	RST	1	RST	8
D0	RNC		RET	NC
D1	POP	D	POP	DE
D2	JNC	adr	JP	NC,adr
D3	OUT	port	OUT	port,A
D4	CNC	adr	CALL	NC,adr
D5	PUSH	D	PUSH	DE
D6	SUI	dd	SUB	dd
D7	RST	2	RST	10H
D8	RC		RET	C
D9	———		EXX	
DA	JC	adr	JP	C,adr
DB	IN	port	IN	A,port
DC	CC	adr	CALL	C,adr
DD	———		see below	
DE	SBI	dd	SBC	A,dd
DF	RST	3	RST	18H
E0	RPO		RET	PO
E1	POP	H	POP	HL
E2	JPO	adr	JP	PO,adr
E3	XTHL		EX	(SP),HL
E4	CPO	adr	CALL	PO,adr
E5	PUSH	H	PUSH	HL
E6	ANI	dd	AND	dd
E7	RST	4	RST	20H
E8	RPE		RET	PE
E9	PCHL		JP	(HL)
EA	JPE	adr	JP	PE,adr
EB	XCHG		EX	DE,HL
EC	CPE	adr	CALL	PE,adr
ED	———		see below	
EE	XRI	dd	XOR	dd
EF	RST	5	RST	28H
F0	RP		RET	P
F1	POP	PSW	POP	AF
F2	JP	adr	JP	P,adr
F3	DI		DI	
F4	CP	adr	CALL	P,adr
F5	PUSH	PSW	PUSH	AF
F6	ORI	dd	OR	dd
F7	RST	6	RST	30H
F8	RM		RET	M
F9	SPHL		LD	SP,HL
FA	JM	adr	JP	M,adr
FB	EI		EI	
FC	CM	adr	CALL	M,adr
FD	———		see below	
FE	CPI	dd	CP	dd
FF	RST	7	RST	38H

Anhang 5: Folgende Anweisungen sind nur im Befehlssatz des Systems Z80 implementiert

Opcode

CB00	RLC	B
CB01	RLC	C
CB02	RLC	D
CB03	RLC	E
CB04	RLC	H
CB05	RLC	L
CB06	RLC	(HL)
CB07	RLC	A
CB08	RRC	B
CB09	RRC	C
CB0A	RRC	D
CB0B	RRC	E
CB0C	RRC	H
CB0D	RRC	L
CB0E	RRC	(HL)
CB0F	RRC	A
CB10	RL	B
CB11	RL	C
CB12	RL	D
CB13	RL	E
CB14	RL	H
CB15	RL	L
CB16	RL	(HL)
CB17	RL	A
CB18	RR	B
CB19	RR	C
CB1A	RR	D
CB1B	RR	E
CB1C	RR	H
CB1D	RR	L
CB1E	RR	(HL)
CB1F	RR	A
CB20	SLA	B
CB21	SLA	C
CB22	SLA	D
CB23	SLA	E
CB24	SLA	H
CB25	SLA	L
CO26	SLA	(HL)
CB27	SLA	A
CB28	SRA	B
CB29	SRA	C
CB2A	SRA	D
CB2B	SRA	E
CB2C	SRA	H
CB2D	SRA	L
CB2E	SRA	(HL)
CB2F	SRA	A
CB38	SRL	B
CB39	SRL	C
CB3A	SRL	D
CB3B	SRL	E
CB3C	SRL	H
CB3D	SRL	L
CB3E	SRL	(HL)
CB3F	SRL	A
CB40	BIT	0,B
CB41	BIT	0,C
CB42	BIT	0,D
CB43	BIT	0,E
CB44	BIT	0,H
CB45	BIT	0,L
CB46	BIT	0,(HL)
CB47	BIT	0,A
CB48	BIT	1,B
CB49	BIT	1,C
CB4A	BIT	1,D
CB4B	BIT	1,E
CB4C	BIT	1,H
CB4D	BIT	1,L
CB4E	BIT	1,(HL)
CB4F	BIT	1,A
CB50	BIT	2,B
CB51	BIT	2,C
CB52	BIT	2,D
CO53	BIT	2,E
CO54	OIT	2,H
CO55	BIT	2,L
CO56	OIT	2,(HL)
CB57	BIT	2,A
CB58	BIT	3,B
CB59	BIT	3,C
CB5A	BIT	3,D
CB5B	BIT	3,E
CB5C	BIT	3,H
CB5D	BIT	3,L
CB5E	BIT	3,(HL)
CB5F	BIT	3,A

Anhang 5:

Opcode		
CBC0	SET	0,B
CBC1	SET	0,C
CBC2	SET	0,D
CBC3	SET	0,E
CBC4	SET	0,H
CBC5	SET	0,L
CBC6	SET	0,(HL)
CBC7	SET	0,A
CBC8	SET	1,B
CBC9	SET	1,C
CBCA	SET	1,D
CBCB	SET	1,E
CBCC	SET	1,H
CBCD	SET	1,L
CBCE	SET	1,(HL)
CBCF	SET	1,A
CBD0	SET	2,B
CBD1	SET	2,C
CBD2	SET	2,D
CBD3	SET	2,E
CBD4	SET	2,H
CBD5	SET	2,L
CBD6	SET	2,(HL)
CBD7	SET	2,A
CBD8	SET	3,B
CBD9	SET	3,C
CBDA	SET	3,D
CBDB	SET	3,E
CBDC	SET	3,H
CBDD	SET	3,L
CBDE	SET	3,(HL)
CBDF	SET	3,A
CBE0	SET	4,B
CBE1	SET	4,C
CBE2	SET	4,D
CBE3	SET	4,E
CBE4	SET	4,H
CBE5	SET	4,L
CBE6	SET	4,(HL)
CBE7	SET	4,A
CBE8	SET	5,B
CBE9	SET	5,C
CBEA	SET	5,D
CBEB	SET	5,E
CBEC	SET	5,H
CBED	SET	5,L
CBEE	SET	5,(HL)
CBEF	SET	5,A

Opcode		
CBF0	SET	6,B
CBF1	SET	6,C
CBF2	SET	6,D
CBF3	SET	6,E
CBF4	SET	6,H
CBF5	SET	6,L
CBF6	SET	6,(HL)
CBF7	SET	6,A
CBF8	SET	7,B
CBF9	SET	7,C
CBFA	SET	7,D
CBFB	SET	7,E
CBFC	SET	7,H
CBFD	SET	7,L
CBFE	SET	7,(HL)
CBFF	SET	7,A
DD09	ADD	IX,BC
DD19	ADD	IX,DE
DD21	LD	IX,dddd
DD22	LD	(adr),IX
DD23	INC	IX
DD29	ADD	IX,IX
DD2A	LD	IX,(adr)
DD2B	DEC	IX
DD34	INC	(IX+offset)
DD35	DEC	(IX+offset)
DD36	LD	(IX+offset),dd
DD39	ADD	IX,SP
DD46	LD	B,(IX+offset)
DD4E	LD	C,(IX+offset)
DD56	LD	D,(IX+offset)
DD5E	LD	E,(IX+offset)
DD66	LD	H,(IX+offset)
DD6E	LD	L,(IX+offset)
DD70	LD	(IX+offset),B
DD71	LD	(IX+offset),C
DD72	LD	(IX+offset),D
DD73	LD	(IX+offset),E
DD74	LD	(IX+offset),H
DD75	LD	(IX+offset),L
DD77	LD	(IX+offset),A
DD7E	LD	A,(IX+offset)
DD86	ADD	A,(IX+offset)
DD8E	ADC	A,(IX+offset)

Anhang 5:

Opcode		
CB60	BIT	4,B
CB61	BIT	4,C
CB62	BIT	4,D
CB63	BIT	4,E
CB64	BIT	4,H
CB65	BIT	4,L
CB66	BIT	4,(HL)
CB67	BIT	4,A
CB68	BIT	5,B
CB69	BIT	5,C
CB6A	BIT	5,D
CB6B	BIT	5,E
CB6C	BIT	5,H
CB6D	BIT	5,L
CB6E	BIT	5,(HL)
CB6F	BIT	5,A
CB70	BIT	6,B
CB71	BIT	6,C
CB72	BIT	6,D
CB73	BIT	6,E
CB74	BIT	6,H
CB75	BIT	6,L
CB76	BIT	6,(HL)
CB77	BIT	6,A
CB78	BIT	7,B
CB79	BIT	7,C
CB7A	BIT	7,D
CB7B	BIT	7,E
CB7C	BIT	7,H
CB7D	BIT	7,L
CB7E	BIT	7,(HL)
CB7F	BIT	7,A
CB80	RES	0,B
CB81	RES	0,C
CB82	RES	0,D
CB83	RES	0,E
CB84	RES	0,H
CB85	RES	0,L
CB86	RES	0,(HL)
CB87	RES	0,A
CB88	RES	1,B
CB89	RES	1,C
CB8A	RES	1,D
CB8B	RES	1,E
CB8C	RES	1,H
CB8D	RES	1,L
CB8E	RES	1,(HL)
CB8F	RES	1,A

Opcode		
CB90	RES	2,B
CB91	RES	2,C
CB92	RES	2,D
CB93	RES	2,E
CB94	RES	2,H
CB95	RES	2,L
CB96	RES	2,(HL)
CB97	RES	2,A
CB98	RES	3,B
CB99	RES	3,C
CB9A	RES	3,D
CB9B	RES	3,E
CB9C	RES	3,H
CB9D	RES	3,L
CB9E	RES	3,(HL)
CB9F	RES	3,A
CBA0	RES	4,B
CBA1	RES	4,C
CBA2	RES	4,D
CBA3	RES	4,E
CBA4	RES	4,H
CBA5	RES	4,L
CBA6	RES	4,(HL)
CBA7	RES	4,A
CBA8	RES	5,B
CBA9	RES	5,C
CBAA	RES	5,D
CBAB	RES	5,E
CBAC	RES	5,H
CBAD	RES	5,L
CBAE	RES	5,(HL)
CBAF	RES	5,A
CBB0	RES	6,B
CBB1	RES	6,C
CBB2	RES	6,D
CBB3	RES	6,E
CBB4	RES	6,H
CBB5	RES	6,L
CBB6	RES	6,(HL)
CBB7	RES	6,A
CBB8	RES	7,B
CBB9	RES	7,C
CBBA	RES	7,D
CBBB	RES	7,E
CBBC	RES	7,H
CBBD	RES	7,L
CBBE	RES	7,(HL)
CBBF	RES	7,A

Anhang 5:

Opcodo		
ED88	LDIR	
ED81	CPIR	
ED82	INIR	
ED83	OTIR	
ED88	LDDR	
ED89	CPDR	
ED8A	INDR	
ED8B	OTDR	
FD09	ADD	IY,BC
FD19	ADD	IY,DE
FD21	LD	IY,dddd
FD22	LD	(adr),IY
FD23	INC	IY
FD29	ADD	IY,IY
FD2A	LD	IY,(adr)
FD2B	DEC	IY
FD34	INC	(IY+offset)
FD35	DEC	(IY+offset)
FD36	LD	(IY+offset),dd
FD39	ADD	IY,SP
FD46	LD	B,(IY+offset)
FD4E	LD	C,(IY+offset)
FD56	LD	D,(IY+offset)
FD5E	LD	E,(IY+offset)
FD66	LD	H,(IY+offset)
FD6E	LD	L,(IY+offset)
FD70	LD	(IY+offset),B
FD71	LD	(IY+offset),C
FD72	LD	(IY+offset),D
FD73	LD	(IY+offset),E
FD74	LD	(IY+offset),H
FD75	LD	(IY+offset),L
FD77	LD	(IY+offset),A
FD7E	LD	A,(IY+offset)
FD86	ADD	A,(IY+offset)
FD8E	ADC	A,(IY+offset)
FD96	SUB	(IY+offset)
FD9E	SBC	A,(IY+offset)
FDA6	AND	(IY+offset)
FDAE	XOR	(IY+offset)
FDB6	OR	(IY+offset)
FDBE	CP	(IY+offset)

Opcodo		
FDCBof06	RLC	(IY+offset)
FDCBofDE	RRC	(IY+offset)
FDCBof16	RL	(IY+offset)
FDCBof1E	RR	(IY+offset)
FDCBof26	SLA	(IY+offset)
FDCBof2E	SRA	(IY+offset)
FDCBof3E	SRL	(IY+offset)
FDCBof46	BIT	0,(IY+offset)
FDCBof4E	BIT	1,(IY+offset)
FDCBof56	BIT	2,(IY+offset)
FDCBof5E	BIT	3,(IY+offset)
FDCBof66	BIT	4,(IY+offset)
FDCBof6E	BIT	5,(IY+offset)
FDCBof76	BIT	6,(IY+offset)
FDCBof7E	BIT	7,(IY+offset)
FDCBof86	RES	0,(IY+offset)
FDCBef8E	RES	1,(IY+offset)
FDCBof96	RES	2,(IY+offset)
FDCBof9E	RES	3,(IY+offset)
FDCBofA6	RES	4,(IY+offset)
FDCBofAE	RES	5,(IY+offset)
FDCBofB6	RES	6,(IY+offset)
FDCBofBE	RES	7,(IY+offset)
FDCBofC6	SET	0,(IY+offset)
FDCBofCE	SET	1,(IY+offset)
FDCBofD6	SET	2,(IY+offset)
FDCBofDE	SET	3,(IY+offset)
FDCBofE6	SET	4,(IY+offset)
FDCBofEE	SET	5,(IY+offset)
FDCBofF6	SET	6,(IY+offset)
FDCBofFE	SET	7,(IY+offset)
FDE1	POP	IY
FDE3	EX	(SP),IY
FDE5	PUSH	IY
FDE9	JP	(IY)
FDF9	LD	SP,IY

Anhang 5:

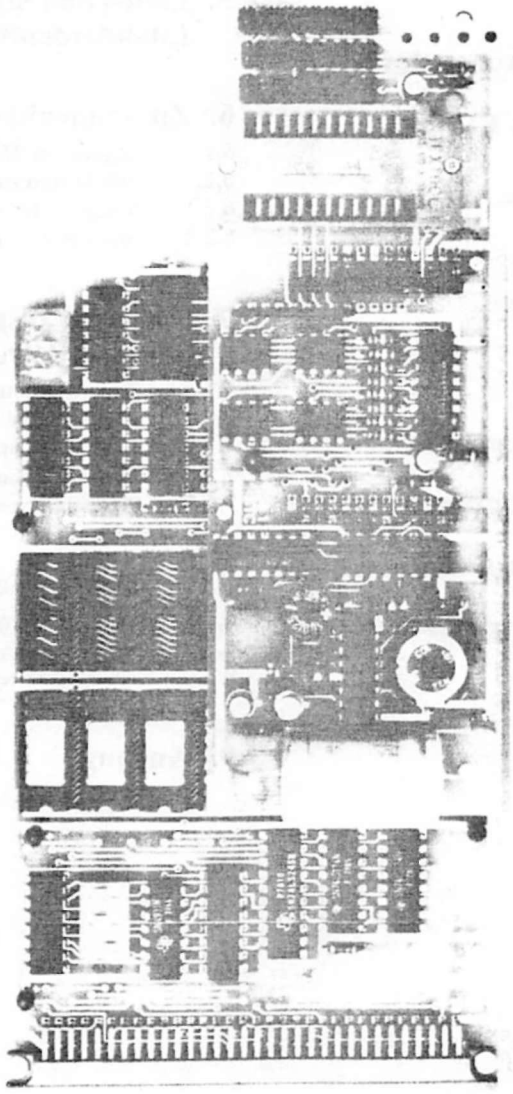
Opcode		
DD96	SUB	(IX+offset)
DD9E	SBC	A,(IX+offset)
DDA6	AND	(IX+offset)
DDAE	XOR	(IX+offset)
DDB6	OR	(IX+offset)
DDBE	CP	(IX+offset)
DDCBof06	RLC	(IX+offset)
DDCBofDE	RRC	(IX+offset)
DDCBof16	RL	(IX+offset)
DDCBof1E	RR	(IX+offset)
DDCBof26	SLA	(IX+offset)
DDCBof2E	SRA	(IX+offset)
DDCBof3E	SRL	(IX+offset)
DDCBof46	BIT	0,(IX+offset)
DDCBof4E	BIT	1,(IX+offset)
DDCBof56	BIT	2,(IX+offset)
DDCBof5E	BIT	3,(IX+offset)
DDCBof66	BIT	4,(IX+offset)
DDCBof6E	BIT	5,(IX+offset)
DDCBof76	BIT	6,(IX+offset)
DDCBof7E	BIT	7,(IX+offset)
DDCBof86	RES	0,(IX+offset)
DDCBof8E	RES	1,(IX+offset)
DDCBof96	RES	2,(IX+offset)
DDCBof9E	RES	3,(IX+offset)
DDCBofA6	RES	4,(IX+offset)
DDCBofAE	RES	5,(IX+offset)
DDCBofB6	RES	6,(IX+offset)
DDCBofBE	RES	7,(IX+offset)
DDCBofC6	SET	0,(IX+offset)
DDCBofCE	SET	1,(IX+offset)
DDCBofD6	SET	2,(IX+offset)
DDCBofDE	SET	3,(IX+offset)
DDCBofE6	SET	4,(IX+offset)
DDCBofEE	SET	5,(IX+offset)
DDCBofF6	SET	6,(IX+offset)
DDCBofFE	SET	7,(IX+offset)

Opcode		
DDE1	POP	IX
DDE3	EX	(SP),IX
DDE5	PUSH	IX
DDE9	JP	(IX)
DDF9	LD	SP,IX
ED40	IN	B,(C)
ED41	OUT	(C),B
ED42	SBC	HL,BC
ED43	LD	(dddd),BC
ED44	NEG	
ED45	RETN	
ED46	IM	0
ED47	LD	I,A
ED48	IN	C,(C)
ED49	OUT	(C),C
ED4A	ADC	HL,BC
ED4B	LD	BC,(adr)
ED4D	RETI	
ED50	IN	D,(C)
ED51	OUT	(C),D
ED52	SBC	HL,DE
ED53	LD	(adr),DE
ED56	IM	1
ED57	LD	A,I
ED58	IN	E,(C)
ED59	OUT	(C),E
ED5A	ADC	HL,DE
ED5B	LD	DE,(adr)
ED5E	IM	2
ED60	IN	H,(C)
ED61	OUT	(C),H
ED62	SBC	HL,HL
ED67	RRD	
ED68	IN	L,(C)
ED69	OUT	(C),L
ED6A	ADC	HL,HL
ED6F	RLD	
ED72	SBC	HL,SP
ED73	LD	(adr),SP
ED78	IN	A,(C)
ED79	OUT	(C),A
ED7A	ADC	HL,SP
ED7B	LD	SP,(adr)
EDA0	LDI	
EDA1	CPI	
EDA2	INI	
EDA3	OUTI	
EDA8	LDD	
EDA9	CPD	
EDAA	IND	
EDAB	OUTD	

KONTRON KIT®

PROM-Programmier- und Speichererweiterung KONTRON-KIT/P

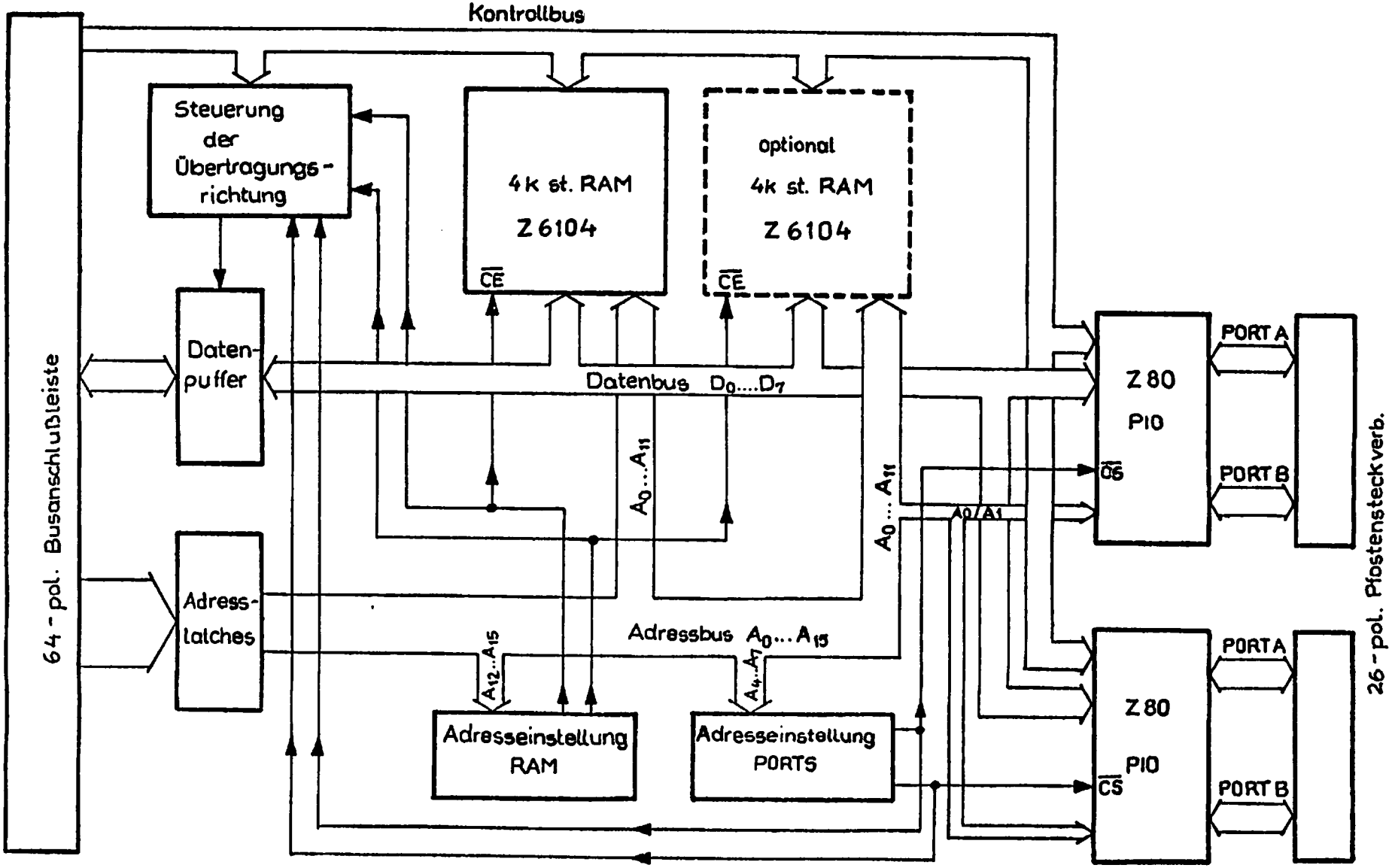
KONTRON
ELEKTRONIK GMBH



Z80
MIKROCOMPUTER-SYSTEME

II KIT/P

Inhalt:



Blockschaltbild Z80-KIT/P
Teil 1 (ECB/S)

1. Einleitung

Durch die beispielhafte Erweiterungsfähigkeit des Z80-KIT kann der Anwender je nach Wunsch oder Notwendigkeit einen entsprechenden Ausbau vornehmen, z.B. durch zusätzliche Speicher-, Ein/Ausgabe oder A/D bzw. D/A-Wandler-Karten. Durch die Kompatibilität des KIT's mit dem ECB-Bus kann diese Erweiterung einfach durch Zustecken von Baugruppen der ECB-Serie erfolgen.
Durch das in der Grundausstattung enthaltene Cassetten-Interface ist es darüber hinaus möglich, Programme auch in nichtflüchtiger Form abzuspeichern. Zur nichtflüchtigen Abspeicherung von Programmen in residenter Form benützt man programmierbare Nur-Lese-Bausteine (PROM's). Geräte zur Programmierung solcher PROM's sind allerdings meist relativ teuer (ca. 4000,– DM).
Die Firma KONTRON hat daher einen kostengünstigen Programmierzusatz KIT/P für den Z80-KIT entwickelt, der technisch und vor allen Dingen preislich äußerst attraktiv ist.

2. Überblick über die Möglichkeiten des KIT/P

Der KIT/P PROM-Programmer ist für das Programmieren von UV-löschbaren PROM's ausgelegt worden. Diese PROM's können beliebig oft gelöscht und wieder programmiert werden.
Folgende Typen sind mit dem KIT/P programmierbar:

Der KIT/P PROM-Programmer ist für das Programmieren von UV-löschbaren PROM's ausgelegt worden. Diese PROM's können beliebig oft gelöscht und wieder programmiert werden.
Folgende Typen sind mit dem KIT/P programmierbar:

EPROM	i2704	½k×8	PROM's mit + 5V, – 5V + 12V
EPROM	i2708	1k×8	Versorgungsspannung
EPROM	i2758	1k×8	PROM's mit einer + 5V
EPROM	i2716	2k×8	Versorgungsspannung
EPROM	Z6716	2k×8	

Durch den modularen Aufbau der Hardware (Programmiermodul) und der Software (Unterprogramm-Aufruf-Technik) ist es möglich, den KIT/P auch auf andere Typen zu erweitern.

3. Schaltungsbeschreibung des KIT/P

3.1. Hardwareausstattung des KIT/P

Die Hardware des KIT/P besteht aus folgenden zwei Komponenten:

3.1.1. Speicher-I/O-Erweiterung

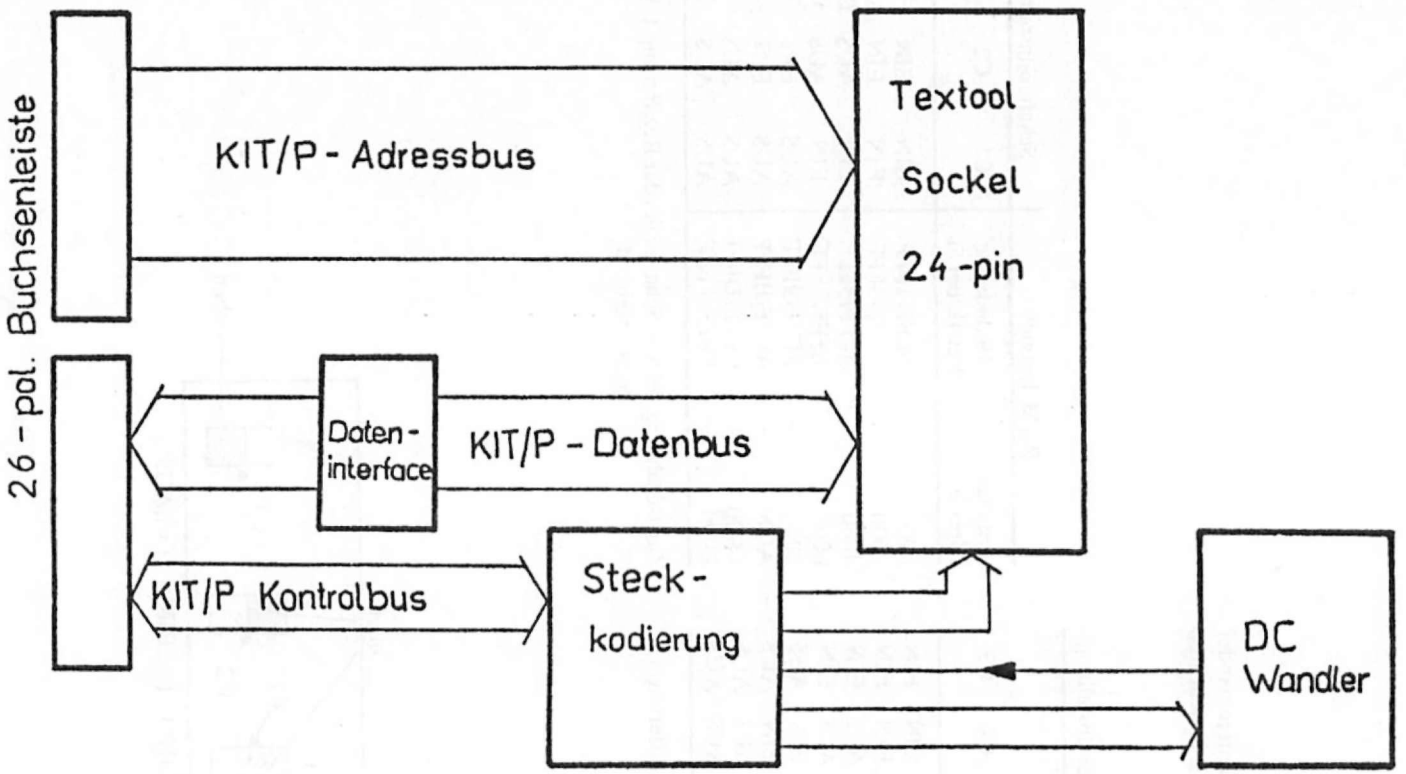
Europakarte steckbar auf einen der fünf Steckplätze des KIT,
- ☐ 2×Z80-PIO ≙ 4 parallele 8bit I/O Ports mit Quittungsleitungen
- ☐ 4 kByte statisches RAM (austauschbar auf 8 kB)
- ☐ Einstellung der Speicheranfangsadresse über Kippschalter in Schritten von 8 k
- ☐ Einstellung der Portadressen über Kippschalter
- ☐ sämtliche Signale mit Busanschlußwert von einer TTL-LS-Last
- ☐ alle Adressen zwischengespeichert
- ☐ Daten über bidirektionalen Schmitt-Trigger-Tri-State-Buffer 74LS245 geführt
- ☐ Ausgangsseitig (= PIO Ein/Ausgänge) zwei 26-polige Stiftleisten

3.1.2. PROM-Programmier-Modul

Programmiermodul, ca. 17×5 cm, als „Huckepackplatine" aufsteckbar
- ☐ 24-poliger Textool Sockel zur Aufnahme der PROM's
- ☐ Kodierstecker zur Auswahl des Promtyps.*
- ☐ Erzeugung der Programmierspannung von 26V mittels DC-Wandler.
  Dadurch bei Verwendung von „5 V-EPROM's" keine zusätzliche Stromversorgung notwendig!
- ☐ Verbindung zu Speicher-I/O-Erweiterung über zwei Buchsenleisten, passend zu den Stiftleisten.
- ☐ Eindeutigkeit der Aufsteckposition durch Kodierstifte auf der Speicher-I/O-Erweiterung gewährleistet.
- ☐ Zuführung der Versorgungsspannungen von außen.
- ☐ Möglichkeit zur Abnahme der +5 V vom KIT direkt**

* Es gibt drei gekennzeichnete Möglichkeiten, den Kodierstecker aufzubringen.
2704 gilt für i2704 EPROM's
2708 gilt für i2708 EPROM's
2716 gilt für i2758, i2716 und Z6716 EPROM's

** Diese Möglichkeit empfiehlt sich, wenn nur PROM's mit einer einzigen +5 V Versorgung verwendet werden sollen. In diesem Fall kann auf dem PROM-Programmierer-Modul ein Jumper J eingelötet werden.



Blockschaltbild Z80-KIT/P
Teil 2 (ECB/P)

3.2. Adressenbelegung

3.2.1. Adreßeinstellung des RAM-Bereiches

Der RAM-Bereich kann mit einem dreifachen Umschalter U1 in Schritten von 8k verändert werden.

3.2.2. Adreßeinstellung der PORT's

Die beiden PIO's liegen adressenmäßig fest hintereinander und können über den dreifachen Umschalter U2 in Schritten zu je 4 Ports eingestellt werden.

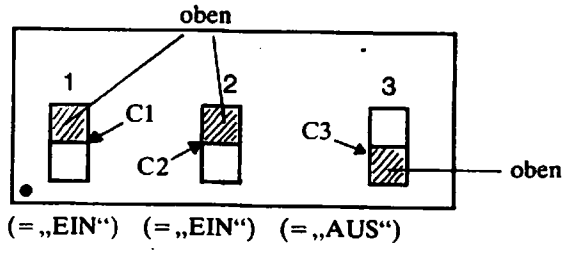
Portadressen (Data/Control)				Schaltereinstellung		
PIO I (STC)		PIO II (STB)				
PORT A D/C	PORT B D/C	PORT A D/C	PORT B D/C	C1	C2	C3
0C/0E	0D/0F	1C/1E	1D/1F	EIN	EIN	EIN
2C/2E	2D/2F	3C/3E	3D/3F	AUS	EIN	EIN
4C/4F	4D/4F	5C/5E	5D/5F	EIN	AUS	EIN
6C/6E	6D/6F	7C/7E	7D/7F	AUS	AUS	EIN
8C/8E	8D/8F	9C/9E	9D/9F	EIN	EIN	AUS
AC/AE	AD/AF	BC/BE	BD/BF	AUS	EIN	AUS
CC/CE	CD/CF	DC/DE	DD/DF	EIN	AUS	AUS
EC/EE	ED/EF	FC/FE	FD/FF	AUS	AUS	AUS

Zur Beachtung: (=„EIN") ≙ Schalter bei der Bezifferung 1, 2 oder 3 herausstehend

RAM-Bereich			Schaltereinstellung		
Anfangsadresse		Endadresse bei 4k/bei 8k	C1	C2	C3
0000	–	0FFF/1FFF	EIN	EIN	EIN
2000	–	2FFF/3FFF	EIN	EIN	AUS
4000	–	4FFF/5FFF	EIN	AUS	EIN
6000	–	6FFF/7FFF	EIN	AUS	AUS
8000	–	8FFF/9FFF	AUS	EIN	EIN
A000	–	AFFF/BFFF	AUS	EIN	AUS
C000	–	CFFF/DFFF	AUS	AUS	EIN
E000	–	EFFF/FFFF	AUS	AUS	AUS

Zur Beachtung: EIN ≙ Schalter bei der Bezifferung 1, 2 oder 3 herausstehend

z.B.



(=„EIN") (=„EIN") (=„AUS")

3.3. Pinbelegung

3.3.1. Bus-Stecker STA

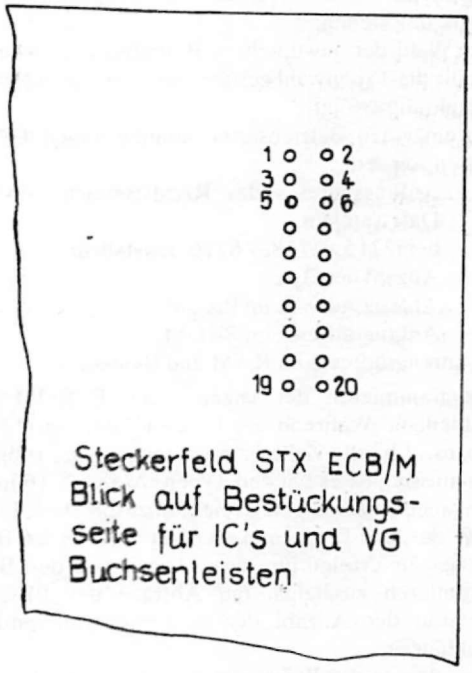
64-poliger Steckverbinder VG 95324

Es handelt sich hier um eine Belegung gemäß der ECB-BUS-NORM, wobei selbstverständlich nur die erforderlichen PIN's angeschlossen sind.

Benennung	Stecker PIN	Bezeichnung
A 0	5c	Adresse 0
A 1	7c	Adresse 1
A 2	6a	Adresse 2
A 3	6c	Adresse 3
A 4	7a	Adresse 4
A 5	8a	Adresse 5
A 6	9a	Adresse 6
A 7	9c	Adresse 7
A 8	8c	Adresse 8
A 9	30a	Adresse 9
A 10	18c	Adresse 10
A 11	17c	Adresse 11
A 12	27c	Adresse 12
A 13	29a	Adresse 13
A 14	18a	Adresse 14
A 15	28c	Adresse 15
D 0	2c	Daten 0
D 1	14c	Daten 1
D 2	4c	Daten 2
D 3	4a	Daten 3
D 4	5a	Daten 4
D 5	2a	Daten 5
D 6	3a	Daten 6
D 7	3c	Daten 7
M 1	20a	Maschinenzyklus 1
MRQ	30c	Memory Request
IORQ	27a	IN/OUT Request
RD	24c	Read
WR	22c	Write
INT	21c	Interrupt
IEI 1	11c	Int. enable in
IEO 1	16c	Int. enable out
0	29c	Clock 2,45 MHz
+5	1 a, c	
GND	32 a, c	

3.3.2. Ein/Ausgabestecker STB und STC

Stift-Nr.	26-pol. WWP-Steckerfeld
1	A_0
2	A_1
3	A_2
4	A_3
5	A_4
6	A_5
7	A_6
8	A_7
9	B_0
10	B_1
11	B_2
12	B_3
13	B_4
14	B_5
15	B_6
16	B_7
17	ASTB
18	BSTB
19	ARDY
20	BRDY
21	NC
22	NC
23	+5V
24	+5V
25	GND
26	GND



4. Beschreibung des KIT/P-Programms

4.1. Programmumfang

- ☐ folgende vier Betriebsarten (Modes) sind möglich:
  - Programmieren (Quelle = RAM)
  - Duplizieren (Quelle = PROM)
  - Listen
  - Verifizieren (Vergleichen)
- ☐ Fehlermeldung bei nicht aufgestecktem Kodierstecker.
- ☐ Anfangsadresse des RAM beim Programmieren und Verifizieren frei wählbar.
- ☐ Beim Programmieren von i2758, i2716 oder Z6716 zusätzlich Anfangsadresse des PROM's und Blocklänge (= Anzahl der zu programmierenden Bytes) wählbar.
- ☐ Fehlermeldung bei Überschreitung der PROM-Größe bei i2716 und Z6716
- ☐ Fehlermeldung bei nicht vollständig gelöschtem PROM
- ☐ Überprogrammieren möglich
- ☐ Fehlermeldung bei Unterschieden zw. PROM-Inhalt und RAM-Inhalt nach dem Programmiervorgang.
- ☐ Anfangsadresse des PROM's beim Listen beliebig einstellbar.
- ☐ Voreinstellung von RAM- und PROM-Startadresse mit 0000 und der Blocklänge entsprechend dem gewählten Typ beim Einsprung in die Betriebsarten Programmieren und Duplizieren
- ☐ Lieferung des KIT/P-Programmes auf Cassette
- ☐ Optionale Lieferung des KIT/P Programms im PROM

4.2. Allgemeine Beschreibung

Das KIT/P-Programm beginnt mit der Frage nach der gewünschten Betriebsart (Mode), wobei die Möglichkeiten

a) Programmieren – Daten im Speicher
b) Duplizieren – Daten in einem PROM
c) Listen – Lesen eines PROM-Inhaltes
d) Verifizieren – Vergleich PROM/Speicher

zur Verfügung stehen.

Vor der Wahl der gewünschten Betriebsart muß der Kodierstecker für die Typenwahl gesteckt werden, da andernfalls eine Fehlermeldung erfolgt.

Zu den einzelnen Betriebsarten können folgende Parameter angegeben werden:

zu a) – Anfangsadresse des RAM-Bereiches in dem die Daten stehen
bei i2716, i2758, Z6716, **zusätzlich:**
– Anzahl der Bytes
– Anfangsadresse im PROM

zu c) – Anfangsadresse im PROM

zu d) Anfangsadresse im RAM und Blocklänge

Das Programmieren der angegebenen EPROM-Typen ist unterschiedlich. Während die Typen i2704 und i2708 immer als Ganzes, d.h. alle Zellen zusammenhängend programmiert werden müssen, ist es bei den Typen i2758, i2716 und Z6716 möglich, auch einzelne Teilbereiche anzusprechen. Im Extremfall heißt das, das Programmieren eines einzelnen Bytes. Auf Grund dessen erfolgt für diese Typen bei der Betriebsart Programmieren zusätzlich die Abfrage der PROM Startadresse und der Anzahl der zu programmierenden Bytes (= Blocklänge).

Vor dem Beginn der Programmierroutine wird der Inhalt des zu programmierenden PROM's geprüft. Ist es nicht gelöscht (d.h. weist nur ein Wort nicht den Wert FF auf), so erscheint eine entsprechende Fehlermeldung. Ein Überprogrammieren ist jedoch möglich.

Die eigentlichen „Brenn Routinen" erfolgen nach den Angaben der Hersteller. Für die Typen i2758, i2716 und Z6716 wurde zusätzlich ein sog. QUICK EXIT eingebaut. Dabei wird **vor** dem Brennvorgang jeder Zelleninhalt mit dem einzuprogrammierenden Wert verglichen. Stimmt er überein, geht das Programm sofort zur nächsten Zelle über.

Im gegenteiligen Fall wird die entsprechende Zelle gegebenenfalls bis zu maximal 5 mal „gebrannt". Dadurch ergeben sich variable Programmierzeiten.

Die typischen Zeiten liegen bei gelöschten EPROM's im Bereich von Minuten.

Während des Programmiervorganges erlöschen alle Anzeigen.

Anschließend an die Brennroutine wird der PROM-Inhalt mit dem RAM-Inhalt verglichen. Bei einer Diskrepanz erfolgt eine Unterbrechung sowie eine Fehlermeldung mit der Möglichkeit, beide Werte (jeweils Adresse und Daten) anzuzeigen. Erst nach einem entsprechenden Kommando wird die Verifizierroutine fortgesetzt.

Ist die Verifizierroutine nach 0 oder n Fehlern abgeschlossen, kehrt das Programm an eine Stelle zurück, die ein sofortiges nochmaliges Programmieren mit den gleichen Daten und Parametern zuläßt. Es kann also jeweils sofort ein weiteres PROM mit den gleichen Werten programmiert werden, bzw. ein sofortiges Nachprogrammieren des gleichen PROM's ist möglich.

4.3. Dialog Programm, Benutzer

Das Programm bringt nach seinem Start zur Festlegung der Parameter Systemmeldungen bzw. -fragen in kodierter Form zur Anzeige und erwartet eine entsprechende Tasten-Eingabe. Treten im Verlauf der Vorbereitung Fehler in Bezug auf Promgröße oder Promtyp auf, so erscheint ebenfalls in kodierter Form eine Fehlermeldung. Desgleichen bei nicht gelöschtem PROM oder bei einem Programmierfehler, der bei der abschließenden Prüfung entdeckt wird. Auch hier erwartet das System eine entsprechende Eingabe.

NB: In allen Teilen des Programmes führen nur dafür vorgesehene Tasten zu einer Aktion. Alle anderen Tasten werden ignoriert und haben keinerlei Wirkung.

Grundsätzlich gilt:

EX Taste: Abschluß des aktuellen und Einleitung des nächsten Abschnittes
0 Taste: Rücksprung in den vorhergegangenen Abschnitt
A Taste: Anzeige eines im aktuellen Abschnitt gültigen Wertes.

Wurde durch Drücken der Taste A ein Wert zur Anzeige gebracht, so **kann** er durch Eintippen einer anderen Zahl verändert werden. Die Zahlen werden dabei sequentiell von rechts nach links in die Anzeige geschoben. Erst wenn der gewünschte Wert angezeigt wird, kann mittels EX Taste dieser Wert abgespeichert und der nächste Abschnitt eingeleitet werden.

4.4. Zusammenstellung der Systemmeldungen und Eingabemöglichkeiten

4.4.1. Auswahl der Betriebsart

Anzeige Anzeige	Aktivität des Benutzers	Erläuterung
Co?		Kommando? Frage nach der gewünschten Betriebsart (Mode)
	Taste 0	Rücksprung ins Monitorprogramm
	Taste 1	Sprung zum Programmieren
	Taste 2	Sprung zum Duplizieren
	Taste 3	Sprung zum Listen
	Taste 4	Sprung zum Verifizieren (Check)

4.4.2. Programmteil Programmieren

Anzeige	Aktivität des Benutzers	Erläuterung
– P 1		Gib Startadresse des RAM-Bereiches!
	Taste 0	Rücksprung zu „Co?"
	Taste EX	Sprung zu ‚P2' (i2758, i2716 und Z6716) bzw. ‚P4'? (i27040 (RAM Startadresse: Voreinstellung = 0000)
	Taste A	Sprung zu „Anzeige RAM-Startadresse"
– Anzeige RAM-Startadresse	Taste 0...F	Einlesen ins Anzeigen register
	Taste EX	Abspeichern RAM Startadresse und Sprung zu ‚P2' (i2758, i2716 und Z6716) bzw. ‚P4' (i2704/08).
– P 2		Gib Blocklänge (nur i2758, i2716 und Z6716)!
	Taste 0	Rücksprung zu ‚P1'
	Taste EX	Sprung zu ‚P3' (Blocklänge: Voreinstellung = 0800)
	Taste A	Sprung zu ‚Anzeige Blocklänge'
– Anzeige Blocklänge	Taste 0...F	Einlesen ins Anzeigenregister
	Taste EX	Sprung zu ‚P3'
– P 3		Gibt PROM Startadresse!
	Taste 0	Rücksprung zu ‚P2'
	Taste EX	Sprung zu ‚P4?' (PROM Startadresse: Voreinstellung = 0000)
	Taste A	Sprung zu ‚Anzeige PROM-Startadresse'
– Anzeige PROM-Startadresse	Taste 0...F	Einlesen ins Anzeigenregister
	Taste EX	Sprung zu ‚P4?'
– P 4?		Fertig zum Programmieren?
	Taste 0	Rücksprung zu ‚P3' (2716 etc.) bzw. ‚P1' (2704/08)
	Taste EX	Sprung zur Programmierroutine. Anschließend Rückkehr zu ‚P4?'

4.4.3 Programmteil Duplizieren

Anzeige	Aktivität des Benutzers	Erläuterung
– D 1		Original PROM einsetzen!
	Taste 0	Rücksprung zu ‚Co?'
	Taste EX	Einlesen PROM-Daten, RAM-Startadresse = 6800H und Sprung zu ‚D2'
– D 2		PROM's wechseln!
	Taste 0	Rücksprung zu ‚D1'
	Taste EX	Sprung zu ‚D3?'
– D 3?		Fertig zum programmieren?
	Taste 0	Rücksprung zu ‚D2'
	Taste EX	Sprung zur Programmierroutine. Anschließend Rückkehr zu ‚D3?'

4.4.4 Listen

Anzeige	Aktivität des Benutzers	Erläuterung
– L 1		Gib PROM Startadresse!
	Taste 0	Rücksprung zu ‚Co?'
	Taste EX	Sprung zur Anzeigenroutine (PROM Startadresse: Voreinstellung = 0000)
	Taste A	Sprung zu ‚Anzeige PROM Startadresse'
– Anzeige PROM Startadresse		
	Taste 0...F	Einlesen ins Anzeigenregister
	Taste EX	Sprung zur Anzeigenroutine
– Anzeigenroutine		Zeigt die aktuelle PROM-Adresse und Daten an
	Taste 0	Rücksprung zu ‚L1'
	Taste IDM	Adresse inkrementieren neue Adresse und Daten anzeigen

4.4.5 Verifizieren (Checken)

Anzeige	Aktivität des Benutzers	Erläuterung
– C 1		Gib RAM Startadresse!
	Taste 0	Rücksprung zu ‚Co?'
	Taste EX	Sprung zu ‚C2'
	Taste A	Sprung zu ‚Anzeige RAM Startadresse'
– Anzeige RAM-Startadresse		
	Taste 0...F	Einlesen in Anzeigenregister
	Taste EX	Sprung zu ‚C2'
– C 2		Gib Blocklänge!
	Taste 0	Rücksprung zu ‚C1'
	Taste EX	Sprung zu ‚C3?'
	Taste A	Sprung zu ‚Anzeige Blocklänge'
– Anzeige Blocklänge		
	0...F	Einlesen in Anzeigenregister
	Taste EX	Abspeichern Blocklänge und Sprung zu ‚C3?'
– C 3?		Fertig zum Verifizieren?
	Taste 0	Rücksprung zu ‚C1'
	Taste EX	Sprung zur Verifizierroutine, anschließend zurück zu ‚C3?'

4.4.6 Fehlermeldungen

Anzeige	Aktivität des Benutzers	Erläuterung
– F 1		PROM-Typ undefiniert
	Taste 0	Rücksprung zu ‚Co?'
– F 2		Größenüberschreitung
	Taste 0	Rücksprung zu ‚P3'
– F 3 ?		PROM nicht leer, trotzdem programmieren?
	Taste 0	Rücksprung zu ‚Co?'
	Taste EX	Sprung zur Programmierroutine; anschließend nach ‚P4?' oder ‚D3?' zurück
– F 4		Programmierfehler
	Taste 0	Rücksprung zu ‚P4?' bzw. ‚D3?'
	Taste EX	Weiter in Verifizierroutine
	Taste A	Sprung zu ‚Anzeige PROM'
– Anzeige PROM		Anzeige PROM Adresse und Daten
	Taste 0	Rücksprung zu ‚P4?' bzw. ‚D3?'
	Taste EX	Weiter in Verifizierroutine
	Taste A	Sprung zu ‚Anzeige RAM'
– Anzeige RAM		Anzeige RAM Adresse und Daten
	Taste 0	Rücksprung zu ‚P4?' oder ‚D3?'
	Taste EX	Weiter in Verifizierroutine
	Taste A	Sprung zu ‚Anzeige PROM'

5. Laden und Starten des Programms

a) Programm auf Cassette
Das PROM-Programmierprogramm wird auf Cassette geliefert und muß vor Beginn des Programmierens geladen werden. Siehe dazu LOAD Vorgang im KIT Anwenderhandbuch.
Da das Programm auf die Anfangsadresse

6000 H

assembliert ist, muß der Anfang des RAM-Bereichs auf diesen Wert eingestellt werden. Beim Duplizieren werden die Daten im RAM ab

6800 H

abgespeichert. Außerdem werden die Port-Adressen

2C H
2D H
3C H
3D H

verwendet und sind demzufolge entsprechend einzustellen.

Schalter-Einstellung für Standardprogramm

		C1	C2	C3
RAM Bereich	(U1)	EIN	AUS	AUS
I/O PORTS	(U2)	AUS	EIN	EIN

Zuerst müssen die beiden KIT/P-Karten zusammengesteckt werden, um dann gemeinsam in einem der KIT-Steckplätze untergebracht zu werden. Anschließend wird das Programm mittels LOAD in den angegebenen Bereich geladen und gestartet. Dazu setzen Sie den Programmzähler auf 6000 H und drücken Die Taste START.

b) Programm im Festwertspeicher
Wird an Stelle der auf Cassette gelieferten Software die PROM-residente Version (Z80 KIT/PDT) eingesetzt, so ist folgendes zu beachten:
Das 2 kByte umfassende Software-Erweiterungs-Paket Z80 KIT/PDT wird auf der ECB/K auf Sockel A3 untergebracht.
Die Startadresse des KIT/P-Programmes ist 0800 H. Die Benützung des RAM-Bereiches beim Duplizieren geschieht, wie bei der Cassettenversion. Die Einstellung des RAM-Bereiches und der PORT's erfolgt demzufolge wie unter a) beschrieben.
Nach Setzen des Programmzählers auf 0800 H kann das Programm durch Drücken der START-Taste zum Ablaufen gebracht werden.

NB.: Die promresidente Version des KIT/P-Programms (Z80-KIT/PDT) greift auf Routinen des Betriebsprogramms zurück. Dabei ist es unerheblich, ob es sich um das Standardbetriebsprogramm oder die TV-Version (Z80 KIT/TV) handelt. Ein Betrieb ohne eines der Betriebsprogramme ist nicht möglich.

Achtung:
Bevor Sie darangehen, Ihr erstes PROM zu programmieren, empfiehlt es sich, die einzelnen Betriebsarten **ohne** PROM durchzuspielen. Testen Sie dabei die Möglichkeiten, die unter 4.4. angegeben sind!
Erst wenn dies ohne Störungen gelingt, können Sie sicher sein, daß Ihre Speicherkarte fehlerfrei funktioniert. Beginnen Sie dann damit, ein Ihnen bekanntes PROM auszulesen. Gelingt auch dies, kann die erste Programmierung durchgeführt werden.

6. Zusammenbau der Platinen

6.1. Allgemeine Hinweise

Beim Aufbau ist, wie schon beim KIT, allergrößte Vorsicht in Hinsicht auf die Verlötung geboten.
Erfahrungsgemäß sind in mehr als 90% aller Fälle, in denen ein KIT „nicht funktioniert", Löt-Kurzschlüsse die Ursache!

- Verwenden Sie nur einen Lötkolben mit sehr dünner Spitze!
- Suchen Sie die günstigste Art, die Lötspitze am Lötpunkt anzusetzen!
- Vermeiden Sie dabei, benachbarte Punkte zu berühren!
- Seien Sie sparsam mit Lötzinn!!!!

Der Aufbau ist sinngemäß zur Beschreibung im Abschnitt I durchzuführen. Die Bestückungspläne liegen der Verpackung bei.

Achtung:
Als Kodierstecker wird die mitgelieferte 20-polige Buchsenleiste verwendet. Zunächst müssen jedoch die jeweils nebeneinander liegenden Lötanschlüsse miteinander verlötet werden. Es entstehen somit 10 in einer Reihe liegende Kurzschlußbrücken.

Auf dem KIT/P Programmiermodul müssen die beiden 26poligen Buchsenleisten STB und STC von der Lötseite her eingebaut werden!

Justierung der Programmierspannung
Die zum Programmieren notwendige Spannung V_{pp} müssen bei den Typen

2704 u. 2708 zwischen 25 V und 27 V

und bei den Typen

2758, 2716 u. 6716 zwischen 24 V und 26 V

liegen!

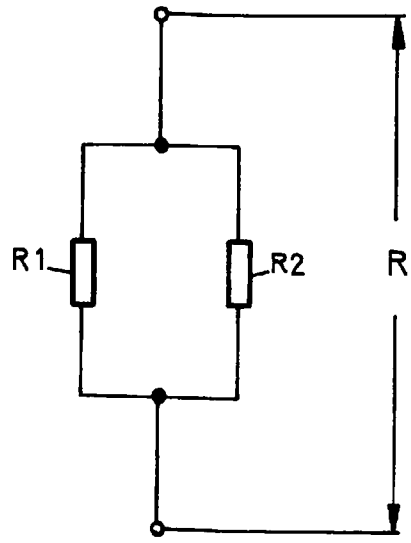
NB.: Ein Überschreiten dieser Werte kann zur Zerstörung des PROM's führen.

Der DC Wandler des Programmiermoduls ist auf eine Spannung von 25,6 V ausgelegt. Die Einstellung erfolgt über die Widerstände R 36 und R 37 nach folgender Formel:

$$V_{PP} = (1 + \frac{R_{36}}{R_{37}}) \cdot 1{,}22\ V$$

Durch Bauteiltoleranzen ist eine Abweichung vom Sollwert möglich. Eine Korrektur ist, sofern notwendig, durch Parallelschalten von entsprechend gewählten Widerständen parallel zu R_{36} bzw. R_{37} zu erreichen.
Widerstand R einer aus R_1 und R_2 gebildeten Parallelschaltung:

$$R = \frac{R_1 \cdot R_2}{R_1 + R_2}$$



6.2. Stücklisten zu KIT/P

6.2.1. Stückliste zu Speicher-I/O-Erweiterung (= „Z80-ECB/S“)

Stck.	Bezeichnung	Bauteil Nr.	Bemerkung
Mikrocomputer Bausteine in leitendem Schaumstoff			
2	Z80-PIO	E1..E5, D1..D5	Z80 progr. Parallel-Schnittstelle
1	HM 7611	E6	Steuerbaustein
8	Z6104-4H	B1...B8	4k×1 Bit Speicherbausteine
1	SN74LS04	D8	
2	SN74LS08	D6, E7	
1	SN74LS20	D7	
2	SN74LS174	A2, A3	
1	SN74LS175	A8	
1	SN74LS245	A4	
1	SN74LS367	A5	
1	SN4931	A6	
2	74C03	E8, A7	Umschalter
Widerstände und Kondensatoren			
3	Widerstand 4,7k	R1..R3	gelb, viol., rot
5	Tantalelko 4,7u/16V	C1..C5	
Elektro-Mechanische Bauteile			
1	Messerleiste VG	STA	64polig A-C
2	Stiftleiste	STB, STC	26polig
2	Distanzstift		Kodierstift
2	IC Sockel		40polig
8	IC Sockel		18polig
1	Ausziehgriff		
2	Schrauben		M2,5×16
2	Muttern		M2,5

6.2.2. Stückliste zum KIT Programmiermodul (=„ECB/P“)

Stck.	Bezeichnung	Bauteil Nr.	Bemerkung
Halbleiter			
1	SN7407		
1	STL497		
2	MC14503B		
4	MC75453P		
Widerstände und Kondensatoren			
1	Widerstand 1,2k, ⅓W		braun, rot, rot
8	Widerstand 39k ⅓W	R25–32	or., weiß, or.
1	Widerstand 24k ⅓W	R36	rot, gelb, or.
2	Widerstand 10k ⅓W	R33, R34	br., schw., or.
3	Widerstand 4,7k ⅓W	R13, R23, R24	gelb, viol., rot
1	Widerstand 2,7k ⅓W	R35	rot, viol., rot
1	Widerstand 1k ⅓W	R14	br., schw., rot
1	Widerstand 1,2k ⅓W	R37	braun, rot, rot
1	Widerstand 100 ⅓W	R38	br., schw., br.
1	Widerstand 50 ⅓W	R39	gr., schw., schw.
1	Widerstand 1 1W	R40	br., schw., gold
1	Ersil-Wstr. 1,2k ⅓W	R15..R22	
1	Ersil-Wstr. 4,7k ⅓W	RR1..R12	
3	2N2222	T1, T2, T4	
1	2N2907	T3	
1	EB 1000u/40V	C2	Aluminium
1	EK 100u/25V	C3	
1	Tantalelko 4,7u/16V	C5	
1	1N5818	D1	
1	Kondensator 2nF	C1	
1	Kondensator 150pF	C4	
Elektro-Mechanische Bauteile			
1	TEXTOOL-Sockel 24polig		
1	Buchsenleiste 20polig		
3	Stiftleiste 20polig		
1	Induktivitätsspule		
2	Buchsenleiste 26polig	STB, STC	

Achtung: STB und STC von Lötseite her einbauen!

7. Speichertestprogramme

7.1. Testprogramm 1

7.1.1. Programmbeschreibung

Im Anschluß finden Sie ein Speichertestprogramm, das es Ihnen erlaubt, jeden beliebigen Speicherbereich mittels MEM READ und MEMWRITE-Operationen zu testen. Der Speicher wird hierzu jeweils mit den Bitmustern

00
01
02
.
.

bis FF

vollständig geladen und anschließend ausgelesen und verglichen. Bei Auftreten eines Fehlers erfolgt ein Rücksprung ins Monitorprogramm mit ERROR Anzeige. Die Adresse der ersten fehlerhaften Speicherzelle, das eingeschriebene Bitmuster und die gelesenen fehlerhaften Daten können angezeigt werden. Nach der Prüfung mit FF erfolgt nach fehlerlosem Durchlauf ein Rücksprung ins Monitorprogramm **ohne** ERROR Anzeige.
Das Programm wird mit der Anfangsadresse 3C50H beginnend in den RAM-Bereich eingegeben (siehe INPUT im KIT-Anwenderhandbuch).
Die Adresse der ersten zu testenden Speicherzelle wird in die Speicherzellen 3C47/3C48 abgelegt, die Anzahl der zu testenden Zellen in 3C49/3C4A.

Anschließend wird das Programm gestartet (siehe KIT Anwenderhandbuch). Beim ersten auftretenden Fehler erfolgt ein Rücksprung ins Monitorprogramm mit ERROR Anzeige.
Die Werte Fehladresse
erwartete Bitkombination
und
gefundene Bitkombination
stehen in den Registern
IX
A'
und
A
und können über das DIS-Kommando angezeigt werden.

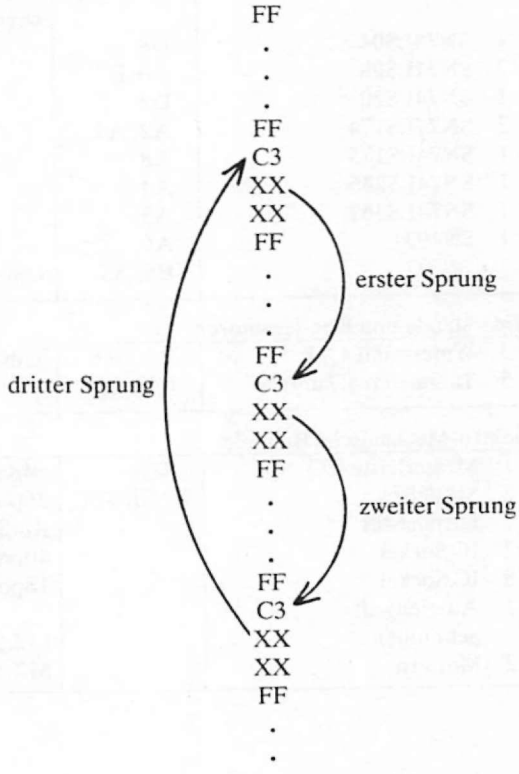
Tritt kein Fehler auf, kann nach der Rückkehr ins Monitorprogramm sofort wieder gestartet werden, oder zuvor ein neues Wertepaar (Startadresse Länge) eingegeben werden.

7.1.2. Siehe nächste Seite

7.2. Testprogramm 2

7.2.1. Programmbeschreibung

Um einen Speicher auf seine Funktionsfähigkeit zu testen, empfiehlt es sich, neben normalen Schreib- und Leseoperationen auch M1 Leseoperationen durchzuführen.
Folgendes sehr einfaches Programm ermöglicht diese Prüfung. Man füllt den zu testenden Speicher, ebenfalls mit einer Unterroutine vollständig mit FF (siehe vorheriges Programm). An willkürlichen Stellen werden sodann Sprungbefehle eingesetzt, die eine fortlaufende Kette untereinander bilden müssen.



Wird dieses Programm gestartet (PC auf den Opcode eines Sprunges), so ergibt sich eine unendliche Programmschleife, bestehend aus der ständigen Wiederholung der Sprungkette.
Tritt ein Fehler in der Adressierung, oder bei einer Leseoperation auf, so wird als unvermeidliche Folge eine mit FF geladene Zelle zum Sprungziel. FF als Opcode gelesen, veranlaßt den Prozessor jedoch zu einem RESTART 38, d.h. der Prozessor setzt seine Programmabarbeitung bei der PC-Adresse 38 fort.
Der Rücksprung ins Monitorprogramm und eine ERROR Anzeige sind die Folge.
Eine Erweiterung dieses einfachen Programms könnte darin bestehen, in einem zyklischen Modus neue Sprünge zu setzen und alte dafür zu löschen.

```
                        RAMTEST         200778              PAGE   1
LOC   OBJ CODE M STMT SOURCE STATEMENT                       ASM 4.5

                    1  ;*****************************************************
                    2  ;*                                                   *
                    3  ;*                                                   *
                    4  ;*                  RAM TEST                         *
                    5  ;*                                                   *
                    6  ;*                                                   *
                    7  ;*****************************************************
                    8          ;
                    9          ;STARTADRESSE IN 3C47H/3C48H
                   10          ;BLOCKLAENGE IN  3C49H/3C4AH
                   11          ;
                   12          ;PROGR.STARTADR. = 3C50H
                   13          ;
                   14          ;
3C50               15          ORG 3C50H
                   16          ;
                   17  TEST:
3C50  3E00         18          LD A,00 ;ERSTE BITKOMBINATION
                   19          ;
                   20          ;
                   21  MTEST1:
3C52  ED4B493C     22          LD BC,(3C49H)    ;BLOCKLAENGE LADEN
3C56  2A473C       23          LD HL,(3C47H)    ;STARTADRESSE LADEN
3C59  ED5B473C     24          LD DE,(3C47H)
3C5D  13           25          INC DE           ;STARTADRESSE+1
3C5E  77           26          LD (HL),A        ;BITMUSTER IN ERSTE
                   27                           ;SPEICHERSTELLE
3C5F  EDB0         28          LDIR             ;UND IN ALLE RESTLICHEN
                   29          ;
                   30          ;
3C61  ED4B493C     31          LD BC,(3C49H)    ;BLOCKLAENGE LADEN
3C65  2A473C       32          LD HL,(3C47H)    ;STARTADRESSE LADEN
                   33          ;
                   34          ;
                   35  MTEST2:
3C68  EDA1         36          CPI              ;VERGLEICHEN
3C6A  2009         37          JR NZ,ERROR      ;WENN NICHT GLEICH
3C6C  EA683C       38          JP PE,MTEST2     ;WENN GLEICH,ZUR
                   39                           ;NAECHSTEN SPEICHERZELLE
                   40          ;
                   41          ;
3C6F  3C           42          INC A            ;NAECHSTES BITMUSTER
3C70  20E0         43          JR NZ,MTEST1     ;WENN NICHT 00
                   44                           ;DANN NEUER DURCHLAUF
3C72  C30B00       45          JP 000BH                ;SONST RUECKSPRUNG
                   46          ;
                   47          ;
                   48  ERROR:
3C75  32173C       49          LD (3C17H),A     ;GEFUNDENES BITMUSTER
                   50                           ;NACH REGISTER A
3C78  2B           51          DEC HL
3C79  22063C       52          LD (3C06H),HL    ;FEHLADRESSE INS REGISTER
                   53                           ;IX
3C7C  7E           54          LD A,(HL)
3C7D  320F3C       55          LD (3C0FH),A     ;ERWARTETES BITMUSTER
                   56                           ;NACH REGISTER A'
                   57          ;
3C80  C33800       58          JP 0038H               ;RUECKSPRUNG MIT ERROR
```

8. Beispiele für einen Programmiervorgang

8.1. Programmieren eines 2716

Zu programmieren sind 200 Bytes; erstes Byte auf RAM Adresse 4300 soll auf PROM Adresse 0000 kommen.

Meldung	Eingabe	Kommentar
Co?		
	1	
F 1		kein Kodierstecker aufgesetzt! Aufstecken!
	0	
Co?		
	1	
P 1		= Gib RAM Startadresse
	A	
0000		
	4	
0004		
	3	
0043		Eingabe 4300
	0	
0430		
	0	
4300		
	EX	4300 wird abgespeichert
P 2		= Gib Blocklänge
	A	
0800		Voreinstellung auf 800
	2	
8002		
	0	
0020		versehentliche Eingabe
	0	2000 statt
0200		200
	0	
2000		
	EX	2000 wird abgespeichert
P 3		Gib PROM Startadresse
	EX	Voreinstellung 0000 wird ohne Anzeige akzeptiert
F 2		Größenfehler
	0	zurück zu P 3!
P 3		
	0	zurück zu P 2!
P 2		
	A	Anzeige Blocklänge!
2000		
	2	
0002		
	0	neue richtige Eingabe 200
0020		
	0	
0200		
	EX	
P 3		
	EX	
P 4?		Fertig zum Programmieren?
	0	man möchte P 3 doch noch sehen
P 3		
	A	
0000		
	EX	
P 4?		
	EX	Der Programmiervorgang beginnt. Dauer im Minutenbereich.
P 4?		Hier könnte der gleiche Datensatz (4300 . . . 44FF) sofort in ein weiteres PROM gebrannt werden. Bei Auftreten eines Programmierfehlers F 4 kann sofort nachprogrammiert werden, indem man mit Taste „0" nach P4? zurückkehrt und EX gibt sonst
	0	
P 3		
	0	
P 2		
	0	
P 1		
	0	
Co?		Rückkehr zu Co?

8.2. Duplizieren eines 2708

Das zu programmierende PROM sei auf Adresse 7A nicht gelöscht und enthalte 8D. Das Original-PROM enthalte dort FD.

Meldung	Eingabe	Kommentar
Co?		
	2	
D 1		Originales PROM einsetzen!
	EX	Inhalt wird abgespeichert beginnend bei Adr. 6800
D 2		PROM's wechseln!
	EX	
D 3?		Fertig zum programmieren?
	EX	
F 3?		PROM nicht leer trotzdem programmieren?
	EX	
		Der Programmiervorgang beginnt; Dauer ca. einige Minuten
F 4		Programmierfehler!
	A	
007A 8D		Anzeige PROM Adr. + PROM-Daten
	A	
687A FD		Anzeige RAM Adresse+RAM-Daten
	A	
007A 8D		Nochmal Anzeige PROM Adr.+ PROM Daten
	EX	FD kann 8D nicht ersetzen. PROM muß nochmal gelöscht werden!
D 3?		Kein weiterer Fehler gefunden
	0	
D 2		
	0	
D 1		
	0	
Co?		

8.3. Listen eines beliebigen PROM's

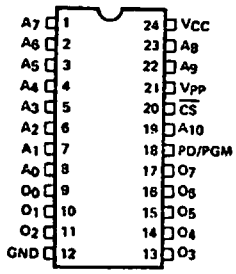
Meldung	Eingabe	Kommentar
Co?		
	3	
L 1		Gib PROM Startadresse!
	A	
0000		
	2	
0002		
	0	Eingabe 20
0020		
	EX	Startadresse wird abgespeichert
0020 XX		
	IDM	
0021 XX		jeweils Anzeige PROM Adresse
	IDM	+ PROM Daten
0022 XX		
	0	
L 1		Gib PROM Startadresse!
	A	
0020		
	0	
0200		
	3	
2003		Eingabe
	0	
0020		
	0	
0300		
	EX	
0300 XX		
	IDM	
0301 XX		
	etc.	
	0	
L 1		
	0	
Co?		

XX = Wert der entsprechenden PROM-Zelle

16 K (2 K x 8) UV Erasable Prom 2716

- Single +5V Power Supply
- Simple Programming Requirements
  - Single Location Programming
  - Programs With One 50ms Pulse
- Low Power Dissipation
  - 525mW Max. Active Power
  - 132mW Max. Standby Power
- Pin Compatible To Intel 2316E ROM
- Fast Access Time: 450ns Max.
- Inputs and Outputs TTL Compatible During Read And Program

PIN CONFIGURATION



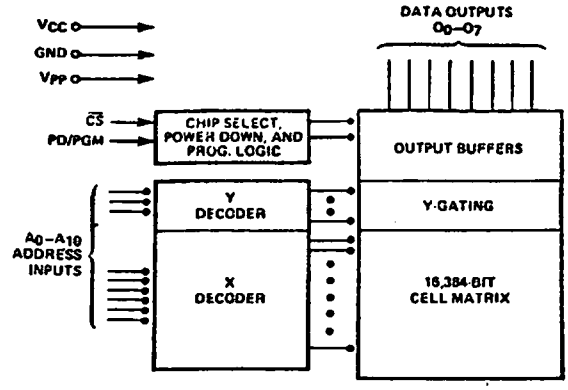
PIN NAMES

A_0–A_{10}	ADDRESSES
PD/PGM	POWER DOWN/PROGRAM
$\overline{CS}$	CHIP SELECT
O_0–O_7	OUTPUTS

MODE SELECTION

MODE \ PINS	PD/PGM (18)	$\overline{CS}$ (20)	V_{PP} (21)	V_{CC} (24)	OUTPUTS (9-11, 13-17)
Read	V_{IL}	V_{IL}	+5	+5	D_{OUT}
Deselect	Don't Care	V_{IH}	+5	+5	High Z
Power Down	V_{IH}	Don't Care	+5	+5	High Z
Program	Pulsed V_{IL} to V_{IH}	V_{IH}	+25	+5	O_{IN}
Program Verify	V_{IL}	V_{IL}	+25	+5	D_{OUT}
Program Inhibit	V_{IL}	V_{IH}	+25	+5	High Z

BLOCK DIAGRAM



Absolute Maximum Ratings*

Temperature Under Bias . . . . . -10°C to +80°C
Storage Temperature . . . . . -65°C to +125°C
All Input or Output Voltages with
Respect to Ground . . . . . +6V to -0.3V
V_{PP} Supply Voltage with Respect
to Ground . . . . . +28V to -0.3V

*COMMENT: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

2716 PROGRAM CHARACTERISTICS

D.C. Programming Characteristics

$T_A = 25°C \pm 5°C$, V_{CC}[2] = 5V ±5%, V_{PP}[2,3] = 25V ±1V

Symbol	Parameter	Min.	Typ.	Max.	Units	Test Conditions
I_{LI}	Input Current (for Any Input)			10	μA	V_{IN} = 5.25V/0.45
I_{PP1}	V_{PP} Supply Current			5	mA	PD/PGM = V_{IL}
I_{PP2}	V_{PP} Supply Current During Programming Pulse			30	mA	PD/PGM = V_{IH}
I_{CC}	V_{CC} Supply Current			100	mA	
V_{IL}	Input Low Level	-0.1		0.8	V	
V_{IH}	Input High Level	2.2		V_{CC}+1	V	

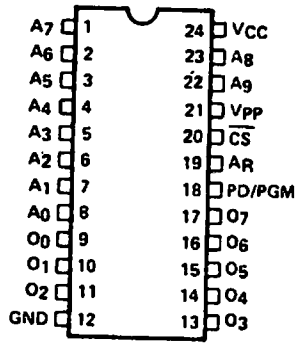
NOTES:

2. V_{CC} must be applied simultaneously or before V_{PP} and removed simultaneously or after V_{PP}. The 2716 must not be inserted into or removed from a board with V_{PP} at 25 ±1V to prevent damage to the device.
3. The maximum allowable voltage which may be applied to the V_{PP} pin during programming is +26V. Care must be taken when switching the V_{PP} supply to prevent overshoot exceeding this 26V maximum specification.

Low Power, +5 V 1 K x 8 UV Eprom 2758

- **Single +5V Power Supply**
- **Simple Programming Requirements**
  Single Location Programming
  Programs With One 50ms Pulse
- **Low Power Dissipation**
  525mW Max Active Power
  132mW Max Standby Power
- **Fast Access Time: 450ns Max In Active and Standby Power Mode**
- **Inputs and Outputs TTL Compatible During Read and Program**
- **Three-State Outputs For or-Ties**

PIN CONFIGURATION



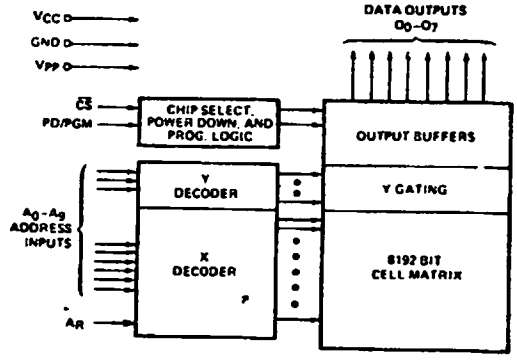
MODE SELECTION

MODE \ PINS	PD/PGM (18)	A_R (19)	$\overline{CS}$ (20)	V_{PP} (21)	V_{CC} (24)	OUTPUTS (9-11, 13-17)
Read	V_{IL}	V_{IL}	V_{IL}	+5	+5	D_{OUT}
Deselect	Don't Care	V_{IL}	V_{IH}	+5	+5	High Z
Power Down	V_{IH}	V_{IL}	Don't Care	+5	+5	High Z
Program	Pulsed V_{IL} to V_{IH}	V_{IL}	V_{IH}	+25	+5	D_{IN}
Program Verify	V_{IL}	V_{IL}	V_{IL}	+25	+5	D_{OUT}
Program Inhibit	V_{IL}	V_{IL}	V_{IH}	+25	+5	High Z

PIN NAMES

A_0–A_9	ADDRESSES
PD/PGM	POWER DOWN/PROGRAM
$\overline{CS}$	CHIP SELECT
O_0–O_7	OUTPUTS
A_R	SELECT REFERENCE INPUT LEVEL

BLOCK DIAGRAM



Absolute Maximum Ratings*

Temperature Under Bias . . . . . -10°C to +80°C
Storage Temperature . . . . . -65°C to +125°C
All Input or Output Voltages with Respect to Ground . . . . . +6V to -0.3V
V_{PP} Supply Voltage with Respect to Ground . . . . . +28V to -0.3V

*COMMENT: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

2758 PROGRAM CHARACTERISTICS

$T_A = 25°C \pm 5°C$, V_{CC}[2] = 5V ±5%, V_{PP}[2,3] = 25V ±1V

D.C. Programming Characteristics

Symbol	Parameter	Min.	Typ.	Max.	Units	Test Conditions
I_{LI}	Input Current (for Any Input)			10	μA	V_{IN} = 5.25V/0.45
I_{PP1}	V_{PP} Supply Current			5	mA	PD/PGM = V_{IL}
I_{PP2}	V_{PP} Supply Current During Programming Pulse			30	mA	PD/PGM = V_{IH}
I_{CC}	V_{CC} Supply Current			100	mA	
V_{IL}	Input Low Level	-0.1		0.8	V	
V_{IH}	Input High Level	2.2		V_{CC}+1	V	
A_R	Select Reference Input Level	-0.1		0.8	V	I_{IN} = 10 μA

NOTES:

2. V_{CC} must be applied simultaneously or before Vpp and removed simultaneously or after Vpp. The 2758 must not be inserted into or removed from a board with Vpp at 25 ±1V to prevent damage to the device.
3. The maximum allowable voltage which may be applied to the Vpp pin during programming is +26V. Care must be taken when switching the Vpp supply to prevent overshoot exceeding this 26V maximum specification.

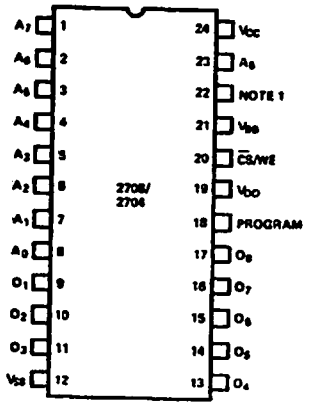
2708, 2704

8K AND 4K UV ERASABLE PROM

- 2708 1024x8 Organization
- 2704 512x8 Organization

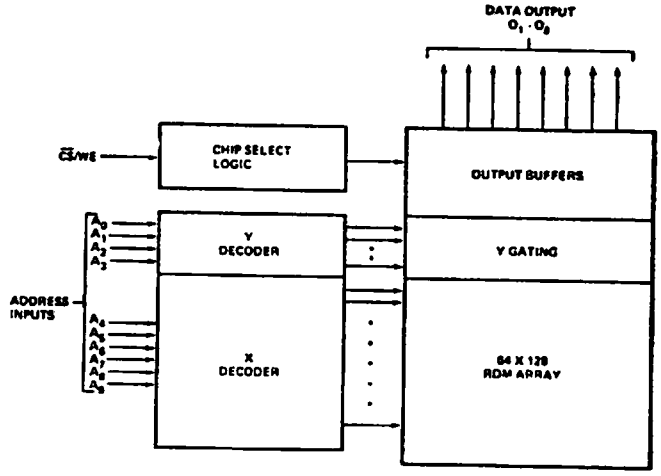
- Fast Programming — Typ. 100 sec. For All 8K Bits
- Low Power During Programming
- Access Time — 450 ns Max.
- Standard Power Supplies — +12V, +5V, -5V
- Static — No Clocks Required
- Inputs and Outputs TTL Compatible During Both Read and Program Modes
- Three-State Output — OR-Tie Capability

PIN CONFIGURATIONS



NOTE 1. 2704: PIN 22 = V_{SS}.
2708: PIN 22 = A_8.

BLOCK DIAGRAM



PIN NAMES

A_0-A_8	ADDRESS INPUTS
O_1-O_8	DATA OUTPUTS
$\overline{CS}$/WE	CHIP SELECT/WRITE ENABLE INPUT

PIN CONNECTION DURING READ OR PROGRAM

MODE	PIN NUMBER						
	9-11, 13-17	12	18	19	20	21	24
READ	D_{OUT}	V_{SS}	V_{SS}	V_{DD}	V_{IL}	V_{BB}	V_{CC}
PROGRAM	D_{IN}	V_{SS}	Pulsed V_{IHP}	V_{DD}	V_{PHW}	V_{BB}	V_{CC}

Absolute Maximum Ratings*

Temperature Under Bias	-25°C to +85°C
Storage Temperature	-65°C to +125°C
V_{DD} With Respect to V_{BB}	+20V to -0.3V
V_{CC} and V_{SS} With Respect to V_{BB}	+15V to -0.3V
All Input or Output Voltages With Respect to V_{BB} During Read	+15V to -0.3V
$\overline{CS}$/WE Input With Respect to V_{BB} During Programming	+20V to -0.3V
Program Input With Respect to V_{BB}	+35V to -0 3V
Power Dissipation	1.5W

*COMMENT

Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

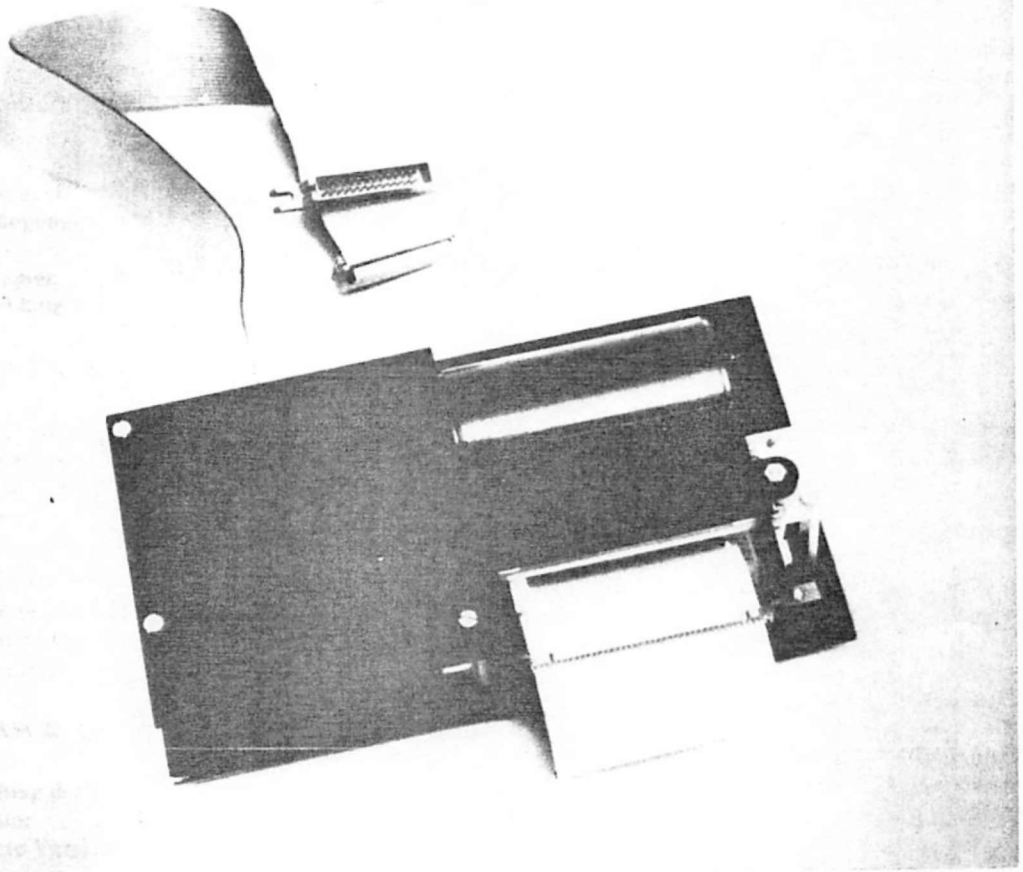
D.C. Programming Characteristics

Symbol	Parameter	Min.	Typ.	Max.	Units	Test Conditions
I_{LI}	Address and $\overline{CS}$/WE Input Sink Current			10	μA	V_{IN} = 5.25V
I_{IPL}	Program Pulse Source Current			3	mA	
I_{IPH}	Program Pulse Sink Current			20	mA	
I_{DD}	V_{DD} Supply Current		50	65	mA	Worst Case Supply Currents
I_{CC}	V_{CC} Supply Current		6	10	mA	All Inputs High
I_{BB}	V_{BB} Supply Current		30	45	mA	$\overline{CS}$/WE = 5V; T_A = 0°C
V_{IL}	Input Low Level (except Program)	V_{SS}		0.65	V	
V_{IH}	Input High Level for all Addresses and Data	3.0		V_{CC}+1	V	
V_{IHW}	$\overline{CS}$/WE Input High Level	11.4		12.6	V	Referenced to V_{SS}
V_{IHP}	Program Pulse High Level	25		27	V	Referenced to V_{SS}
V_{ILP}	Program Pulse Low Level	V_{SS}		1	V	$V_{IHP} - V_{ILP}$ = 25V min.

KONTRON®

KONTRON-KIT/D

KONTRON
ELEKTRONIK GMBH



Z80
MIKROCOMPUTER-SYSTEME

III. Z80-KIT/D

Inhalt: Seite

1. Einleitung

Der Drucker-Zusatz Z80-KIT/D stellt eine preisgünstige Ergänzung des Einfach-Computer- und Lernsystems KONTRON Z80-KIT dar.
Er erlaubt mit der mitgelieferten Software den Ausdruck von Speicherstellen in Hexadezimalform und den Ausdruck von ASCII-Zeichen.
Die Software ist in zwei Unterprogramme (ASCII-, HEX-Ausdruck) aufgeteilt und kann vom Anwender wahlweise aufgerufen werden.
Der Anschluß des Druckers an den Z80-KIT erfolgt über die parallele Schnittstelle (PIO) der ECB/K.

2. Bestandteile des KIT/D

Der Drucker-Zusatz besteht aus

- Grundplatte mit Papierhalterklappe, durch die Papier nachgelegt werden kann.
- KONTRON-Drucker 5020 B mit Papierhalter und Ansteuerungselektronik
- 26-pol. Scotchflex-Flachkabelverbinder (mit Zusatzplatine am Drucker befestigt); 26-pol. Pfosten-Verbinder (3M – Nr. 3399) zum Aufsecken auf den 26-poligen Pfosten-Steckverbinder STB der ECB/K. Zusätzlich aufgequetschter Stiftstecker (3M – Nr. 3328) für den Zugang zum nicht benutzten Port B der PIO.

3. Technische Beschreibung des KONTRON-Druckwerks

3.1. Allgemeines

Es handelt sich um ein seriell arbeitendes Druckwerk. Gedruckt wird mittels eines beweglichen Druckkopfes, der sieben Elektroden trägt. Die Druckrichtung ist dabei je nach Einbaulage von rechts nach links bzw. von links nach rechts. Die Zeichen werden über eine 5 × 7 Punktematrix durch die Steuer-Logik gebildet.
Das Druckprinzip basiert auf dem punktweisen Abbrennen einer dünnen Metallschicht des Papiers. Die darunterliegende schwarze Schicht ergibt den notwendigen Kontrast.
(Siehe 3.9.).

3.2. Zeichengröße

Höhe: ca. 3 mm; festgelegt durch die Anordnung der sieben Elektroden
Breite: Variabel; hängt von der Zahl der in einer Zeile zu druckenden Zeichen ab.
Einstellung bei KIT/D;
Breite: ca. 1,5 mm

3.3. Zeichensatz

Der Zeichensatz hängt vom Zeichengenerator der Kontroll-Logik ab; es wird der aus 64 Zeichen bestehende ASCII-Zeichensatz verwendet. Änderungen des Zeichensatzes sind durch Einsatz anderer Zeichengeneratoren (PROM) möglich.

3.4. Zeichenzahl

Die Zahl der Zeichen pro Zeile wird mit der Zeichenbreite von der Steuer-Logik und von der dazugehörigen Druckersoftware bestimmt.

Beim KIT/D sind maximal 29 Zeichen pro Zeile erlaubt.
Mit dem Potentiometer auf der Steuer-Logik (siehe Lageplan der Lötbrücken; Anhang) ist die Zeichenbreite und damit auch die „Breite des gesamten Ausdrucks veränderbar.

3.5. Zeilenabstand

Der Zeilenabstand beträgt 5 mm.
Er ist durch die Mechanik des Druckwerkes festgelegt.

3.6. Druckgeschwindigkeit

Die Druckgeschwindigkeit beträgt ca. 2 Zeilen pro Sekunde.
Es besteht die Möglichkeit zur Umrüstung auf ca. 1 Zeile pro Sekunde. (Siehe 7.).
Dies kann durch Änderung von zwei Lötbrücken erreicht werden.

3.7. Umgebungsbedingungen

Das Druckwerk arbeitet ohne jegliche Schmiermittel. Dadurch kann es auch unter extremen Bedingungen arbeiten.

3.8. Motor

Das Druckwerk wird von einem Glockenankermotor angetrieben. Die Maximalleistung des Motors ist wesentlich größer als die Betriebsleistung. Dies garantiert eine konstante Laufgeschwindigkeit während des Druckvorgangs, wodurch ein Kopfpositionssignal entfällt und die Zeichenanforderungslogik durch einen frei laufenden Takt gesteuert werden kann.

3.9. Metallisiertes Papier

Das metallisierte Papier besteht aus Zellulose, wobei eine Seite mit einer durchgehenden, gleichmäßigen Aluminiumschicht bedampft ist. Zwischen Papier und Aluminium ist eine Zwischenschicht aus schwarzem Kontrastmaterial aufgebracht.
Physikalische und mechanische Eigenschaften:

- Gewicht: 37 Gramm/Quadratmeter (+/– 8%) gemessen nach Tappi 410 0S – 68 Normen
- Dicke: 41 Mikrometer (+/– 10%) gemessen nach Tappi 0S – 68 Normen
- Zugfestigkeit: in Längsrichtung: 6,5 kg/15 mm +/– 2, gemessen nach Tappi T 404 TS Normen
- Reißfestigkeit: längs und quer: 17 Gramm +/– 5, gemessen nach Tappi T 414 TS – 65 Normen

3.10. Geräuschentwicklung

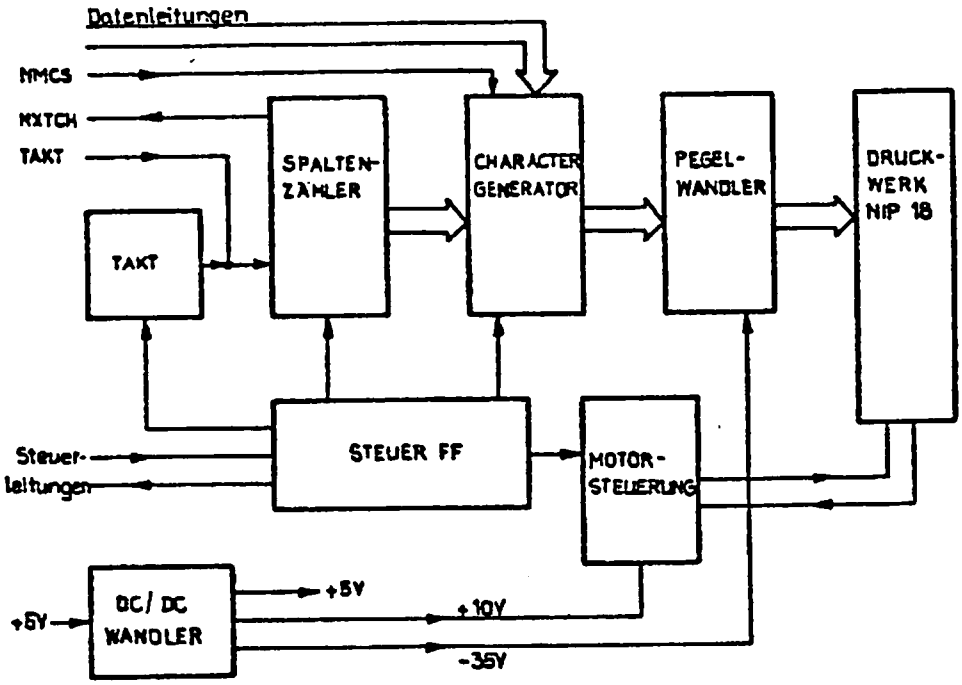
Während des Druckvorganges ist der Geräuschpegel des Druckwerks ohne Gehäuse, gemessen in einem echofreien Raum, weniger als 54 Dezibel.
In der Bereitschaftsstellung ist das Druckwerk absolut geräuschlos.

Zeichensatz

4. Elektronischer Aufbau des Druckers

Die Logik des Druckers ist in CMOS-Technologie aufgebaut.
Als Zeichen-Generator wird ein TTL-PROM verwendet.

4.1. Blockschaltbild KIT/D



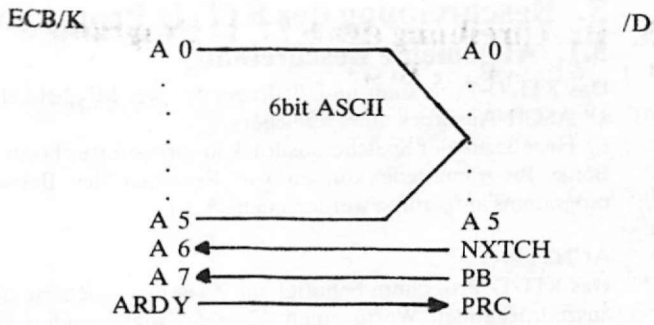
4.2. Anschluß des KIT/D

Der Anschluß des KIT/D an den Z80-KIT erfolgt über PORT A der parallelen Schnittstelle (PIO) der ECB/K.
Die Ansteuerung des Dateneingangs des KIT/D erfolgt über die 6 Datenleitungen A 0 . . . . A 5 des PORT A der PIO auf der ECB/K. (6 Bit ASCII, Bit-parallel, Zeichen-seriell).

Die 3 Steuersignale

- NXTCH (Next Character)
- PB (Printer Busy)
- PRC (Print Command)

laufen ebenfalls über den PORT A (A 6, A 7 und ARDY) der ECB/K-PIO.
Mittels des aufgequetschten Stiftsteckers (3M – Nr. 3328) des Flachbandkabels zwischen ECB/K und KIT/D kann über das verbleibende PORT B der PIO auf der ECB/K frei verfügt werden.



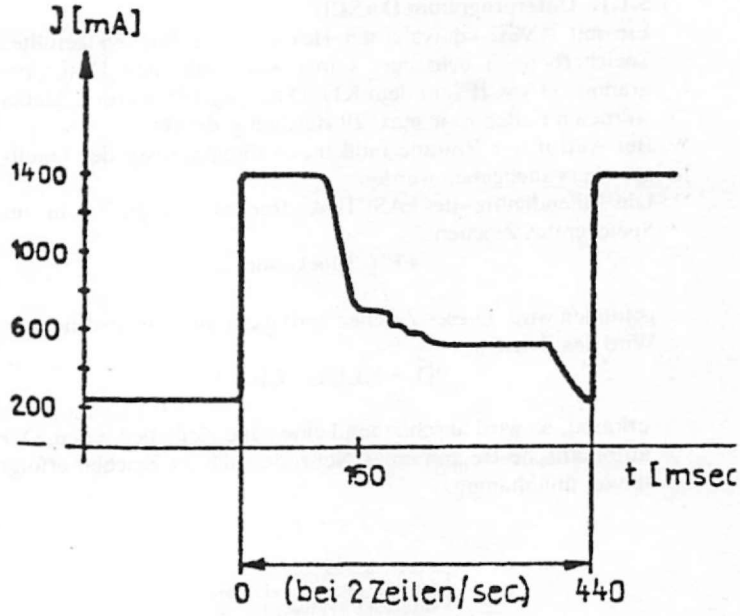
Die gezeigten Verbindungen werden durch das mitgelieferte Flachband hergestellt.

4.3. Verbindungen zwischen ECB/K und KIT/D

ECB/K STB Stift Nr.	Signal		KIT/D 20-pol. Steckverbinder Stift Nr.	Signal	
13	+5V		1	+5V	
26	+5V		2	+5V	
4	A 5		3	A 5	
5	A 4		4	A 4	
6	A 3		6	A 3	Daten
7	A 2	Port A	8	A 2	
8	A 1		10	A 1	
9	A 0		12	A 0	
3	A 6		13	NXTCH	Steuer-
2	A 7		16	PB	leitungen
12	ARDY		9	PRC	
1	GND		19	GND	
4	GND		20	GND	

4.5. Stromaufnahme

Die Stromaufnahme im Ruhezustand beträgt ca. 235 MA.

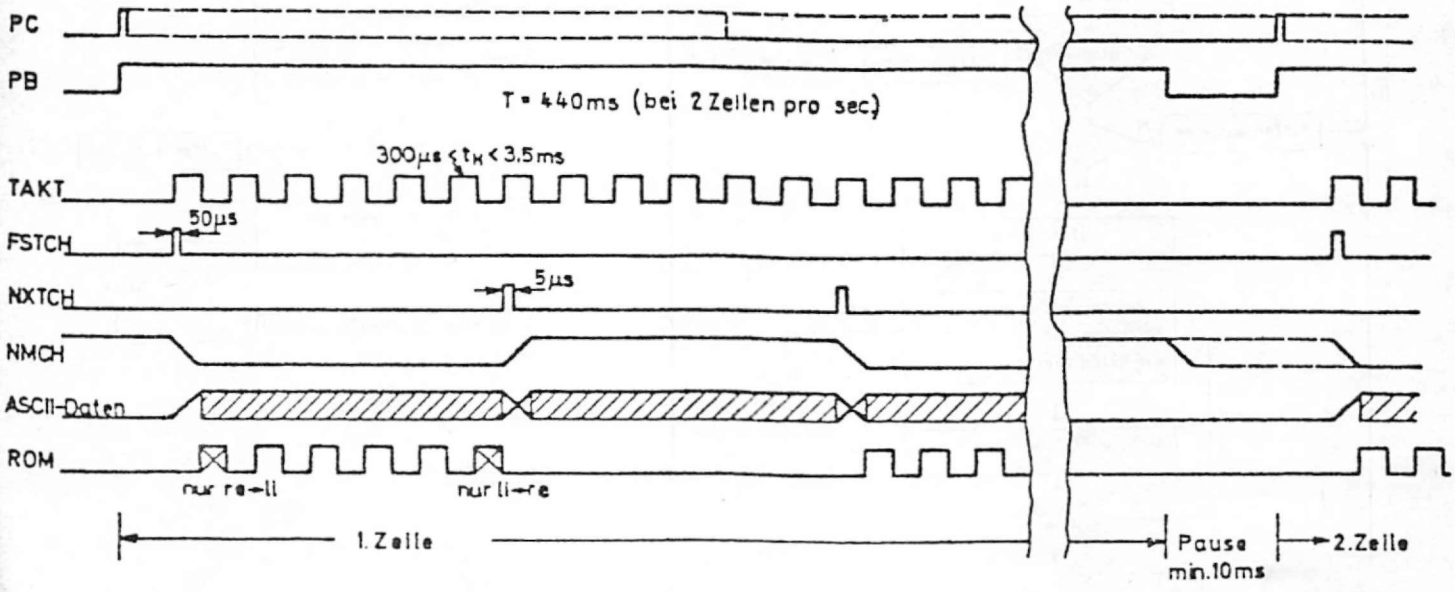


Stromaufnahme für $U_{Betrieb} = 5{,}0$ Volt

4.4 Impulsdiagramm

ASCII-Daten: schraffierter Bereich kennzeichnet Gültigkeit. 500 µs nach FSTCH bzw. NXTCH gültig (bei 2 Zeilen/s, 32 Zeichen/Zeile)

NMCH muß H werden mit NXTCH, das dem letzten zu druckenden Zeichen folgt,
muß L werden mit FSTCH, kann L werden mit PB (bzw. PB)



5. Beschreibung des KIT/D Programms

5.1. Allgemeine Beschreibung

Das KIT/D-Programm umfaßt folgende zwei Möglichkeiten:
a) ASCII-Ausdruck eines Speichers
b) Hexadezimaler Speicherausdruck in formatierter Form
Beide Programmteile können wie Routinen des Betriebsprogramms aufgerufen werden (siehe 5.3.).

ACHTUNG:
Das KIT/D-Programm benötigt zur Zwischenspeicherung der auszudruckenden Werte einen 32 Bytes umfassenden Speicherbereich.
Dieser Bufferbereich liegt bei der Adresse 3CA2 beginnend. Es wird demnach der Speicher im Bereich

3CA2 bis 3CC1

vom KIT/D-Programm belegt.

ACHTUNG:
Das KIT/D-Programm arbeitet mit Interrupts im Interrupt-Mode-2 (IM2).

Auf diese Tatsache ist besonders zu achten, wenn die Drucker-Routinen von einem Anwender-Programm aufgerufen werden, das ebenfalls mit Interrupt arbeitet.

Das Retten des I-Registers der CPU muß vor dem Aufruf im Anwenderprogramm realisiert werden.

Das Laden der I-Register von CPU und PIO (Int.-Vektor) für die Druckerroutinen erfolgt automatisch im Programmteil PIOIN, dem Initialisierungsprogramm, das von den beiden Routinen DASCII und DHEX aufgerufen wird.

Die Interrupttabelle, die die Adresse der Interruptserviceroutine enthält, liegt auf den Adressen

3CA0 und 3CA1

5.1.1. Unterprogramm DASCII

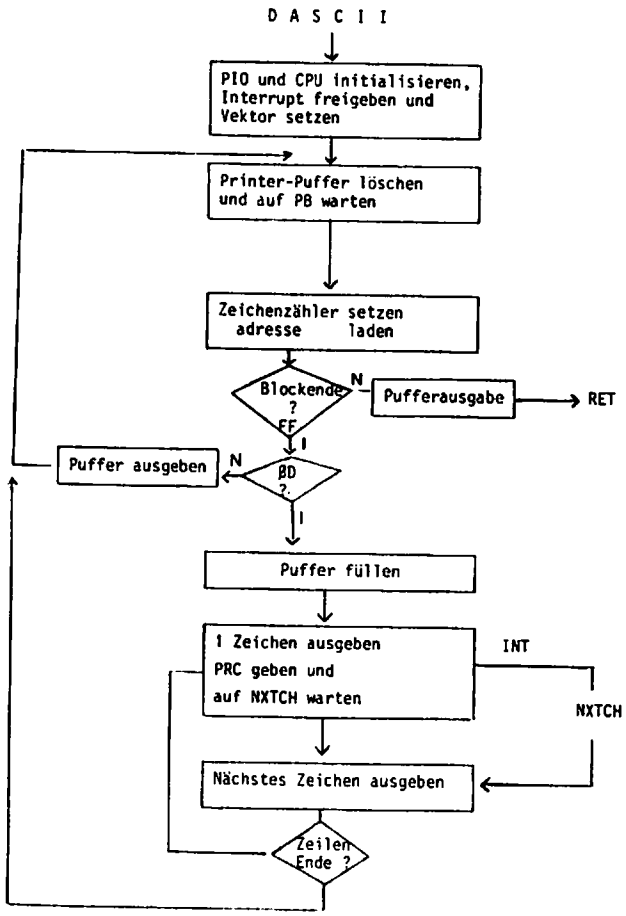
Ein mit ASCII-äquivalenten Hexadezimal-Zeichen gefüllter Speicherbereich beliebiger Länge kann mit dem Unterprogramm „DASCII" auf dem KIT/D ausgegeben werden. Dabei werden n Zeilen zu je max. 29 Zeichen gedruckt.
Bei Aufruf der Routine muß die Anfangsadresse des Quellspeichers angegeben werden.
Die Beendigung des ASCII-Andruckes erfolgt, wenn im Speicher das Zeichen

FF ≙ Blockende

gefunden wird. Dieses Zeichen wird nicht mehr ausgedruckt.
Wird das Zeichen

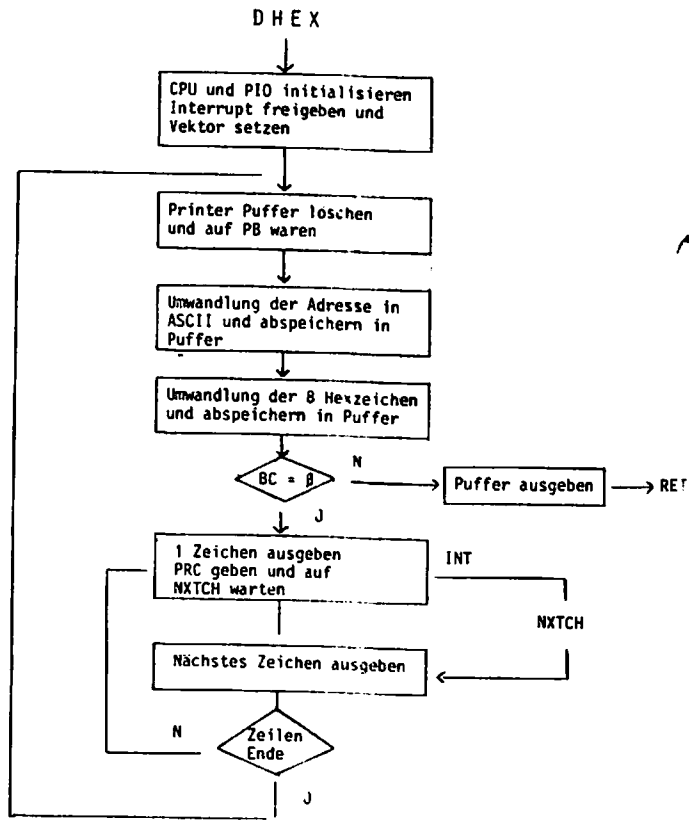
0D ≙ NEUE ZEILE!

erkannt, so wird anschließend eine neue Zeile begonnen. Der automatische Beginn einer Neuzeile nach 29 Zeichen erfolgt davon unabhängig.



5.1.2. Unterprogramm DHEX

Der Ausdruck von Hexadezimal-Zeichen dient zur Ausgabe von Programmen und Speicherstellen und erfolgt in formatierter Form.
Dabei wird in folgendem Format ausgegeben:
Als erstes wird die 16-Bit Adresse des ersten Speicherplatzes ausgegeben.
Danach folgen zwei Leerzeichen und darauf der Ausdruck von max. 8 Speicherstellen pro Zeile, jeweils durch Leerzeichen getrennt.
Bei jeder neuen Zeile wird wieder mit dem Ausdruck der aktuellen Adresse begonnen.
Bei Aufruf der Routine müssen die Anfangsadresse des Quellspeichers und die Anzahl der zu druckenden Bytes angegeben werden.
Die Beendigung des Ausdruckes erfolgt automatisch nach dem letzten Zeichen.



5.2. Verwendbare ASCII-Zeichen

Der vom Drucker erzeugbare Zeichensatz ist abhängig vom verwendeten Zeichengenerator.
Grundsätzlich ist eine Darstellung aller 7Bit-ASCII-Zeichen möglich. Um die Möglichkeiten des Z80-KIT möglichst weitgehend auszunutzen, wurde die Benutzung nur eines einzigen Ports angestrebt.

Daraus ergibt sich für den Drucker die Beschränkung des ASCII-Zeichenvorrats auf sechs Datenbits.

Darüber hinaus war jedoch die Kombination CR (Carriage Return) für das Anfordern einer Neuzeile (ASCII: 0D) wünschenswert. Dieses Problem wurde softwaremäßig gelöst. Wird 0D (≙ CR) von der Software erkannt, sorgt diese dafür, daß der Rest der aktuellen Druckzeile (der Drucker kennt kein Neuzeile-Kommando!) mit Leerzeichen (im Zeichensatz des Druckers enthalten!) aufgefüllt wird.

Als weiteres Zusatzzeichen wurde FF aufgenommen, bei dessen Auftreten im Programmteil „DASCII" ein Abbruch des Ausdruckes erreicht wird.

5.3. Zusammenstellung der KIT/D-Routinen

5.3.1. Cassetten-Version

ASCII druckt den Inhalt eines Speicherbereiches in Form von ASCII-Zeichen aus Startadresse des Quellspeichers beim Aufruf im DE Registerpaar.
Benützte Register: A B C D E F H L (IM2!)
Benützer Speicher: 3CA0 . . . . 3CBF
Einsprungpunkt: 6020

HEX druckt den Inhalt eines Speicherbereiches in hexadezimaler Form aus. Angabe der Adresse am Zeilenanfang.
Startadresse des Quellspeichers beim Aufruf im DE Registerpaar, Anzahl der Bytes im BC Registerpaar.
Benützte Register: A B C D E F H L I (IM2!)
Benützer Speicher: 3CA0 . . . . 3CBF
Einsprungpunkt: 608A

5.3.2. PROM-residente Version (siehe KIT/PDT)

ASCII druckt den Inhalt eines Speicherbereiches in Form von ASCII-Zeichen aus.
Startadresse des Quellspeichers beim Aufruf im DE Registerpaar.
Benützte Register: A B C D E F H L I (IM2!)
Benützte Speicher: 3CA0 . . . . 3CBF
Einsprungpunkt: 0E7F

HEX druckt den Inhalt eines Speicherbereichs im hexadezimaler Form aus. Angabe der Adresse am Zeilenanfang.
Startadresse des Quellspeichers beim Aufruf im DE Registerpaar, Anzahl der Bytes im BC Registerpaar.
Benützte Register: A B C D E F H L I (IM2!)
Benützter Speicher: 3CA0 . . . . 3CBF
Einsprungpunkt: 0EE9

5.4. Anwendungsbeispiele

BEISPIEL 1:
Es sollen die ersten 100 (HEX) Byte des Monitorprogramms ausgedruckt werden.
Dazu erstellt man z.B. ein, aus zwei Befehlen bestehendes Programm. Der erste Befehl ist der Aufruf (CALL) der Printer-Routine HEX, während der zweite Befehl den Rücksprung ins Betriebsprogramm darstellt. Die Angabe von Startadresse (DE) bzw. Länge (BC) des Speicherbereichs erfolgt durch direktes Setzen der Register. Das Zwei-Befehle-Programm lautet:

```
CALL HEX      CD E9 0E

JP MONITOR    C3 0B 00    (KIT/M2)
              C3 1D 00    (KIT/M1)
```

Dieses Programm wird im RAM bei 3C50 abgespeichert. Anschließend werden die Register

D, E und C auf 00
und
B auf 01

gesetzt.
Setzt man jetzt den Befehlszähler (PC) auf 3C50 und drückt die Taste START, wird der gewünschte Speicherbereich ausgedrückt. Während des Druckens erlöschen die Anzeigen des Z80-KIT.

BEISPIEL 2:
Ausdruck eines Textes.
Der gewünschte Text muß vor dem Drucken in Form von ASCII-Zeichen eingegeben werden. Die Anfangsadresse sei 3C60.
Anschließend erfolgt wie in Beispiel 1 der Aufruf der Printer-Routine mit dem Rücksprung ins Betriebsprogramm nach Abschluß des Druckvorganges. Die Eingabe der Parameter erfolgt zusätzlich per Programm.
Der einzugebende Text lautet im ASCII-Format:

```
48 45 52 6A 4C 49 43 48 45 4E

20 47 4C 55 45 43 4B 57 55

4E 53 43 48 20 5A 55 4D 4D

4B 4F 4E 54 52 4F 4E 20 4B 49 54

2F 44 FF
```

Geben Sie diese Kombinationen beginnend bei 3C60 in den Speicher ein!
Anschließend schreiben Sie beginnend bei 3C50 die Parametereingabe, den Aufruf und den Rücksprung ins Betriebsprogramm.

```
LD DE, 3C60H   11 60 3C

CALL ASCII     CD 7F0E

JP MONITOR     C3 0B 00    (KIT/M2)
               C3 1D 00    (KIT/M1)
```

Setzen Sie jetzt den PC auf 3C50 und drücken Sie die Taste START.
Sie können anschließend den vorher eingegebenen Text lesen.

6. Laden des KIT/D-Programmes

(nur bei Cassettenversion!)

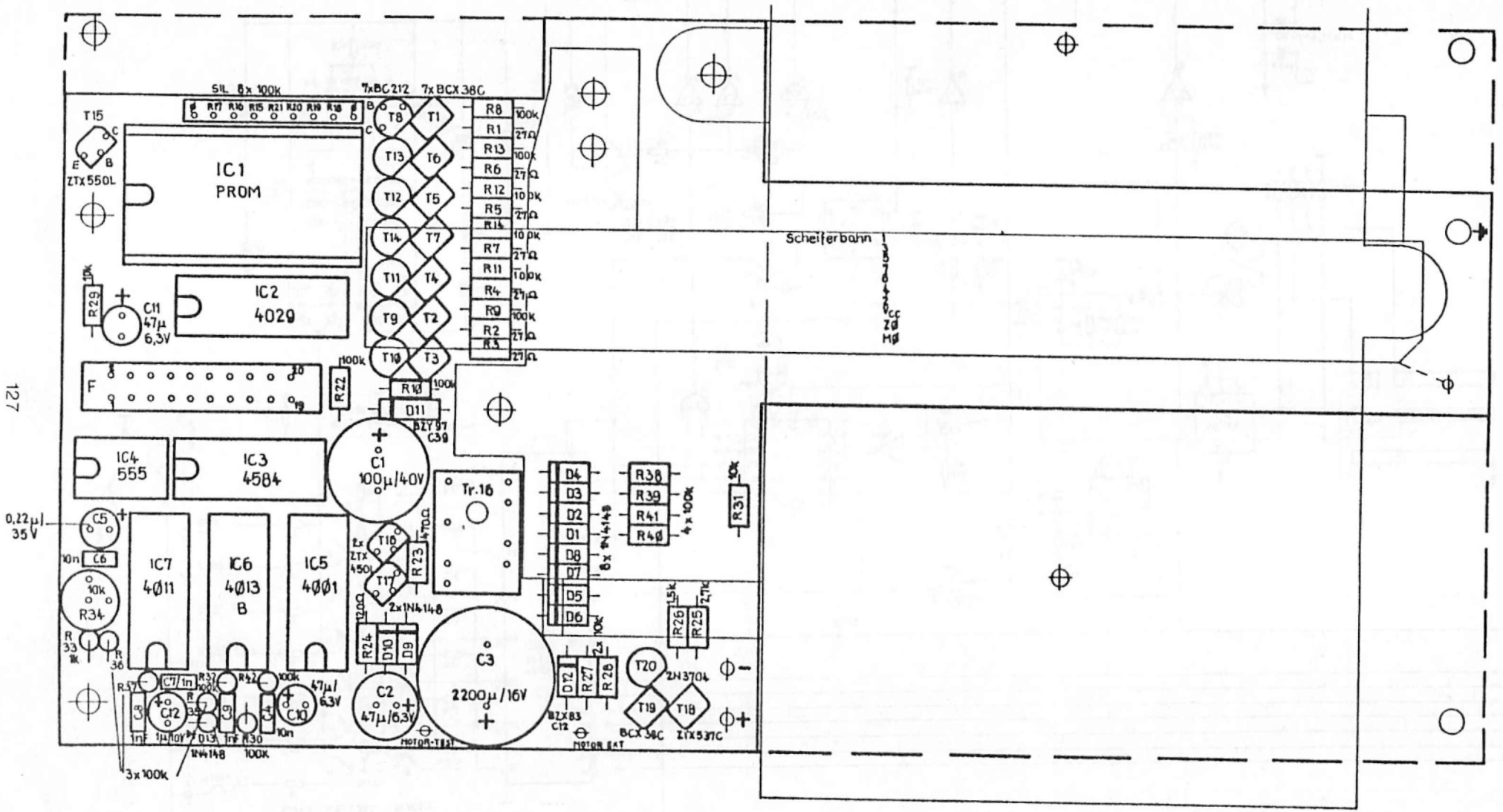
Das KIT/D-Programm wird standardmäßig auf Cassette geliefert und muß vor dem Einsatz in dem RAM-Bereich geladen werden. Siehe dazu LOAD-Vorgang im KIT Anwenderhandbuch (Teil I).
Da das Programm auf die Anfangsadresse

6000H

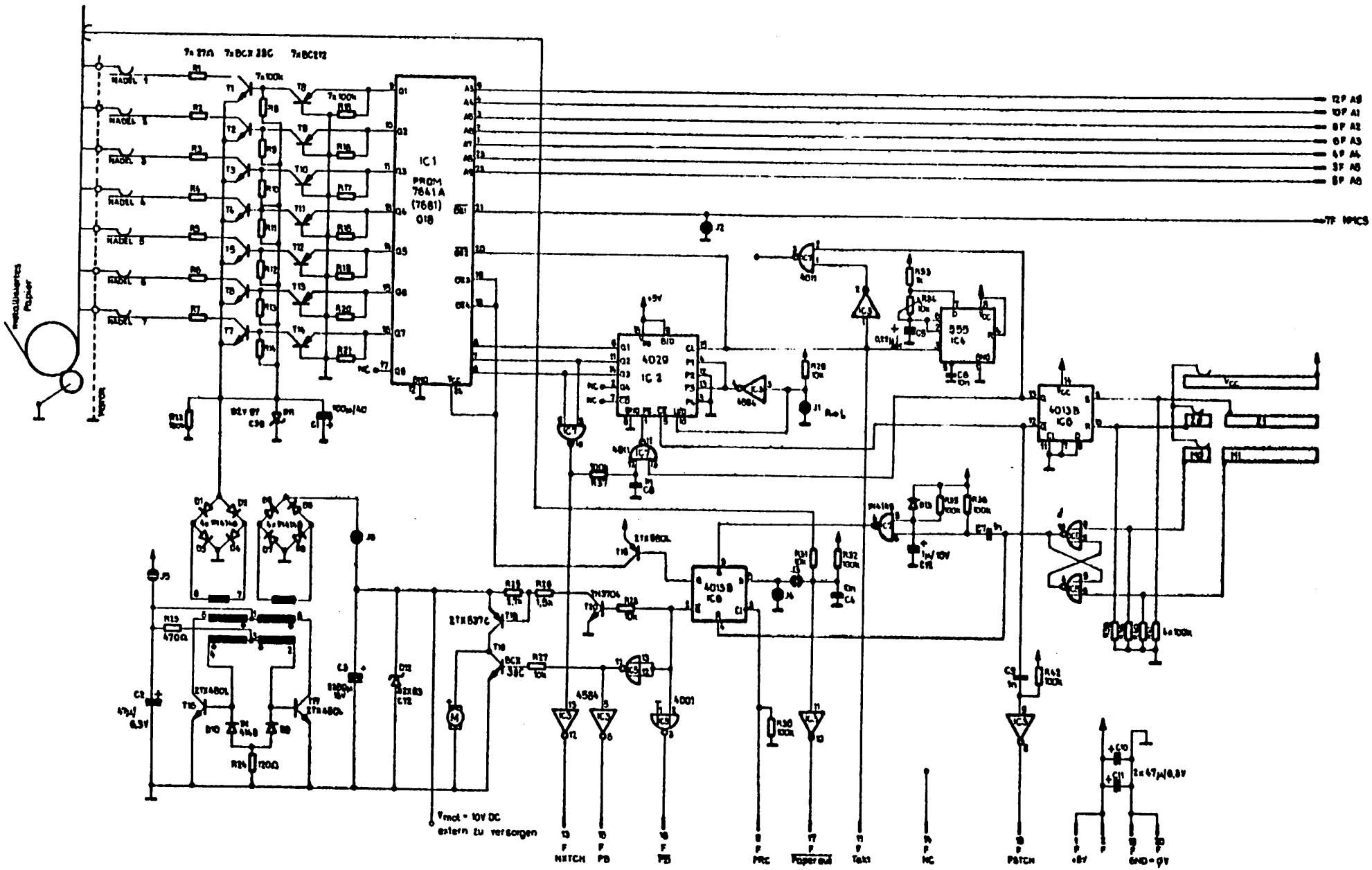
assembliert ist und eine Länge von 10CH Bytes aufweist, muß der zur Verfügung stehende Speicher diesen Bereich überdecken.
Die Port-Adressen sind, da PORT A der PIO auf der ECB/K verwendet wird, mit

08 H DATA
0A H CONTROL

Die Einsprungpunkte sind in Kap. 5.3.1. angegeben.
Falls die PROM-residente KIT/D Software (siehe KIT/PDT, TEil V) verwendet wird, entfällt der Ladevorgang.
Die Einsprungpunkte dieser Version sind in Kap. 5.3.2. angegeben.

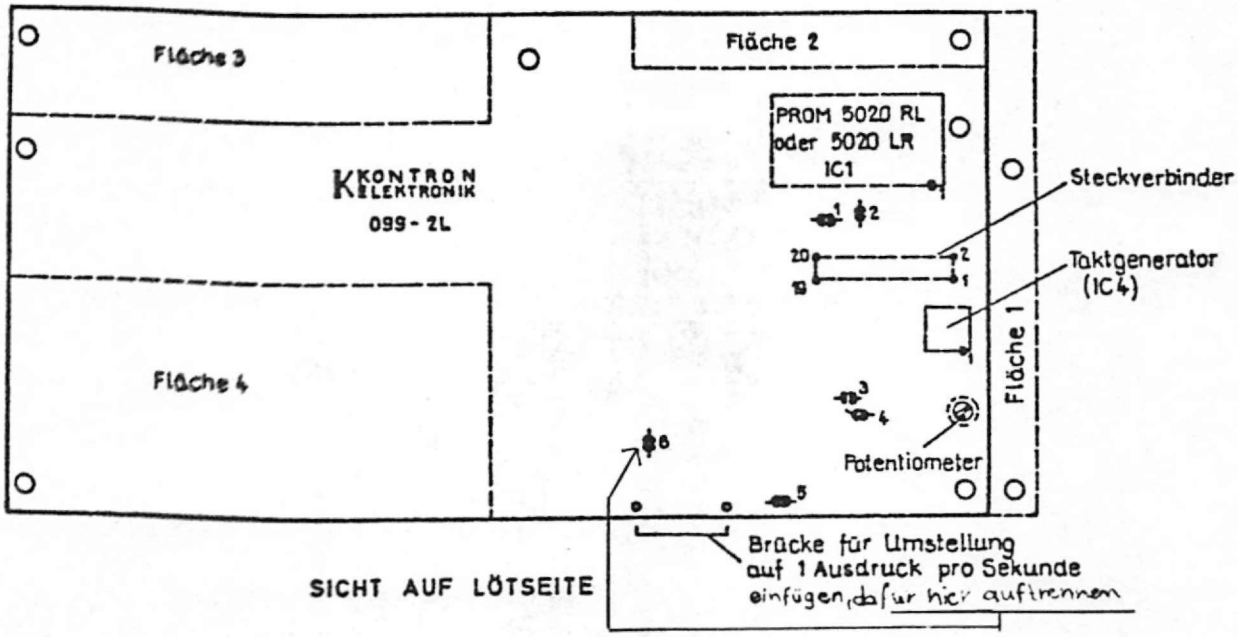


Bestückungsplan



Stromlaufplan

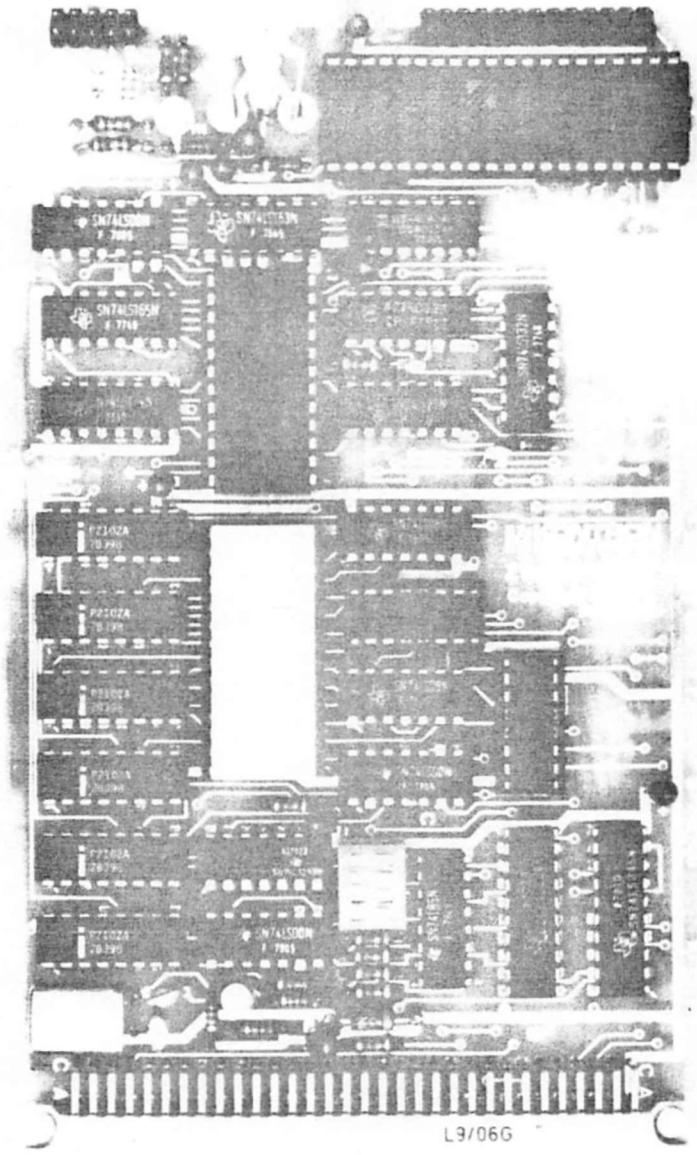
Lageplan für Lotbrücke und Potentiometer



KONTRON-KIT®

KONTRON-KIT/V
KONTRON-KIT/VZ

KONTRON
ELEKTRONIK GMBH



Z80
MIKROCOMPUTER-SYSTEME

IV. Z80-KIT/V Z80-KIT/VZ

Inhalt: Seite

1. Gegenüberstellung von Z80-KIT/V und Z80-KIT/VZ

Bei den Baugruppen Z80-KIT/V und Z80-KIT/VZ handelt es sich in beiden Fällen um Baugruppen zur Bildschirmansteuerung.

Unterschiede bestehen in folgenden 3 Punkten:

- Z80-KIT/VZ wird zusammengebaut und getestet geliefert. und kann direkt auf jeden ECB- oder KIT-Bus aufgesteckt werden. Z80-KIT/V ist ein Bausatz, der als besonders preisgünstige Erweiterung zum Z80-KIT zu verstehen ist.

- Während die Baugruppe Z80-KIT/V über die parallele Schnittstelle des, im Z80-KIT befindlichen Einplatinencomputers (ECB/K bzw. ECB/C) angesteuert wird, besitzt die Baugruppe Z80-KIT/VZ eine zusätzliche Parallelschnittstelle.

- Die Einheit Z80-KIT/VZ besitzt einen eigenen Busanschluß, die Baugruppe Z80-KIT/V wird an die parallele Schnittstelle der ECB/K bzw. ECB/C angeschlossen.

2. Schaltungsbeschreibung

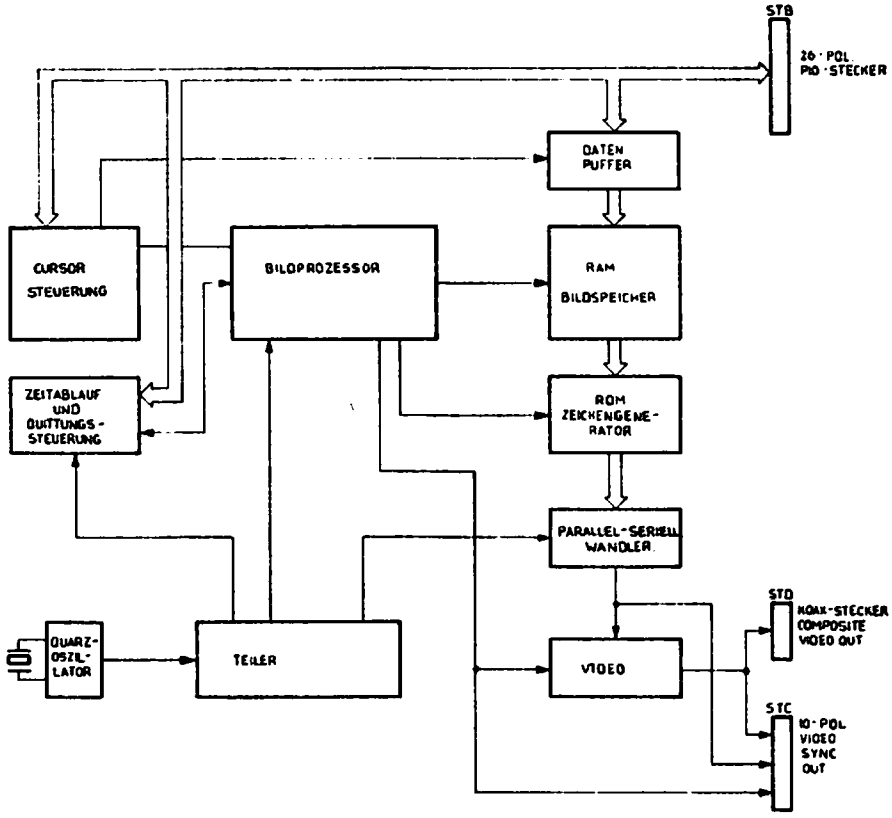
2.1. Schaltungsbeschreibung Z80-KIT/V

Die Baugruppe umfaßt folgende Funktionseinheiten:

Bildprozessor (Video-Controller)
1 K RAM-Bildspeicher
Zeichengenerator
Parallel-Serienwandler
Video-Mischer und Ausgangsverstärker
Anschlußseitig 26-Pol Pio-Stecker, 10 pol. Videostecker für getrennte Video- und Synchronisationssignale, Koaxstecker für Composite-Video-Out Signale.

Aus dem Blockschaltbild ist die Verknüpfung der einzelnen Funktionseinheiten zu erkennen.
Sämtliche I/O Leitungen des PIO's sind u.a. an den PIO-Stecker STB geführt. Die Pinbelegung ist die gleiche wie bei den ECB/C und ECB/K-Baugruppen.
Der Anschluß des Z80-KIT/V erfolgt über eine 26-polige Flachkabelstechverbindung mit dem I/O Port der ECB/C bzw. ECB/K Baugruppe.

Blockschaltbild Z80-KIT/V



2.2. Schaltungsbeschreibung Z80-KIT/VZ

Die Baugruppe umfaßt folgende Funktionseinheiten:

Zwei Parallelschnittstellen (1 Stück Z80-PIO)
Decoder
Bus-Pufferung
Bildprozessor (Video-Controller)
1 K RAM-Bildspeicher
Zeichengenerator
Parallel-Serienwandler
Video-Mischer und Ausgangsverstärker
Anschlußseitig 26-pol. PIO-Stecker, 10-pol. Videostecker für getrennte Video- und Synchronisationssignale, Koaxstecker für Composite-Video-Out Signale.
Busseitig 64-pol. Stecker nach DIN 41612 (VG 95324)

Aus dem Blockschaltbild ist die Verknüpfung der einzelnen Funktionseinheiten zu erkennen.
Die Busleitungen sind gepuffert.

Die Adresse des Z80-PIO kann über DIL-Schalter S1 festgelegt werden.
Es stehen 16 Adressen zur Verfügung. Die Schalterstellung ist invertiert zu betrachten.

ADRESSE	SCHALTER S1/4	S1/3	S1/2
F X H	AUS	AUS	AUS
E X H	AUS	AUS	AUS
USW.			
I X H	EIN	EIN	EIN
0 X H	EIN	EIN	EIN

Durch direkte Auswahl der I/O-Bausteine auf de
oder ECB/K Karte müssen die Adreßbits 2 und 3
LOG 1 sein. Bei einer Schalterstellung

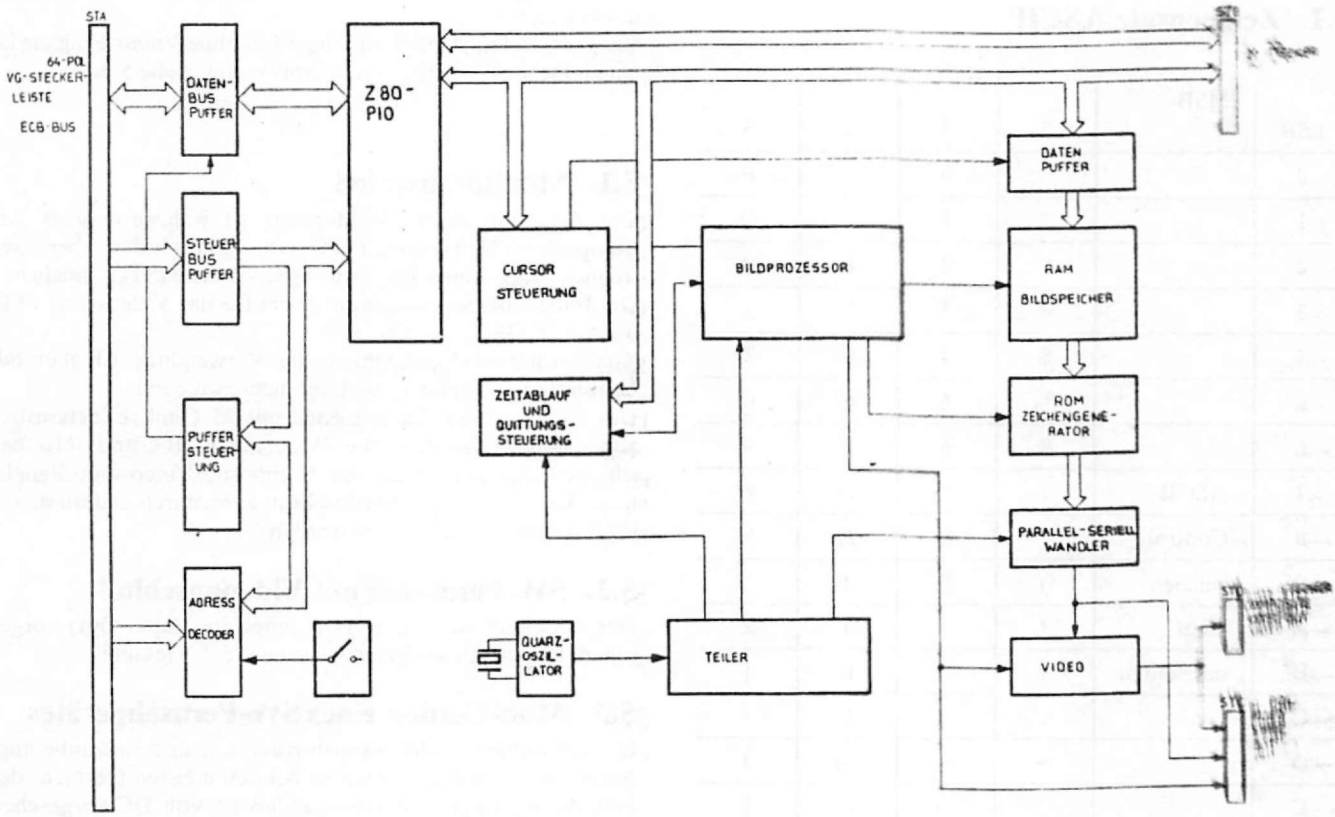
Zum Beispiel: S1/4 AUS S1/3 EIN S1/2 AUS

Am Schalter S1 muß der PIO wie folgt adressiert
I/O Adresse (hex)

ACH = PIO PORT A DATA
ADH = PIO PORT B DATA
AEH = PIO PORT A CONTROL
AFH = PIO PORT B CONTROL

Sämtliche I/O-Leitungen des PIO's sind an den P
STB geführt. Die Pinbelegung ist die gleiche w
ECB/C und ECB/K-Baugruppen.

Blockschaltbild Z80-KIT/VZ



3. Ansteuerung der Z80-KIT/V bzw. Z80-KIT/VZ Baugruppe

Maximal werden von der Bildschirmansteuerung 120 Zeichen pro Sekunde verarbeitet (entspricht 8,3 Ms/Zeichen). Das Löschen des Bildschirmes benötigt ca. 132 Ms.
Eine entsprechende Zeitablaufsteuerung erfolgt automatisch über die Handshakeleitungen BRDY und BSTB des Z80-PIO in der Betriebsart Output.
(Mode 0, siehe Z80-PIO-Manual)

Arbeitsweise:
Bei Ausgabe eines Zeichens geht die BRDY-Leitung auf LOG1.
Nach Abarbeitung (8,3 Ms, bzw. 132 Ms) durch das Videoboard wird BSTB aktiv und löst einen Interrupt aus. Bei entsprechender softwaremäßiger Bearbeitung kann somit die maximale Datenausgabegeschwindigkeit der Z80-KIT/V bzw. Z80-KIT/VZ Baugruppe genutzt werden.

4. Darstellungsweise

Die Zeichendarstellung erfolgt in 16 Zeilen zu 64 Zeichen. Der Zeichenumfang beträgt 64 Zeichen (ASCII-UPPER CASE), die Darstellung erfolgt in einer 5 × 7 Punktmatrix.
Die augenblickliche Schreibposition wird durch den Cursor angezeigt. Durch Umstecken des Jumpers J1 kann zwischen Darstellung schwarze Zeichen auf weißem Grund oder Invers gewählt werden.

4.1. Zeichensatz ASCII

LSB	MSB 0 1	2	3	4	5
0			0	C	P
1		!	1	A	Q
2		"	2	B	R
3		≠	3	C	S
4		$	4	D	T
5		%	5	E	U
6		&	6	F	V
7	ASCII-	/	7	G	W
8	Control-	(	8	H	X
9	zeichen	)	9	I	Y
A	nicht	*	:	J	Z
B	darstellbar	+	;	K	[
C		,	<	L	/
D		-	=	M	J
E		.	>	N	↑
F		/	?	O	-

4.2. Bildschirm- und Cursorsteuerung

ASCII	KEY	HEX	Bedeutung
BS	CNTRL H	08H	Cursor links
HT	CNTRL I	09H	Cursor rechts
LF	CNTRL J	0AG	Cursor nach unten
VT	CNTRL K	0BH	Cursor nach oben
FF	CNTRL L	0CH	Bild Löschen und Cursor zum Bildanfang
CR	CNTRL M	0DH	Zeilenende löschen und Cursor zum Zeilenanfang
SUB	CONTR Z	1AH	Löschen der laufenden Zeile
ESC	SHIFT+CONTR K	1BH	Zeilenvorschub (Bild 1 Zeile nach O. rollen)
FS	SHIFT+CONTR L	1CH	Cursor zum Bildanfang
GS	SHIFT+CONTR M	1DH	Cursor zum Zeilenanfang

Bermerkung:
Mit Erreichen der untersten Bildzeile erfolgt automatisch Übergabe in den ROLL-MODE.

5. Video-Schnittstelle

Die Z80-KIT/V bzw. Z80-KIT/VZ-Baugruppe ist für den Anschluß eines Schwarz-weiß-Monitors oder eines handelsüblichen SW-Fernsehers mit Video-Eingang ausgelegt. Dadurch erreicht man wesentlich höhere Bildqualität und Störsicherheit gegenüber der Verwendung des Antenneneingangs des Fernsehers.
Bei Verwendung eines Fernsehgerätes ohne Video-Eingang ist ein einfacher Eingriff im Gerät notwendig (siehe 5.3.)

5.1. Monitoranschluß

Der Anschluß eines SW-Monitors ist wahlweise über das Composite-Video-Signal (STC oder STD) oder über Getrennte Video- und Composite-Sync.-Signale (STC) möglich.
Die Impedanz des Ausgangstreibers für das Videosignal (T1) beträgt 75 Ohm +/- 5%.
Koaxleitung wird empfohlen, die Verwendung ist aber bei Leitungslängen unter 1,80 Meter nicht zwingend.
Das Kabel soll am Leitungsende mit 75 Ohm reflektionsfrei abgeschlossen werden. Die Widerstände R7 und R16 bestimmen die Amplitude des Composite-Video-Out-Signals. Eine Anpassung an den Sichtschirm kann durch ändern dieser Widerstände vorgenommen werden.

5.2. SW-Fernseher mit Videoanschluß

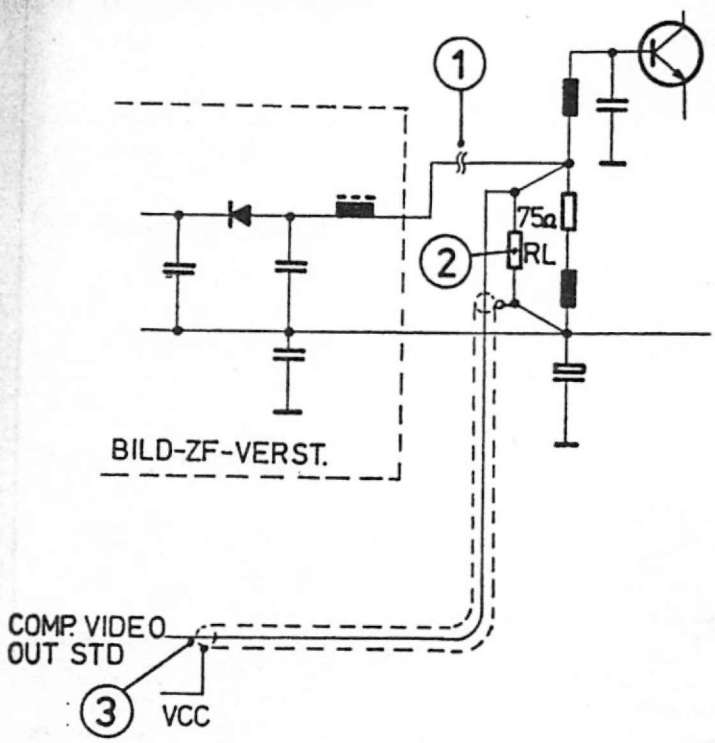
Der Anschluß ist über STD (Composite-Video-Out) vorgesehen. Hierzu gilt sinngemäß das unter 5.1. Gesagte.

5.3. Modifikation eines SW-Fernsehgerätes

Das Fernsehgerät sollte transistorisiert sein und muß unbedingt Netztrennung besitzen. Dies ist bei den meisten Geräten der Fall, die für einen Batterienanschluß 12 Volt DC vorgesehen sind.
Die Modifikation ist in Bild 2 beschrieben.

Zu Punkt 3:
Je nach verwendetem Fernsehgerät kann es notwendig sein, die Amplitude des Composit-Video-Signals zu vergrößern. Dies ist durch ändern der Widerstände R7, R16 möglich. (Z.B. mit R7 = 680 Ohm, R16 = 10 Ohm vout = ca. 2,5 V_{SS}).

Bild 2: Modifikation des SW-Fernsehers



1. Ausgangsleitung vom Bild-ZF-Verstärker auftrennen
2. Koaxkabel anschließen, Abschlußwiderstand 75 Ohm
3. Z80-KIT/VZ
   Jumper J3 öffnen
   Jumper J2 schließen
   Kondensator C5 kurzschließen

Warnung:

Eingriffe in das Fernsehgerät sollten nur bei gezogenem Netzstecker vorgenommen, und von technisch qualifizierten Personen durchgeführt werden.

Bitte beachten Sie, daß weder Hersteller noch das den KIT/V bzw. den KIT/VZ vertreibende Unternehmen für irgendwelche Schäden haften, die mittelbar oder unmittelbar an Personen oder Sachen durch Einsatz dieses Produktes entstehen!

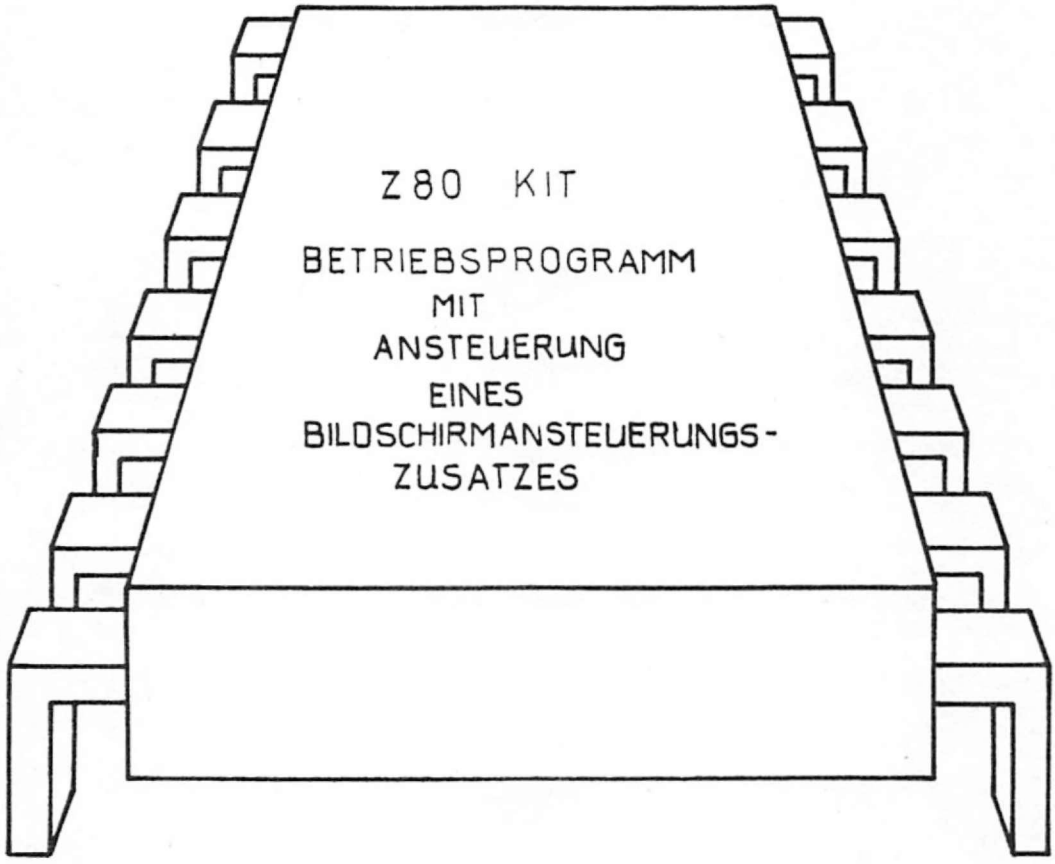
6. Betrieb mit dem Betriebsprogramm Z80-KIT/TV

Beide Baugruppen (Z80-KIT/V und Z80-KIT/VZ) eignen sich im Zusammenhang mit dem TV Betriebsprogramm Z80-KIT/TV zum Aufbau einer Bildschirmanzeige für das Z80-KIT. Für weitere Informationen siehe Teil VI Z80-KIT/TV.

KONTRON KIT®

KONTRON-KIT/PDT

KONTRON
ELEKTRONIK GMBH



Z80
MIKROCOMPUTER-SYSTEME

Inhalt Seite

1. Einleitung

Beim Z80-KIT/PDT handelt es sich um die PROM-residente Software zu den Erweiterungspaketen KIT/P und KIT/D. Zusätzlich sind Treiberroutinen zur Ansteuerung der TTY-Schnittstelle auf der ECB/K sowie Standardarithmetikroutinen enthalten.
Durch die PROM-residente Version der Programme entfällt die eventuelle Notwendigkeit des Einlesevorgangs von der Cassette. Gleichzeitig kann der sonst vom entsprechenden Programm belegte Speicherbereich anderweitig benützt werden.

2. Einsatz des Z80-KIT/PDT

Der Z80-KIT/PDT wird in einem 2k Byte Festwertspeicher (= „Firmware") geliefert.
Der Festwertspeicher kann ohne Hardwareänderung auf Steckplatz A3 der ECB/K untergebracht werden (Adreßbereich 0800H bis 0FFFH). Ein Einsatz auf einer anderen, entsprechend ausgerüsteten Platine (z.B. ECB/E16) ist möglich, sofern der Adreßbereich 0800H bis 0FFFH dort eingestellt wird.

NB.:
Das Betreiben des Z80-KIT/PDT ist nur im Zusammenhang mit den Betriebsprogrammen Z80-KIT/M2 (1 k Byte) oder Z80-KIT/TV (2 k Byte) möglich, da das Programm Z80-KIT/PDT auf Routinen dieser Betriebsprogramme zurückgreift.

3. Aufruf der Programme

3.1. Aufruf des Promprogrammers

Der Aufruf erfolgt wie bei der Cassettenversion. Dabei wird gemäß dem Starten eines Anwenderprogramms der Programmzähler auf die Startadresse gesetzt und anschließend die Taste START gedrückt. (Siehe auch KIT/P, Teil II)

Einsprungadresse: 0800

3.2. Aufruf der Drucker-Routinen

Der Aufruf erfolgt wie bei der Cassetten-Version. Dabei werden die beiden Programmteile über CALL-Befehle in ein Anwenderprogramm eingebunden.
(Siehe auch KIT/D, Teil III)

ASCII druckt den Inhalt eines Speicherbereiches in Form von ASCII-Zeichen aus.
Startadresse des Quellspeichers beim Aufruf im DE Registerpaar.
Benützte Register: A B C D E F H L I (IM2!)
Benützter Speicher: 3CA0 . . . . 3CC3
Einsprungspunkt: 0E7F

HEX druckt den Inhalt eines Speicherbereichs in hexadezimaler Form aus. Angabe der Adresse am Zeilenanfang.
Startadresse des Quellspeichers beim Aufruf im DE Registerpaar. Anzahl der Bytes im BC Registerpaar.
Benützte Register: A B C D E F H L I (IM2!)
Benützter Speicher: 3CA0 . . . . 3CC3
Einsprungpunkt: 0EE9

3.3. Aufruf der TTY Routinen

Der Aufruf erfolgt über CALL-Befehle aus dem Anwenderprogramm.

USART INITIALISIERUNG
Bringt den auf der ECB/K sitzenden seriellen Schnittstellenbaustein in einen definierten Zustand.
Keine Übergabeparameter
Einsprungpunkt: 0F6B

TTYIN
Wartet auf einen Tastendruck von der angeschlossenen ASCII-Tastatur ASCII-Zeichen der entsprechenden Taste beim Verlasssen der Routine im A-Register.
Benützte Register: A, F
Einsprungpunkt: 0F81

TTYOUT
Gibt einen Charakter an das angeschlossene Device aus. Auszugebendes ASCII-Zeichen beim Einsprung in die Routine im A-Register.
Benützte Register: A, F
Einsprungpunkt: 0F8C

3.4. Aufruf der Standardarithmetik

Es sind folgende drei Routinen verfügbar:
16 × 16-BIT Integer Multiplikation
16/ 8-BIT Integer Division
32/ 16-BIT Integer Division
Der Aufruf erfolgt über CALL-Befehle aus einem Anwenderprogramm.

N.B.: Die Ergebnisse der drei Routinen stellen binäre Werte dar. Sind für eine Darstellung dezimale Werte im BCD-Format notwendig, ist eine entsprechende Umrechnung vorzunehmen!

3.4.1. 16 × 16-BIT Integer Multiplikation

Beim Einsprung in die Routine Multiplikand im DE-Registerpaar, Multiplikator im HL-Registerpaar.
Beim Verlassen der Routine Ergebnis in den Registerpaaren HL (höherwertiger Teil) und DE (niederwertiger Teil)
Benützte Register: A, B, C, D, E, H, L, D', E', H', L'
Einsprungpunkt: 0F97

3.4.2. 16/8-BIT Integer Division

Beim Einsprung in die Routine Dividend im HL-Registerpaar, Divisor im A-Register.
Beim Verlassen der Routine Ergebnis im HL-Registerpaar, Rest im DE-Registerpaar.
Benützte Register: A, B, C, D, E, H, L, B'
Einsprungpunkt: 0FBE

3.4.3. 32/16-BIT Integer Division

Beim Einsprung in die Routine Dividend in den Registerpaaren HL (höherwertiger Teil) und DE (niederwertiger Teil).
Beim Verlassen der Routine Ergebnis ebenfalls in den Registerpaaren HL (höherwertiger Teil) und DE (niederwertiger Teil).

Benützte Register: A, B, C, D, E, H, L, B', C', H', L'
Einsprungpunkt: 0FDD

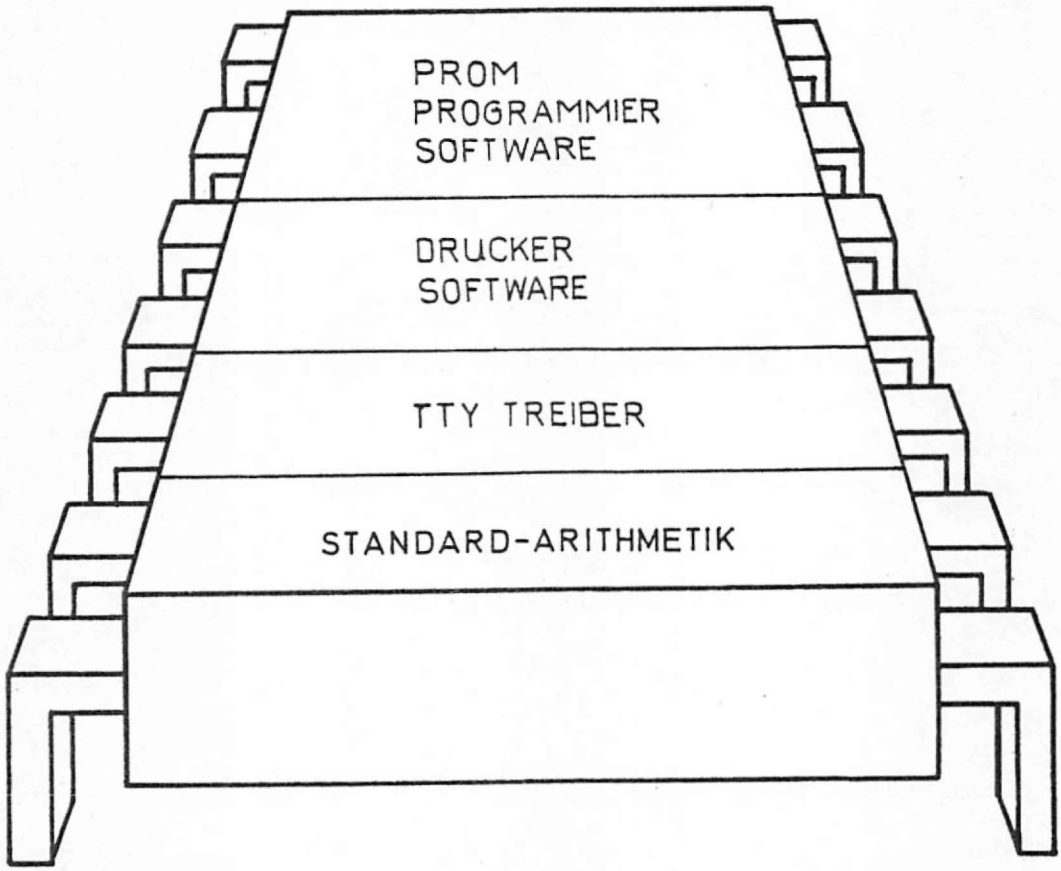
4. Listing der TTY Treiber Routinen

```
TTY TREIBER ROUTINEN FUER EC  USART          180778              PAGE   1
 LOC   OBJ CODE M STMT SOURCE STATEMENT                            ASM 4.5

                      1  *H TTY TREIBER ROUTINEN FUER ECB/K
                      2
0F6B                  3          ORG 0F6BH
                      4
                      5  SETUP:                    ;ROUTINE ZUR USART-
                      6                            ;INITIALISIERUNG
                      7
                      8
0F6B  97              9          SUB A             ;AKKU LOESCHEN
0F6C  D306           10          OUT (USDAT+2),A   ;DREIMALIGE ANWEISUNG,
0F6E  D306           11          OUT (USDAT+2),A   ;UM EINEN DEFINIERTEN
0F70  D306           12          OUT (USDAT+2),A   ;ZUSTAND ZU ERREICHEN
                     13
0F72  3E40           14          LD A,40H          ;RESETANWEISUNG
0F74  D306           15          OUT (USDAT+2),A
                     16
0F76  00             17          NOP
0F77  00             18          NOP               ;8 T-ZYKLEN ZEIT
                     19                            ;FUER RESET
                     20
0F78  3ECE           21          LD A,0CEH         ;STEUERWORT:
                     22                            ;  ASYN MODE (X16)
                     23                            ;  8 DATA BITS
                     24                            ;  NO PARITY
                     25                            ;  2 STOP BITS
0F7A  D306           26          OUT (USDAT+2),A
                     27
0F7C  3E27           28          LD A,27H          ;STEUERWORT:
                     29                            ;  RTS = '1'
                     30                            ;  ENABLE RXRDY
                     31                            ;  ENABLE TXRDY
                     32                            ;  DTR = '0'
0F7E  D306           33          OUT (USDAT+2),A
0F80  C9             34          RET
                     35
                     36
                     37
                     38  TTYIN:                    ;EIN ASCII CHARACTER IN DEN AKKU
0F81  DB06           39          IN A,(USDAT+2)
0F83  CB4F           40          BIT 1,A           ;RECEIVER READY ?
0F85  28FA           41          JR Z,TTYIN        ;WENN NICHT
0F87  DB04           42          IN A,(USDAT)      ;SONST DATA IN DEN AKKU
0F89  CBBF           43          RES 7,A           ;RESET PARITY
0F8B  C9             44          RET
                     45
                     46
                     47
                     48  TTYOUT:                   ;EIN ASCII CHARACTER AUS DEM AKKU
0F8C  F5             49          PUSH AF           ;AF RETTEN
                     50  READY:
0F8D  DB06           51          IN A,(USDAT+2)
0F8F  CB47           52          BIT 0,A           ;TRANSMITTER READY ?
0F91  28FA           53          JR Z,READY        ;WENN NICHT
0F93  F1             54          POP AF            ;SONST AF ZURUECK UND
0F94  D304           55          OUT (USDAT),A     ;DATA SENDEN
0F96  C9             56          RET
                     57
                     58
```

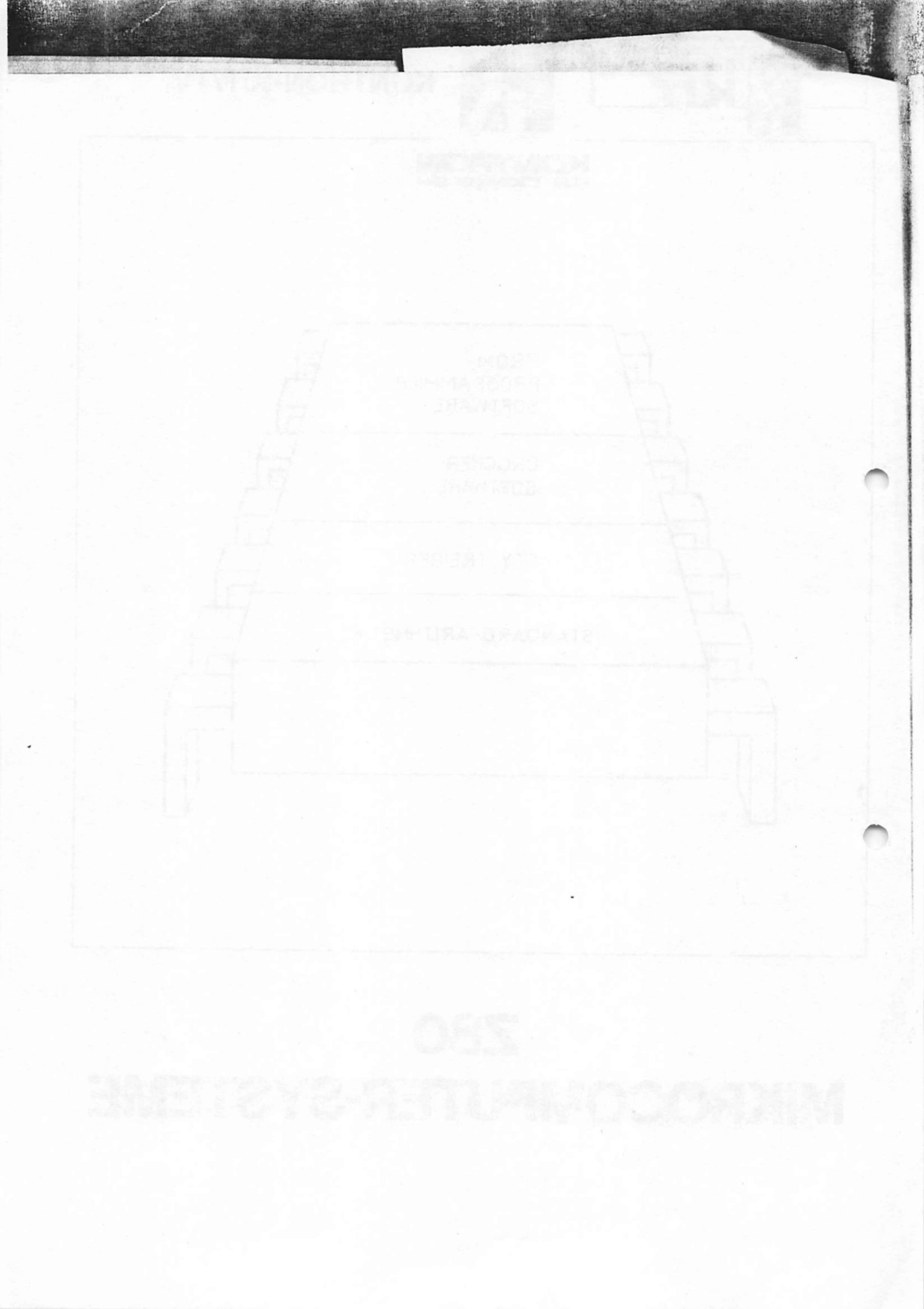
KONTRON KIT®

KONTRON-KIT/TV

KONTRON
ELEKTRONIK GMBH



Z80
MIKROCOMPUTER-SYSTEME



VI Z80-KIT/TV

Inhalt

Z80-KIT/TV

1. Einleitung

Z80-KIT/TV ist ein für Bildschirmbetrieb erweitertes Z80-KIT-Betriebsprogramm. Neben den bereits im Z80-KIT Manual beschriebenen Möglichkeiten, verfügt das Z80-KIT/TV Betriebsprogramm über eine zusätzliche Ansteuerung eines TV-Gerätes.
Bedingung für diese zusätzliche Anzeige ist eine der Baugruppen Z80-KIT/V (Bausatz) oder Z80-KIT/VZ. Ohne entsprechende Video-Hardware kann das TV-Betriebsprogramm wie das 1k-Betriebsprogramm benutzt werden.

2. Beschreibung des Betriebsprogramms Z80-KIT/TV

2.1. Allgemeine Beschreibung

Das Z80-KIT/TV Betriebsprogramm wird wie das 1k Betriebsprogramm als Firmware in einem Festwertspeicher geliefert. Es umfaßt 2 kByte und kann ohne jede Hardware-Änderung auf Steckplatz A2 der ECB/K untergebracht werden. Es umfaßt sämtliche Möglichkeiten des 1k-Programms (siehe hierzu Teil I).
Hinzu kommt eine Darstellung der KIT-Kommandos auf dem Bildschirm. Es werden jeweils das Kommando sowie die entsprechenden Daten bzw. Adressen dargestellt. Zusätzlich ist die Ausgabe von Speicherbereichen und Registerblöcken möglich. Da es sich dabei um zusätzliche Kommandos handelt, die mit Hilfe der KIT-Anzeige nicht dargestellt werden können, erfolgt in diesem Fall am KIT-Display keine Reaktion.
Die Darstellung auf dem Bildschirm erfolgt gemäß der Hardware (siehe Z80-KIT/V oder Z80-KIT/VZ) in 16 Zeilen zu 64 Zeichen. Selbstverständlich besteht auch beim Z80-KIT/TV die Möglichkeit, Routinen des Betriebsprogramms vom Anwenderprogramm aus aufzurufen. Eine Aufstellung finden Sie in Kap. 3.2.

2.2. Adreßeinstellung für Z80-KIT/V bzw. Z80-KIT/VZ

Das Betriebsprogramm Z80-KIT/TV ist zur Ansteuerung eines Bildschirms mit Hilfe der Einheiten Z80-KIT/V bzw. Z80-KIT/VZ geeignet. Die beiden Baugruppen unterscheiden sich lediglich durch die Adresse, des zur Ansteuerung verwendeten PIO. Standardmäßig ist das Betriebsprogramm Z80-KIT/TV zur Zusammenarbeit mit dem Z80-KIT/VZ bzw. Z80-KIT/V ausgelegt.

	Adresse für KIT/V Z80-KIT/TV1	Adresse für KIT/VZ Z80-KIT/TV2
Control	0B	FF
Data	09	FD
	(= PIO ECB/K)	(= PIO KIT/VZ)
Im Betr. prog. auf Adr.		
0497	0B	FF
049B	0B	FF
049F	0B	FF
04B1	09	FD
04D5	09	FD

2.3. Interruptverhalten

Das Betriebsprogramm Z80-KIT/TV arbeitet bezüglich der Bildschirmdarstellung mit Interrupt im MODE 2.
Auf diese Tatsache ist besonders zu achten, wenn die Bildschirm-Ansteuerroutinen des Betriebsprogramms von einem Anwenderprogramm aufgerufen werden, das ebenfalls mit Interrupt arbeitet.
Die vom Betriebsprogramm verwendete Interrupt-Service-Routine kann bei Aufruf der Bildschirm-Ansteuerroutinen durch das Anwenderprogramm mitverwendet werden. Der entsprechende Interruptvektor lautet

0718 H (I Reg. CPU 07)
(I Reg. PIO 18)

und ist bereits durch die Systeminitialisierung in den angegebenen Registern gespeichert. Die Verwendung anderer, vom Anwender erstellter Interrupt-Service-Routinen ist selbstverständlich ebenfalls möglich.

Achtung:
Bei Rücksprüngen vom Anwender ins Betriebsprogramm muß das I Register der CPU auf 07, das I Register der PIO auf 18 gesetzt sein!

2.4. Mitbenutzbare Routinen

Zusammenstellung

OBTAST
prüft, ob eine Taste des KIT Tastenfeldes gedrückt wurde. Bei der Rückkehr sind zwei Fälle möglich:
a) eine Taste wurde gedrückt
ZERO FLAG zurückgesetzt
b) kein Tastendruck
ZERO FLAG gesetzt
keine Übergabeparameter
Benützte Register: A B C D E F H L IX
Einsprungpunkt: 0251

WETAST
prüft, um welche Taste es sich handelt. Bei der Rückkehr sind zwei Fälle möglich:
a) es war immer noch die gleiche Taste gedrückt (während eines Tastendruckes mehrere Abfragen, da Prozessor sehr schnell!) Das A-Register enthält den Wert FF.
b) neue Taste erkannt
Das A-Register enthält den Tastencode der entsprechenden Taste
Keine Übergabeparameter
Benützte Register: A B C D E F H L IX
Einsprungpunkt: 0287

LEDS1
bringt die, in den zur Data Anzeige gehörenden LED Puffern stehenden Werte bei Data zur Anzeige.
Adresse der ‚Marke' beim Aufruf im HL Register-Paar.
a) Bit 0 von Marke gleich 0 → Anzeige ohne Konversion
b) 0 von Marke gleich 1 → Anzeige mit Konversion
Benützte Register: A B C D E F H L IX IY
Einsprungpunkt: 0309

LEDS2
bringt die, in den zur Address-Anzeige gehörenden LED Puffern stehenden Werte bei Address zur Anzeige.
Adresse der ‚Marke' beim Aufruf im HL Register-Paar
a) Bit 0 von Marke gleich 0 → Anzeige ohne Konversion
b) Bit 0 von Marke gleich 1 → Anzeige mit Konversion
Benützte Register: A B C D E F H L IX IY
Einsprungpunkt: 0317

ZEITKO
Zeitschleife von ca. 0,5 sec. Dauer
Keine Übergabeparameter
Benützte Register: A B C F
Einsprungpunkt: 0489

LOAD
Liest auf Cassette abgespeicherte Programme oder Daten ein
Startadresse des Zielspeichers beim Aufruf im Speicher 3C44/45 (siehe Kommando Load)
Benützte Register: A F H L
Einsprungpunkt: 00BE

STORE
speichert im RAM stehende Programme oder Daten auf Cassette ab
Startadresse und Endadresse des Quellspeichers beim Aufruf im Speicher 3C40/41 und 3C42/43 (siehe Kommando STORE)
Benützte Register: A B C D E F H L
Einsprungpunkt: 00FB

ABSPE1
entnimmt den beiden zur Data Anzeige gehörenden LED Puffern die dort gespeicherten (und angezeigten) Halbworte (0,1 . . . F), bildet ein Gesamtwort (00,01 . . . FF) und speichert es in die gewünschte Speicherzelle ab
Adresse der gewünschten Speicherzelle beim Aufruf im IY-Register
Benützte Register: A B F H L IX IY
Einsprungadresse: 041E

ABSPE2
entnimmt den vier zur Address-Anzeige gehörenden LED Puffern die dort gespeicherten (und angezeigten) Halbworte, bildet zwei Gesamtworte (high und low Byte) und speichert sie in die gewünschte Speicherzelle (low Byte) bzw. die nächstfolgende (high Byte) ab
Niedrigere Adresse der beiden gewünschten Speicherzellen beim Aufruf im IY-Register
Benützte Register: A B F H L IX IY
Einsprungadresse: 0426

RESPE1
entnimmt dem gewünschten Speicher den Wert, teilt dieses Gesamtwort auf in zwei Halbworte und belegt damit die beiden zur Data Anzeige gehörenden LED Puffer
Adresse der gewünschten Speicherzelle beim Aufruf im IY Register
Benützte Register: A B F H L IX IY
Einsprungadresse: 0443

RESPE2
entnimmt dem gewünschten und dem nachfolgenden Speicher den Wert, teilt die beiden Gesamtworte (low und high Byte) jeweils auf in zwei Halbworte und belegt damit die vier zur Address-Anzeige gehörenden LED Puffer
Niedrigere Adresse der beiden gewünschten Speicherzellen beim Aufruf im IY Register
Benützte Register: A B F H L IX IY
Einsprungpunkt: 044B

TAREG1
verschiebt die Inhalte der zur Data Anzeige gehörenden LED Puffer um eine Stelle in Richtung höherwertiger Stelle. Die dadurch freiwerdende unterste Stelle (Data 1) wird mit einem neuen Wert belegt
Neu einzulesender Wert beim Aufruf im A-Register
Benützte Register: A B C D E F H L
Einsprungpunkt: 036A

TAREG2
verschiebt die, in den zur Address-Anzeige gehörenden LED Puffern stehenden Werte in Richtung höherwertiger Stellen. Die freiwerdende unterste Stelle (Address1) wird mit einem neuen Wert belegt
Neu einzulesender Wert beim Aufruf im A-Register
Benützte Register: A B C D E F H L
Einsprungpunkt: 037E

ANZABF
zusammen gesetzte Routine aus LEDS1, LEDS2, OBTAST und WETAST, zeigt die in den LED Puffern gespeicherten Werte an und führt eine Tastenabfrage durch. Rücksprung bei Erkennen einer neuen Taste.
Adresse der ‚Marke' im HL Registerpaar
Bit 0 siehe LEDS1 bzw. LEDS2
a) Bit 1 von Marke gleich 1 → Data Anzeige
b) Bit 1 von Marke gleich 0 → keine Data Anzeige
c) Bit 2 von Marke gleich 1 → Address Anzeige
d) Bit 2 von Marke gleich 0 → keine Address Anzeige
Benützte Register: A B C D E F H L IX IY
Einsprungpunkt: 0400

FEHLER
bringt die ERROR Lampe zum Leuchten
Keine Übergabeparameter
Benützte Register: A F
Einsprungpunkt: 03B7

FEHLEX
löscht die ERROR Anzeige
Keine Übergabeparameter
Benützte Register: A F
Einsprungpunkt: 03BB

Folgende Unterprogramme sind nur in Verbindung mit einer Platine ECB-V oder ECB-VZ sinnvoll.

Bitte beachten Sie unbedingt, daß vor Aufruf dieser Unterprogramme das User-I-Register auf 07 gesetzt ist (siehe 2.3.).

SCREEN
Bildschirm wird gelöscht und Cursor geht nach links oben
Keine Übergabeparameter
Benützte Register: A B C F
Einsprungpunkt: 04AE

TVOUT
Das in Register A enthaltene ASCII-Zeichen wird auf dem Bildschirm ausgegeben bzw. wenn das in Register A enthaltene ASCII-Zeichen ein Cursor-Befehl ist, so wird dieser ausgeführt.
Keine Übergabeparameter
Keine Registerbeeinflussung
Einsprungpunkt: 04CB

PUTA
Die beiden im Register A enthaltenen Hexadezimalhalbbytes werden als 2 ASCII-Zeichen auf dem Bildschirm dargestellt (0 . . . F)
Keine Übergabeparameter
Keine Registerbeeinflussung
Einsprungpunkt: 04EB

PUTAB
Wie PUTA, jedoch folgt nach den beiden Zeichen ein BLANK.
Einsprungpunkt: 04FC

LINE FEED
Cursor geht zum Zeilenanfang der nächsten Zeile
Keine Übergabeparameter
Keine Registerbeeinflussung
Einsprungpunkt: 0507

PUTHL
Die im Registerpaar HL enthaltenen 4 Hexadezimalhalbbytes werden als 4 ASCII-Zeichen auf den Bildschirm ausgegeben (0 . . . F). Nach dem 4. Halbbyte folgt ein BLANK.
Keine Übergabeparameter
Keine Registerbeeinflussung
Einsprungpunkt: 051E

OUTSTR
Mit diesem Unterprogramm ist es möglich, lange Textblocks auszugeben. Die Anfangsadresse des abgespeicherten Textblockes muß im Registerpaar HL stehen. Der Text muß mit dem Zeichen * enden. (Dieses Zeichen = 2A). Das Endezeichen * wird nicht ausgegeben.
Benützte Register: HL (=Textzeiger)
Einsprungpunkt: 0510

ROUT1
Die Register A B C D E F H L IX IY PC SP werden incl. Überschrift auf den Bildschirm ausgegeben.
Benützte Register: A F H L
Keine Registerbeeinflussung
Einsprungpunkt: 0537

ROUTX
Wie ROUT;, jedoch ohne Überschrift
Einsprungpunkt: 0546

ROUT2
Wie ROUT1, jedoch werden statt der Register A . . . F die Zweitregister A' . . . F' ausgegeben.
Einsprungpunkt: 0579

2.5. Einsprungpunkte

Für den Rücksprung vom Userprogramm ins Betriebsprogramm existieren vier verschiedene Möglichkeiten:

Zusammenstellung

a) Einsprung bei 0000
Das Betriebsprogramm durchläuft die Initialisierungsroutine; sämtliche Userwerte werden auf 00 gesetzt. Der Bildschirm wird gelöscht.

b) Einsprung bei 018B
Das Betriebsprogramm durchläuft die Aktualisierungsroutine; sämtliche aktuellen Registerwerte des Userprogramms mit Ausnahme des Programmzählers und des HL Registerpaars werden in dem dafür vorgesehenen RAM Bereich abgelegt (siehe Kap. 35).

NB: In den Speicherzellen 3CFE und 3CFF muß 0B bzw. 00 stehen! HL und PC sind gesondert zu retten.
Auf dem Bildschirm erscheint der aktuelle Registersatz ohne Zweitregister mit Überschrift.

c) Einsprung bei 000B
Das Betriebsprogramm läuft ohne zusätzliche Funktion an.
Keine Beeinflussung der Userwerte.
Keine Bildschirmaktion.

d) Einsprung bei 0038
Das Betriebsprogramm durchläuft die Fehlerroutine (ERROR Lampe an). Keine Beeinflussung der Userwerte. Auf dem Bildschirm erscheint die Meldung Befehl = FF

Beispiel
Nach Beendigung eines Userprogrammes soll ein Rücksprung ins Betriebsprogramm ohne zusätzliche Funktionen erfolgen:

	Userprogramm
JP 000B H (C3 0B 00)	Rücksprung

2.6 Systemkonstante

Adresse	Inhalt	Kommentar	
		PORTS:	
07D5	0C	DEFB 0CH	;DATA 1 LED
D6	0D	DEFB 0DH	;DATA 2 LED
D7	0E	DEFB 0EH	;ADR. 1 LED
D8	0F	DEFB 0FH	;ADR. 2 LED
D9	1C	DEFB 1CH	;ADR. 3 LED
DA	1D	DEFB 1DH	;ADR. 4 LED
			;Hex zu Sieben-
		SEGM:	Segment Tabelle
DB	5F	DEFB 05FH;	0
DC	06	DEFB 06H;	1
DD	3B	DEFB 3BH;	2
DE	2F	DEFB 2FH;	3
DF	66	DEFB 66H;	4
E0	6D	DEFB 6DH;	5
E1	7D	DEFB 7DH;	6
E2	07	DEFB 07H;	7
E3	7F	DEFB 7FH;	8
E4	6F	DEFB 6FH;	9
E5	77	DEFB 77H;	A
E6	7C	DEFB 7CH;	B
E7	59	DEFB 59H;	C
E8	3E	DEFB 3EH;	D
E9	79	DEFB 79H;	E
EA	71	DEFB 71H;	F
		Userregister:	
3C00		SPC	Programmzähler
02		SSP	Stackpointer
04		SIY	IY Register
06		SIX	IX „
08		SR	R „
09		SI	I „
0A		SE0	E' „
0B		SD0	D' „
0C		SC0	C' „
0D		SB0	B' „
0E		SF0	F' „
0F		SA0	A' „
10		SL0	L' „
11		SH0	H' „
12		SE	E „
13		SD	D „
14		SC	C „
15		SB	B „
16		SF	F „
17		SA	A „
18		SL	L „
19		SH	H „
		LED Puffer:	
30		LED Puffer 1	DATA ; Puffer
31		LED Puffer 2	DATA 2 Puffer
32		LED Puffer 3	ADDRESS 1 Puffer
33		LED Puffer 4	ADDRESS 2 Puffer
34		LED Puffer 5	ADDRESS 3 Puffer
35		LED Puffer 6	ADDRESS 4 Puffer

3. Hinweise zur Bedienung des Z80-KIT bei Verwendung des Betriebsprogramms Z80-KIT/TV (Kommandosprache)

3.1. Setzen von Registern auf einen gewünschten Wert

Dies wird durch folgende Tastenfolge erreicht:

SET register XXX . . . X EX (X = 0,1, ... F)

Für „register" ist **die** Taste zu drücken, die dem gewünschten Register zugeordnet ist. Nach dem Drücken dieser Taste verlöschen die jeweils nicht benutzten Anzeigen, während auf den verbleibenden LED's der augenblickliche Wert dieses Registers angezeigt wird.
XXX . . . X ist eine beliebig lange Folge von Hex-Zeichen, die von rechts nach links in die entsprechende Anzeige geschoben wird. Steht der gewünschte Wert auf der Anzeige, kann mit EX abgeschlossen werden. Der neue Wert wird dann übernommen und abgespeichert.
Auf dem Bildschirm wird nach dem Kommando

„SET register"

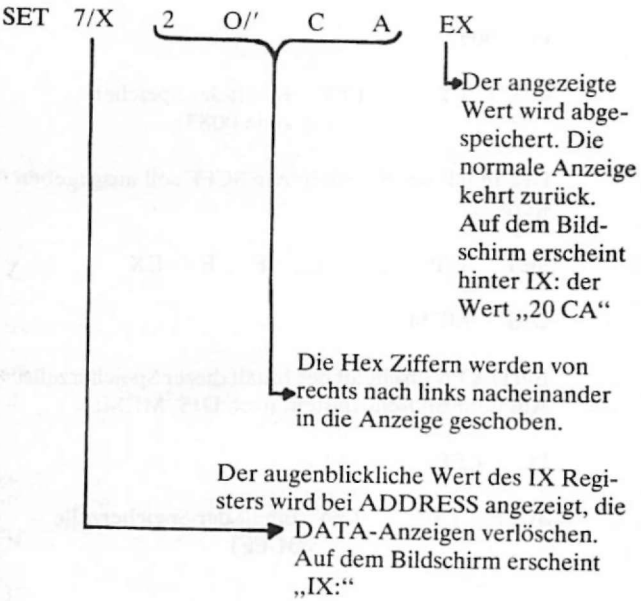
das angesprochene Register auf dem Bildschirm angezeigt. Z.B.:

„PC:"

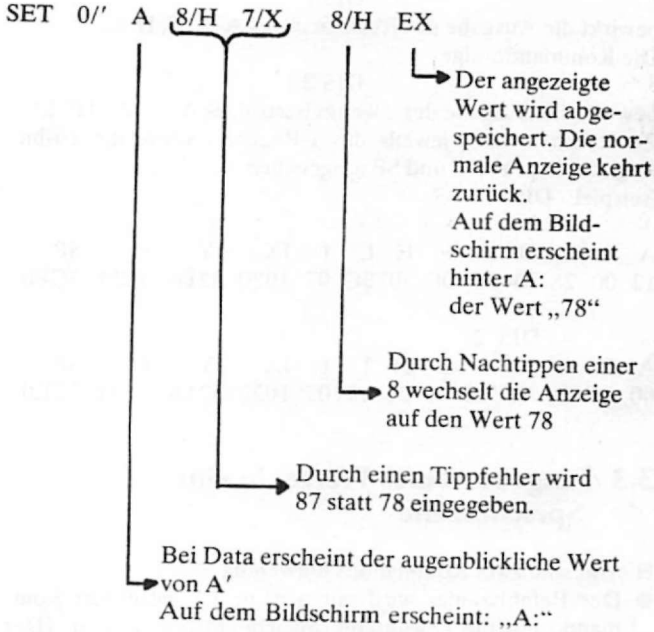
Nach Eingabe der Daten und Betätigen der Taste „EX" wird auch der in das Register übernommene Wert dargestellt.
Sonderform: Die Kommandofolge „SET 1" oder „SET 2" hat die gleiche Funktion wie „DIS 1" oder „DIS 2" (siehe 3.2.2.), hierbei können Register nicht geändert werden.

Beispiele:

Das IX Register (16 bit!) soll auf den Wert 2 OCA gesetzt werden.



Das 8 bit Register A' soll auf den Wert 78 gesetzt werden



3.2. Anzeige von Registerinhalten

3.2.1. Anzeige von einzelnen Registern

Dies wird durch folgende Tastenfolge erreicht:

DIS register

Für „register" ist wiederum **die** Taste zu drücken, die dem gewünschten Register zugeordnet ist. Der Inhalt des Registers erscheint an den Anzeigeelementen und zwar bei ADDRESS (4 Stellen), wenn es sich um ein 16 bit Register handelt (z.B. PC) bzw. bei DATA (2 Stellen), wenn es sich um ein 8 bit Register handelt (z.B. A Register).
Auf dem Bildschirm erscheint gleichzeitig das gewünschte Register sowie dessen Inhalt.
Beispiele:
Der Inhalt des 8 bit Registers H soll angezeigt werden.

DIS 8/H

Das Ergebnis erscheint bei DATA (2-stellig!)

Auf dem Bildschirm erscheint:
H: XX (XX = augenblicklicher Wert von H)
Der Inhalt des 16 bit umfassenden Stack Pointers (SP-Register) soll angezeigt werden

DIS 5/S

Das Ergebnis erscheint bei ADDRESS (4-stellig)
Auf dem Bildschirm erscheint
SP: XXXX (XXXX = augenbl Wert von SP)

3.2.2. Anzeige von Registersätzen
(nur auf dem Bildschirm!)
Die Kommandofolge

DIS 1

bewirkt die Ausgabe des Registersatzes A . . . F, H, L.
Die Kommandofolge

DIS 2

bewirkt die Ausgabe des Zweiregistersatzes A′ . . . F′, H′, L′.
Zusätzlich werden jeweils das I-Register, sowie die 16-bit-Register IX, IY, PC und SP ausgegeben.
Beispiel: DIS 1

```
A  B  C  D  E  F  H  L  I  IX   IY   PC   SP
12 00 28 31 FF 00 50 3C 07 1020 321A 3C51 3CE0
```

DIS 2

```
A′ B′ C′ D′ E′ F′ H′ L′ I  IX   IY   PC   SP
00 3C 29 7F 66 48 25 17 07 1020 321A 3C51 3CE0
```

3.3 Eingeben eines Wertes in eine Speicherzelle

Hierfür sind zwei Kommandos notwendig

- Der Befehlszähler wird mit dem in 3.1. gezeigten Kommando auf die gewünschte Speicheradresse gesetzt. (Der Wert erscheint bei ADDRESS)

  SET 4/P adresse EX

- Anschließend wird die Tastenfolge

  SET MEM wert EX

  gegeben.
  Nach SET MEM erscheint bei DATA der augenblickliche Inhalt der Speicherzelle
  Der Ausdruck „wert" ist durch den gewünschten Wert zu ersetzen.
  Auf dem Bildschirm erscheint nach SET 4/P:

  PC:

  Nach Drücken der EX-Taste wird zusätzlich die eingetippte Adresse dargestellt.
  PC: XXXX (XXXX = eingetippter Wert)

  Wird nun SET MEM Wert EX gegeben, erscheint in einer neuen Zeile die eben eingegebene Adresse mit dem eingetippten Wert:

  XXXX : YY (XXXX Adr. d. Speichers
  YY zugewiesener Wert)

Beispiele: Die Speicherzelle mit der Adresse 3C83 soll den Wert FF erhalten

SET 4/P 3/I C 8/H 3/I EX

Auf dem Bildschirm steht anschließend:

PC : 3C83

SET MEM F F EX

Auf dem Bildschirm wird unter der vorigen Zeile

3C83

ausgegeben.
Die Speicherzelle mit der Adresse 3CC7 soll den Wert 12 erhalten

SET 4/P 3/I C C 7/X EX

SEM MEM 1 2 EX

Auf dem Bildschirm erscheint:

PC : 3CC7

3CC7 12

3.4. Ausgabe des Inhaltes einer Speicherzelle auf die Anzeige

Auch hier sind zwei Kommandos notwendig. Das erste Kommando dient dabei wie unter 3.1. der Identifizierung der gewünschten Zelle, während das zweite Kommando die Ausgabe Ihres Inhalts verursacht.

SET 4/P adresse EX

DIS MEM

Auf dem Bildschirm wird in bekannter Weise zuerst

PC : XXXX (XXXX: eingegebene Adresse)

dargestellt.
Nach DIS MEM erscheint zusätzlich
XXXX YY (XXXX: eingegebene Adresse
YY: aktueller Wert dieser Speicherzelle)

Beispiele: Der Inhalt der Speicherzelle 0083 soll ausgegeben werden

SET 4/P 0/′ 0/′ 8/H 3/I EX

DIS MEM

Bei DATA erscheint der Inhalt dieser Speicherzelle
Auf dem Bildschirm steht nach Abschluß der beiden Kommandos:

PC : 0083

0083 YY (YY = Inhalt der Speicherzelle 0083)

Der Inhalt der Speicherzelle 3CFF soll ausgegeben werden

SET 4/P 3/I C F F EX

DIS MEM

Bei DATA erscheint der Inhalt dieser Speicherzelle.
Auf dem Bildschirm steht nach DIS MEM:

PC : 3CFF

3CFF YY (YY Inhalt der Speicherzelle 3CFF)

3.5. Eingabe eines Anwenderprogrammes

Für die Eingabe von Programmen ist der Eingabemodus vorgesehen, der das bequeme Eingeben auch längerer Programme ermöglicht. Das Setzen des Befehlszählers ist daher nur einmal nötig und geschieht danach automatisch.

a) Als erstes wird der Befehlszähler auf **die** Adresse gesetzt, auf der der erste Befehl des Anwenderprogrammes stehen soll (siehe 4.5.1.).

SET 4/P adresse EX

b) Daraufhin wird der Eingabemodus eingeleitet indem die Taste

INPUT

gedrückt wird.

c) Nun können Sie ein Programm eingeben, und zwar in folgender Weise

X . . X EX

X . . X EX

X . . X EX

X = gewünschte Daten in hexadezimalem Format

Der Befehlszähler, der jeweils die aktuelle Speicheradresse enthält, wird bei jedem EX um 1 erhöht. Gleichzeitig kommt der aktuelle Inhalt dieser Zelle zur Anzeige. Es werden immer Adresse und zugehörige Daten **parallel** angezeigt. Der bei DATA stehende Wert **kann** dann durch Eintippen von Hex-Ziffern verändert werden. Ein sofortiges Drücken der Taste EX, **ohne** vorherige Veränderung führt zur Wieder-Übernahme der alten (noch auf der Anzeige stehenden) Daten.

d) Haben Sie Ihr Programm vollständig eingegeben, müssen Sie den Eingabemodus wieder verlassen.
Dies geschieht durch Drücken der Taste STORN
Als sichtbare Kontrolle für das Verlassen des Input-Modus, wird die gesamte Anzeige kurzzeitig ausgeschaltet.

Auf die Eingabe

INPUT DATEN EX

erscheint auf dem Bildschirm die Überschrift:

INPUT

sowie die Adresse der angesprochenen Speicherzelle und der, in diese Speicherzelle übernommene Wert. Mit jeder weiteren Dateneingabe werden nunmehr ohne Adressen die, in die nächsthöheren Speicherzellen eingegebenen Werte angezeigt. Wenn die angesprochene Speicherzelle eine auf Null endende Adresse hat (z.B.: 3C60), so wird diese Adresse in einer neuen Zeile angezeigt. Durch diese Art der Bildschirmanzeige steht das eingegebene Programm jeweils im Blockformat zur Kontrolle auf dem Bildschirm bereit.

Beispiel:

INPUT

3C58 19 7E 06 0F C5 06 FF 00

3C60 ED 79 10 FB . . . . . .

3770 . . . . . .

Haben Sie Ihr Programm vollständig eingegeben, müssen Sie den Eingabemodus wieder verlassen.
Auf das Komando STORN erscheint auf dem Bildschirm das Wort:

END

welches das Beenden des INPUT-Modus bestätigt.

Beispiel:
Das folgende kleine Programm soll eingegeben werden. Anfangsadresse soll 3C70 sein.

3E
0F
D3
0A
78
D3
08
C3
0B
00

Folgende Aktivitäten sind notwendig

a) SET 4/P 3/I C 7/X 0I' EX

b) INPUT

c) 3/I E EX
0/' F EX
.
.
.
.
0/' 0/' EX

d) STORN
Am Bildschirm steht nach dem STORN Kommando

PC : 3C70
INPUT
3C70 3E 0F D3 0A 78 D3 08 C3 0B 00 END

3.6. Ausgabe von größeren, zusammenhängenden Speicherbereichen

3.6.1. Sequentielle Ausgabe

Wie bei der Eingabe ist auch bei der Ausgabe längerer Programme eine Bedienungs-Erleichterung vorgesehen. Auch hier ist lediglich die Eingabe der Anfangsadresse notwendig, das Weiterschalten geschieht dann automatisch.

a) Zuerst wird der Befehlszähler auf die Anfangsadresse des gewünschten Speicherbereichs gesetzt und der Wert dieser Zelle abgerufen (4.5.4.)

SET 4/P adresse EX

DIS MEM

Der Inhalt dieser Zelle wird bei DATA angezeigt, während die zugehörige Adresse bei ADDRESS steht.
Auf dem Bildschirm erscheint:
PC : XXXX (XXXX = eingegebene Adresse)
XXXX YY (YY = Inhalt dieser Speicherzelle)

b) Wollen Sie nun den Inhalt der nachfolgenden Zelle ausgeben, drücken Sie nur

IDM

Dies hat zur Folge, daß der Befehlszähler um eins erhöht wird und der Inhalt der damit bezeichneten Speicherzelle bei DATA erscheint.
Auf dem Bildschirm wird dieser Inhalt mit Adresse in einer neuen Zeile dargestellt. Bei weiterem Betätigen von IDM werden die folgenden Werte jeweils hinter den bereits vorhandenen Wert geschrieben. Handelt es sich um eine auf 0 endende Adresse (z.B. 3C80), so wird eine neue Zeile (mit Adresse) begonnen.

Durch festgesetztes Drücken von IDM können somit z.B. auch längere Programme durchgesehen und überprüft werden, da nur eine Taste zu betätigen ist und die Speicher-Adresse und ihr Inhalt jeweils gleichzeitig angezeigt werden.
Auf dem Bildschirm bleibt dabei die Gesamtinformation als „Block" erhalten.

Beispiel:
Das unter 3.5. angegebene Programm befindet sich im Speicher mit Anfangsadresse 3C70 und soll schrittweise angezeigt werden.

a) SET 4/P 3/I C 7/X 0/' EX

DIS MEM

	ADDRESS	DATA
Danach ergibt sich:	3C70	3E
b) IDM →	3C71	0F
IDM →	3C72	D3
IDM →	3C73	0A
IDM →	3C74	78

usw.

Auf dem Bildschirm ergibt sich:

PC : 3C70

3C70 3E

3C71 0F D3 0A 78 usw.

3.6.2. Blockausgabe (nur am Bildschirm)

Die unter 3.6.1 beschriebene Ausgabe kann bezüglich des Bildschirms wesentlich schneller erreicht werden. Durch die Kommandofolge

SET 4/P XXXX EX (XXXX = gewünschte Anfangsadresse)
DIS DIS YY EX (YY = Anzahl der darzustellenden Bytes)

erfolgt eine Ausgabe von YY Bytes auf den Bildschirm. Die Information (= Speicherinhalt) wird im Blockformat dargestellt. Bei ADDRESS und DATA der KIT-Anzeige werden Adresse und Inhalt der letzten der YY Speicherzellen angezeigt.
Die größtmögliche Bytezahl, die ausgegeben werden kann, beträgt 100H (≙ DIS DIS 00 EX). Bitte beachten Sie, daß mehr als F1H Bytes nur dann sinnvoll sind, wenn der Befehlszähler eine Endadresse von 0 (z.B. 3C50) hat, da auf dem Bildschirm nur 16 Zeilen zur Verfügung stehen und andernfalls dadurch der obere Teil aus der Anzeige hinausgeschoben wird. Durch betätigen der Taste IDM kann der ausgegebene Speicherblock weitergeführt werden.

Beispiel:
Ausgabe der ersten zwanzig (hex) Bytes des Betriebsprogramms
Kommandofolge

SET 4/P 0/' 0/ 0/' 0' EX

DIS DIS 20 EX

Auf dem Bildschirm wird folgendes gezeigt:
PC : 0000

0000 3E 00 3D . . . . . CB C6

0010 CB CE . . . . . CA F1 00

3.7. Starten eines Anwenderprogramms

Dazu setzen Sie den Befehlszähler (3.1.) auf die Anfangsadresse ihres Programmes und drücken daraufhin die Taste

START

Solange das Anwenderprogramm bearbeitet wird, erlöschen alle Anzeigeelemente, (außer das Anwenderprogramm wird zur Ansteuerung benützt).

Beispiel:
Ihr Anwenderprogramm steht im Speicher mit der Anfangsadresse 3C66. Sie wollen es starten

SET 4/P 3/I C 6/Y 6/Y EX

START

Die Anzeigen erlöschen und leuchten erst nach Beendigung des Anwenderprogramms wieder auf.
Keine Reaktion am Bildschirm.

3.8. Programmausführung im Einzelschrittverfahren

(„single step")
Hierfür setzen Sie Ihren Befehlszähler auf die gewünschte Adresse im Programm (oder lassen das Programm per Breakpoint zum Halten kommen) und betätigen anschließend die Taste

STEP

Es wird genau **ein Befehl** ausgeführt. Bei ADDRESS erscheint der nun aktuelle Stand des Befehlszählers, bei DATA der Inhalt dieser Speicherzelle = Operations Code des nächsten, auszuführenden Befehls.
Gleichzeitig zur KIT-Anzeige erscheint auf dem Bildschirm der komplette Registersatz (ohne Zweitregister). Bei jedem weiteren „single step" erscheinen die Register ohne Überschrift. Wird wieder eine Registerüberschrift gewünscht, ist vor dem nächsten „single step" die Taste STORN zu drücken.

Beispiel:
Gegeben sei wieder das Beispiel aus 3.5. Der Befehlszähler sei bereits auf 3C70 gesetzt und ein DIS MEM durchgeführt. Das Register B enthalte 55.

	ADDRESS	DATA
	3C70	3E
STEP →	3C72	D3
STEP →	3C74	78
STEP →	3C75	D3

usw.

Auf dem Bildschirm ergibt sich:

PC : 3C70
3C70 3E

A	B	C	D	E	F	H	L	I	IX	IY	PC	SP
0F	55	XX	XX	XX	XX	XX	XX	XX	XXXX	XXXX	3C72	3CE0
0F	55	XX	XX	XX	XX	XX	XX	XX	XXXX	XXXX	3C74	3CE0
55	55	XX	XX	XX	XX	XX	XX	XX	XXXX	XXXX	3C75	3CE0

usw.

(X = 0 . . . F)

Bitte machen Sie sich an dieser Stelle klar, daß EIN BEFEHL im System Z80 EIN, ZWEI, DREI ODER VIER BYTES umfaßt.

Beim Einzelschrittverfahren springt der Befehlszähler bei jedem Betätigen der Taste STEP um 1, 2, 3 oder 4, je nachdem, ob die Ausführung 1, 2, 3 oder 4 Bytes langen Befehls durch die Betätigung ausgelöst wurde.

3.9. Setzen, Löschen und Anzeigen eines Haltepunkts

(Breakpoint) (nur im RAM möglich)

NB: Es ist immer nur ein Haltepunkt möglich. Die Veränderung eines bestehenden Haltepunkts erfolgt über einen erneuten Setzvorgang, wobei der alte Breakpoint automatisch gelöscht wird.

a) Das Setzen eines Haltepunkts auf eine bestimmte Programmadresse wird durch folgende Tastenfolge erreicht:

SET BR adresse EX

„adresse" ist dabei durch den gewünschten Wert zu ersetzen. Nach SET BR erscheint bei ADDRESS die augenblickliche Haltepunktadresse. Der Wert 0000 bedeutet kein BREAK gesetzt.
Auf dem Bildschirm erscheint nach SET BR:

BREAK:

Nach Eintippen der neuen Adresse und nach Abschluß durch EX wird die neue Adresse hinter „BREAK:" dargestellt.

Beispiel:
In dem bekannten Anwenderprogramm, das im Speicher von 3C70 an untergebracht ist, soll zu Prüfzwecken ein Haltepunkt auf Adresse 3C74 gesetzt werden.
NB: Ein Haltepunkt darf selbstverständlich nur auf solche Speicherzellen gesetzt werden, in denen der Opcode eines Befehls steht!
Folgendes Kommando ist notwendig:

SET BR 3/IC 7/X 4/P EX

Auf dem Bildschirm steht:

BREAK: 3C74

Wird das Anwenderprogramm gestartet, und erreicht der Befehlszähler diese Adresse, so wird das Programm angehalten. Sichtbar wird dies durch das Wiederaufleuchten der Anzeigeelemente. Bei ADDRESS erscheint der Wert der Breakpointadresse, bei DATA der Opcode des nächsten Befehls.
Auf dem Bildschirm wird der komplette Registersatz (ohne Zweitregister) angezeigt. Bei PC (Befehlszähler) steht die Adresse des nächsten auszuführenden Befehls.

Durch das Auflaufen auf den Haltepunkt (per START oder STEP) wird dieser gelöscht. Dadurch kann sofort ein neuer Haltepunkt, auch mit dem alten Wert vereinbart werden.
Eine weitere Abarbeitung des Programms kann an dieser Stelle mit START oder STEP ausgelöst werden.

b) Das Löschen eines Haltepunktes kann auf zweierlei Arten erfolgen
- Wie oben beschrieben durch Überlaufen mit START oder STEP wenn das Programm den Haltepunkt erreicht hat,
- Durch Setzen der Haltepunktadresse auf den Wert 0000 wenn Ihr Mikrocomputer nicht mit der Abarbeitung eines Anwenderprogrammes beschäftigt ist.

c) Das Anzeigen der aktuellen Breakpointadresse erfolgt durch das Kommando

DIS BR

3.10. Abspeichern auf Cassette

Hierzu ist es notwendig die Anfangs- und Endadresse des abzuspeichernden Programmstücks einzugeben. Diese Adressen werden in insgesamt 4 Speicherzellen des RAM abgelegt, von wo aus sie von der entsprechenden Programmroutine abgeholt werden.
Die Adressen lauten wie folgt:

3C40	Startadresse	low Byte
3C41	Startadresse	high Byte
3C42	Endadresse	low Byte
3C43	Endadresse	high Byte

Beispiel:
Ein Programm, das im Speicher von Adresse 3C60 bis 3CA7 steht, soll auf Cassette übertragen werden. Die Eingabe der Start- bzw. Endadresse erfolgt durch die Kommandofolge:

SET 4/P 3/I C 4/P 0/' EX

INPUT

6/X 0/' EX
3/I C EX
A 7/X EX

3/I C EX

STORN

Um später ein problemfreies Wiedereinlesen zu gewährleisten, muß folgendes beachtet werden:

Der Beginn einer „Sendung" wird auf Grund der Einfachheit des Cassetten-Interface durch ein längerdauerndes high Signal gebildet, dem die fallende Flanke des Startbits des ersten Wortes folgt. Um später das Programm sicher wieder laden zu können, empfiehlt es sich, dieses high-Signal auf ca. ½Minute auszudehnen (siehe „Load"). Die mit dem Zählwerk des Cassettenrecorders verbundene Ungenauigkeit kann damit ausgeschaltet werden.

Das Ende der „Sendung" wird mit Hilfe eines Breakcharakters markiert, der automatisch dem USER-Programm angeschlossen wird.

Das obengenannte Programm soll auf Cassette übertragen werden

- Stellen Sie Ihr Cassettengerät so ein, daß ein definierter Bereich des Bandes zur Bespielung zur Verfügung steht u. notieren Sie den Wert des Bandzählwerkes.
- Schalten Sie auf Aufnahme und steuern Sie Ihr Gerät voll aus, bzw. verwenden Sie die automatische Aussteuerung.
- Lassen Sie jetzt das Band laufen (damit wird das längerdauernde high auf das Band geschrieben).
- Nach ca. 30 Sekunden notieren Sie wieder den Stand des Bandzählwerkes und drücken die Taste Store. Auf der rechten Anzeige wird, solange auf das Band geschrieben wird, ein S für STORE angezeigt. Nach Beendigung des Abspeichervorgangs erscheint wieder die normale Anzeige.

Auf dem Bildschirm wird das Kommando sowie Start- und Endadresse des abzuspeichernden Bereiches angezeigt.

z.B.: STORE 3C50 3C78

Der Cursor bleibt solange in der gleichen Position, bis der Abspeichervorgang beendet ist. Erst danach springt er in eine neue Zeile

- Schalten Sie jetzt Ihr Cassettengerät ab. Damit ist Ihr Programm auf Band geschrieben.

Bei der Bedingung Endadr. < Anfangsadr., wird zusätzlich zu ERROR Lampe die Meldung

ENDADR. < STARTADR.

am Bildschirm ausgegeben und es erfolgt keine Datenübertragung.

3.11. Laden von Cassette in den Speicher

Hierzu ist es notwendig, die Anfangsadresse des Speicherbereiches, in den die Information geladen werden soll, anzugeben.
Die Speicherzellen in denen diese Adresse steht sind:

3C44	Startadresse	low Byte
3C45	Startadresse	high Byte

Beispiel:
Ein auf Band stehendes Programm soll in den Speicher geladen werden. Die Anfangsadresse sei 3C80
Die Eingabe erfolgt durch:

```
SET   4/P   3/I   C   4/P   4/P   EX

INPUT
8/H   0/'   EX

3/I   C     EX

STORN
```

- Stellen Sie Ihr Cassettengerät auf den ersten der beiden notierten Werte des Bandzählwerkes und schalten Sie auf Wiedergabe.
- Lassen Sie das Band laufen.
- Sobald das Bandzählwerk in etwa den Mittelwert der beiden beim Abspeichern notierten Werte zeigt, drücken Sie die Taste LOAD.

Auf dem Bildschirm wird das Kommando selbst, und die Anfangsadresse des Zielspeichers angezeigt

z.B.: LOAD 3C60

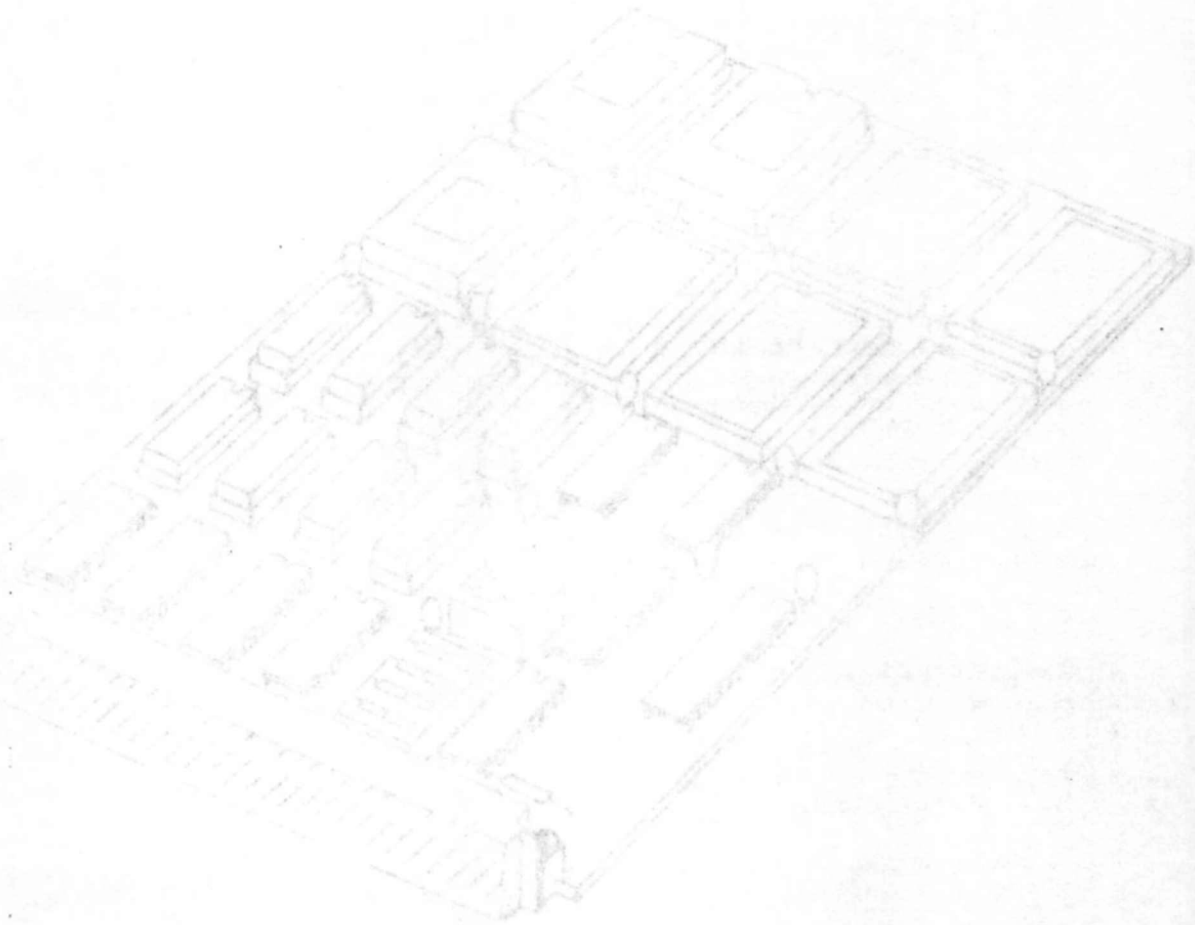
Der Cursor bleibt solange in der gleichen Zeile stehen, bis der Ladevorgang abgebrochen wird.
Das Programm wartet jetzt auf die erste fallende Flanke am Eingang der seriellen Schnittstelle, was gleichbedeutend mit dem Beginn des ersten Wortes ist. Anschließend beginnt der eigentliche Ladevorgang. Während des Ladevorgangs wird auf der rechten Anzeige ein L für LOAD angezeigt.
Nach Beendigung des Ladevorgangs (Erkennen des Breakcharakters) leuchtet wieder die normale Anzeige auf.

- Schalten Sie Ihr Cassettengerät ab.

Damit ist die Information in den Speicher geladen.
NB: Sollte während des Ladens das Lämpchen ERROR aufleuchten, so wurde ein Fehler bei der Übertragung erkannt und der gesamte Vorgang ist zu wiederholen.
Auf dem Bildschirm wird dabei die Art des Fehlers angegeben:
z.B.:

PARITY ERROR.

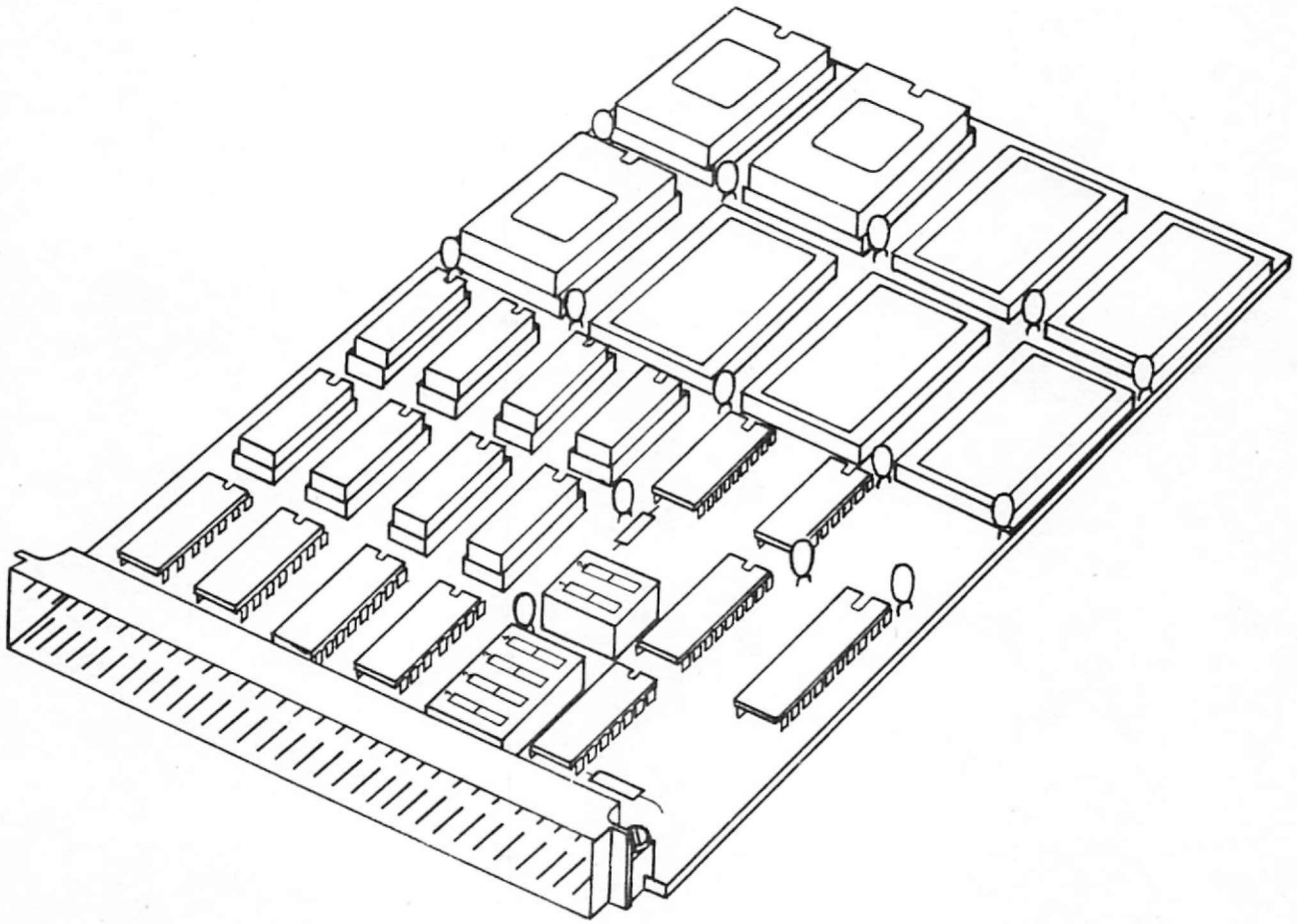
Die ERROR Lampe erlischt, sobald erneut LOAD gedrückt wird.



KONTRON KIT®

BASIC Erweiterung KONTRON KIT/ZBASIC

KONTRON
ELEKTRONIK GMBH



Z80 MIKROCOMPUTER-SYSTEME

VII Z80-KIT/ZBASIC

Inhalt: Seite

1. Übersicht

KIT/ZBASIC ist neben den bereits bestehenden Bausteinen eine weitere Möglichkeit zum Ausbau des bekannten Billig-Computer-Systems Z80-KIT.
Die Erweiterung besteht aus einer Steckkarte im Einfach-Europaformat, auf der der BASIC-Interpreter ZBASIC in PROM-residenter Form untergebracht ist. Für den Aufbau von Benutzerprogrammen steht zusätzlich statisches RAM zur Verfügung.
ZBASIC stellt einen äußerst leistungsfähigen BASIC-Interpreter dar, der speziell über ein umfangreiches Mathematikpaket verfügt. Als Beispiele sei die Möglichkeit zur Berechnung des Hyperbelsinus oder der Arcustangensfunktion genannt.
ZBASIC ist für den Betrieb mit einer ASCII-Tastatur und einer Bildschirmanzeige vorbereitet.
Die Ausgaben erfolgen über eine parallele Schnittstelle. Das verwendete Format paßt zu den beiden Bildschirm-Ansteuerungs-Einheiten Z80-KIT/V bzw. Z80-KIT/VZ, die beide ohne Veränderung eingesetzt werden können.
Die Eingaben werden von ZBASIC über eine interruptgesteuerte Parallelschnittstelle erwartet. Bei Verwendung von Z80-KIT/V bzw. Z80-KIT/VZ ist ein direkter Anschluß an den noch verbleibenden Port der entsprechenden PIO möglich.

KIT/ZBASIC kann durch Einstecken in einen freien Steckplatz des Z80-KIT-Busses und anschließendes Starten sofort in Betrieb genommen werden.
Einzige Voraussetzung hierfür ist das Vorhandensein einer ASCII-Tastatur und einer Bildschirmanzeige.

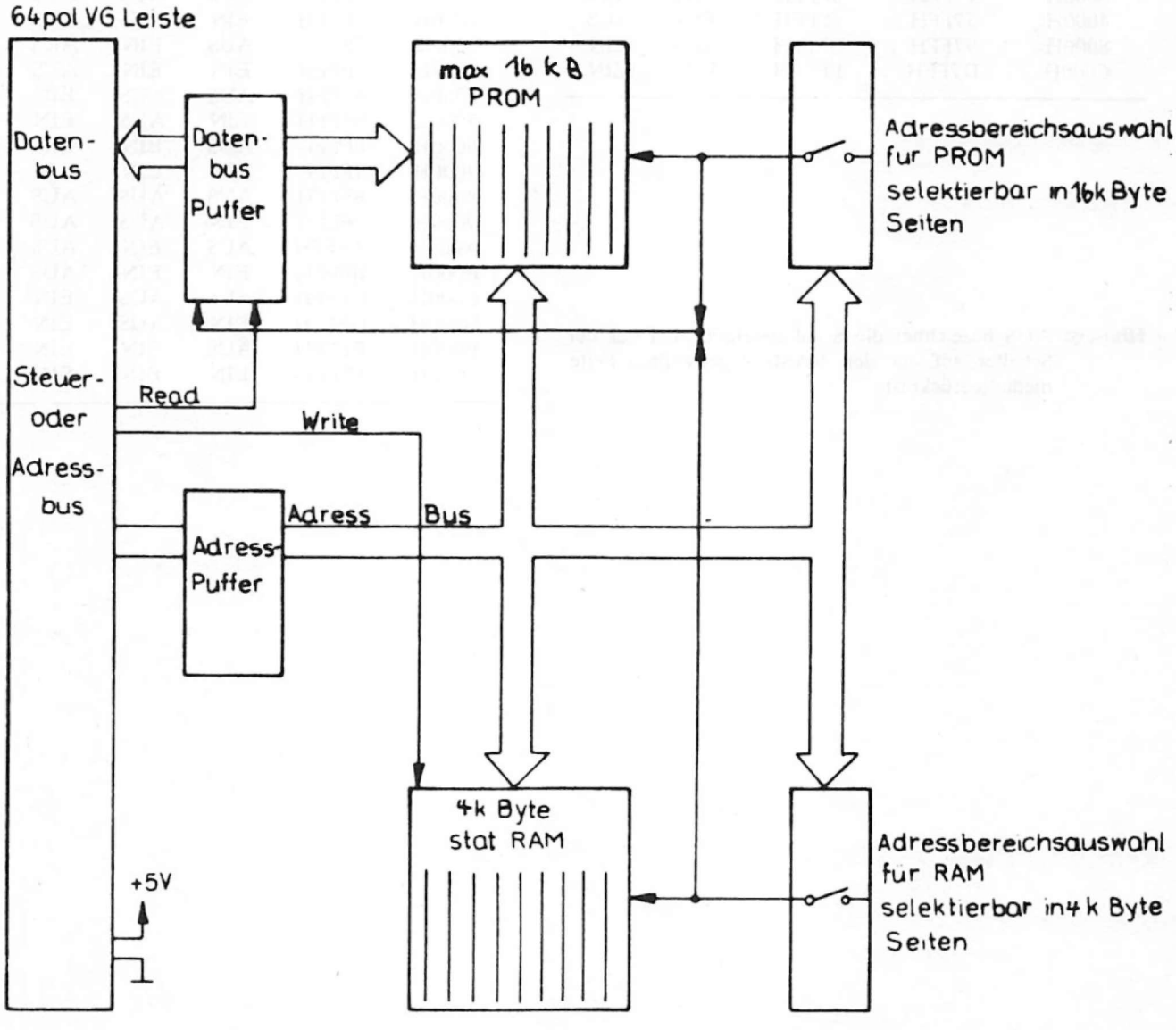
2. Schaltungsbeschreibung

2.1. Hardwareausstattung

KIT/ZBASIC enthält folgende Hardwarekomponenten:

- 6 KByte ROM (ZBASIC in insg. drei EPROMs)
- Platz für den Aufbau von weiteren max. 10 KByte ROM
- 4 KByte statisches RAM
- Einstellung der Speicheranfangsadressen von ROM und RAM in Schritten zu 16 bzw. 4 KByte
- volle Daten- und Adreßbuspufferung; Busanschlußwert = 1 TTL-LS Last.
- BUS-Anschluß über 64-poligen Stecker nach DIN 41612 (VG 95324).

2.2. Blockschaltbild



2.3. Adressenbelegung

2.3.1. Adressenbelegung Speicher

Sowohl der ROM- als auch der RAM-Bereich können mit Hilfe der vorgesehenen Kippschalter auf bestimmte Bereiche festgelegt werden. Die sich ergebenden Speicheranfangsadressen entsprechen dabei dem ganzzahligen Vielfachen der entsprechenden Speichergröße. Das 4 KByte große RAM kann somit beginnend bei

0000H
1000H
2000 H
. . . . .
. . . . .

usw. bis max. F000H

liegen. Insgesamt ergeben sich 16 verschiedene Möglichkeiten.

Der 16 KByte umfassende ROM-Speicher kann demgegenüber nur 4 verschiedene Lagen einnehmen. Die Speicheranfangsadressen lauten hier

0000H
4000H
8000H
und C000H

Bei der Wahl der entsprechenden Bereiche muß darauf geachtet werden, daß keine Doppelbelegung entsteht!

2.3.1.1. Adresseinstellung ROM

Die Auswahl des ROM-Bereiches erfolgt über den zweipoligen DIL-Schalter C.4.

ROM-Bereich			Schalter	
Anfangs-adr.	Endadr. bei 6 KB	Endadr. bei max. 16 KB	CI	C2
0000H	17FFH	3FFFH	AUS	AUS
4000H	57FFH	7FFFH	EIN	AUS
8000H	97FFH	BFFFH	AUS	EIN
C000H	D7FFH	FFFFH	EIN	EIN

Hinweis: AUS bezeichnet die Schalterstellung, bei der der Schalter auf der den RAMs zugewandten Seite niedergedrückt ist.

2.3.1.2. Adresseinstellung RAM-Bereich

Die Adresseinstellung des RAM-Bereiches erfolgt über den vierpoligen DIL-Schalter C.5.

RAM		Schalter			
Anfangs-adr.	End-adr.	C1	C2	C3	C4
0000H	0FFFH	AUS	AUS	AUS	AUS
1000H	1FFFH	EIN	AUS	AUS	AUS
2000H	2FFFH	AUS	EIN	AUS	AUS
3000H	3FFFH	EIN	EIN	AUS	AUS
4000H	4FFFH	AUS	AUS	EIN	AUS
5000H	5FFFH	EIN	AUS	EIN	AUS
6000H	6FFFH	AUS	EIN	EIN	AUS
7000H	7FFFH	EIN	EIN	EIN	AUS
8000H	8FFFH	AUS	AUS	AUS	EIN
9000H	9FFFH	EIN	AUS	AUS	EIN
A000H	AFFFH	AUS	EIN	AUS	EIN
B000H	BFFFH	EIN	EIN	AUS	EIN
C000H	CFFFH	AUS	AUS	EIN	EIN
D000H	DFFFH	EIN	AUS	EIN	EIN
E000H	EFFFH	AUS	EIN	EIN	EIN
F000H	FFFFH	EIN	EIN	EIN	EIN

2.3.2. Adressenbelegung I/O

ZBASIC verfügt über die Ansteuerroutinen einer Bedienungs- und Anzeigeeinheit. Die entsprechende Hardware (Tastatur, TV-Gerät, TV-Ansteuerung) muß vom Anwender zusätzlich implementiert werden.
Um eine möglichst große Flexibilität zu erreichen, wurde ZBASIC so ausgelegt, daß es unter folgenden Z80-KIT-Betriebsprogrammen ablaufen kann:

Z80-KIT/TV1	transp. 2 KByte für Z80-KIT/V
Z80-KIT/TV2	transp. 2 KByte für Z80-KIT/VZ
Z80-KIT/M2	transp. 1 KByte

Die beim Starten von ZBASIC durchlaufene Initialisierung stellt fest, um welches Betriebsprogramm es sich handelt und nimmt eine dementsprechende Einstellung der beiden Ports vor. Port A der ausgewählten PIO wird im INPUT-Mode betrieben und dient dem Anschluß der Tastatur, Port B des gleichen Bausteins wird als OUTPUT zur Bildschirmansteuerungseinheit benutzt.

Z80-KIT/TV1
Ist ein Z80-KIT mit einem Betriebsprogramm Z80-KIT/TV1 (Ansteuerung Z80-KIT/V) ausgestattet, so nimmt ZBASIC eine Einstellung des bereits durch Z80-KIT/TV1 benutzten PIOs auf der Zentralkarte ECB/K vor. Die dabei benutzten I/O-Adressen lauten:

PIO ECB/K

PORT A	DATA	08H
PORT A	CONTROL	0AH
PORT B	DATA	09H
PORT B	CONTROL	0BH

Für die Inbetriebnahme ist lediglich die Tastatur mit den entsprechenden PINs des Pfostensteckverbinders STB der ECB/K zu verbinden. Das Anschlußschema ist in Kapitel 2.4.3. erläutert.

Z80-KIT/TV2
Ist ein Z80-KIT mit einem Betriebsprogramm Z80-KIT/TV2 (Ansteuerung Z80-KIT/VZ) ausgestattet, so nimmt ZBASIC eine Einstellung des bereits durch Z80-KIT/TV2 benutzten PIOs auf der Bildschirmansteuerungskarte Z80-KIT/VZ vor. Die hier belegten I/O-Adressen lauten:

PIO KIT/VZ

PORT A	DATA	0FCH
PORT A	CONTROL	0FEH
PORT B	DATA	0FDH
PORT B	CONTROL	0FFH

Für den Betrieb ist es lediglich notwendig, die Tastatur mit den entsprechenden PINs des Pfostensteckverbinders STB der Z80-KIT/VZ-Platine zu verbinden. Das Anschlußschema ist in Kapitel 2.4.3. gezeigt.

Z80-KIT/M2
Wird ein Z80-KIT/M2 Betriebsprogramm benützt, so nimmt ZBASIC die gleiche Einstellung vor wie bei Vorhandensein eines Z80-KIT/TV2-Betriebsprogrammes.

Dadurch ist es möglich, auch Systeme ohne TV-Betriebsprogramm mit KIT/ZBASIC auszustatten. Als Zusatz ist dann jedoch die Bildschirmansteuerungseinheit Z80-KIT/VZ notwendig.

2.4. PIN-Belegung

2.4.1. BUS-Stecker STA

Der 64-polige Normstecker STA (DIN 41612, VG 95324) ist gemäß dem Standard des ECB-BUS belegt.

STA-Belegung

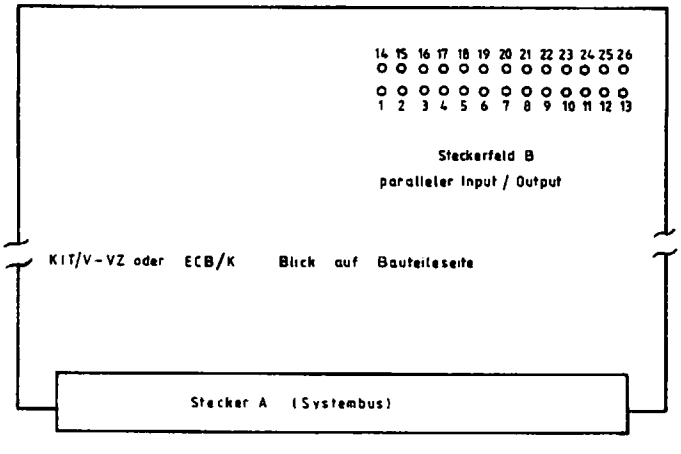
Signal	PIN	Kommentar
A0	5c	
A1	7c	
A2	6a	
A3	6c	
A4	7a	
A5	8a	
A6	9a	
A7	9c	Adreßbus
A8	8c	
A9	30a	
A10	18c	
A11	17c	
A12	27c	
A13	29a	
A14	18a	
A15	28c	
D0	2c	
D1	14c	
D2	4c	
D3	4a	Datenbus
D4	5a	
D5	2a	
D6	3a	
D7	3c	
$\overline{RD}$	24c	
$\overline{WR}$	22c	
$\overline{MREQ}$	30c	Kontrollbus
$\overline{REFR}$	28a	
+5 V	1a	
+5 V	1c	Versorgungs-
GND	32a	spannung
GND	32c	

STB-Belegung

Signal	PIN		Kommentar
A7	2	*	
A6	3	*	
A5	4	*	
A4	5	*	PORT A
A5	6	*	Daten
A2	7	*	
A1	8	*	
A0	9	*	
$\overline{ASTB}$	10	*	STROBE für PORT A
ARDY	12		READY für PORT A
B7	15		
B6	16		
B5	17		
B4	18		PORT B
B3	19		Daten
B2	20		
B1	21		
B0	22		
$\overline{BSTB}$	11		STROBE für PORT B
BRDY	25		READY für PORT B
+5 V	13	*	
+5 V	26	*	Versorgungs-
GND	1	*	spannung
GND	14	*	

2.4.2. Stecker STB

ACHTUNG: Stecker STB ist nicht auf der Platine von KIT/ZBASIC enthalten! Mit STB ist der Pfostensteckerverbinder auf der Zentralkarte ECB/K oder auf der Bildschirmansteuereinheit Z80-KIT/VZ gemeint! Beide Stecker haben eine identische Belegung:



2.4.3. Anschluß einer ASCII-Tastatur

Bei Anschluß einer ASCII-Tastatur sind die mit * versehenen Signale der obigen STB-Tabelle (PORT A !) zu verdrahten.

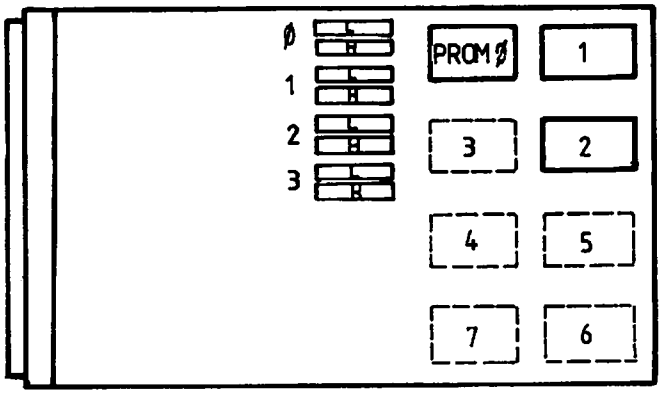
2.5. Erweiterung des ROM-Bereiches

Auf der Europakarte von KIT/ZBASIC befinden sich insgesamt acht Plätze für 24-polige Festwertspeicherbausteine. Drei davon sind durch den BASIC-Interpreter belegt. Die restlichen stehen dem Anwender für eigene Zwecke zur Verfügung.
Es können wahlweise Festwertspeicher des Typs:

	i2758	1 KByte single 5 V EPROM
oder		
	i2716	2 KByte single 5 V EPROM

oder in Anschluß und elektrischen Werten kompatible Bausteine eingesetzt werden.
Jedem Platz ist ein Adreßraum von 2 KByte zugeordnet, wobei sich die noch freien Plätze an die durch den BASIC-Interpreter belegten anschließen. Die Reihenfolge der einzelnen Plätze ist aus nachfolgendem Bild ersichtlich.

Stecker STA



Bitte beachten Sie, daß bei der Verwendung von 1 KByte-Bausteinen kein zusammenhängender Speicherbereich entsteht, da wie oben erwähnt jedem Platz 2 KByte zugeordnet sind.
Beachten Sie weiterhin, daß die WORST-CASE Zugriffszeit der EPROMs 380 nsec. nicht überschreiten darf, da sonst ein Arbeiten mit maximaler CPU-Frequenz nicht möglich ist. Die Verwendung selektierter Bausteine ist daher notwendig.
Eine zweite Möglichkeit besteht durch Einfügen eines WAIT-Zyklus mit Hilfe eines Hardwarezusatzes. Die entsprechende Schaltung findet man im CPU-Manual unter dem Titel:

„Adding a Wait State to any M1 Cycle"

3. Softwareumgebung

3.1. Initialisierung

Wie bereits unter 2.4. beschrieben, durchläuft ZBASIC nach dem Starten eine Initialisierungsroutine. Durch sie wird eine Anpassung an das vorhandene Betriebsprogramm des Z80-KIT, sowie an die vorgegebene RAM-Speichergröße für ZBASIC vorgenommen.

3.1.1. I/O-Initialisierung

Wesentlichster Teil ist die Vorbereitung der beiden Parallel-Schnittstellen für den Betrieb mit Tastatur und Bildschirm. Die in Frage kommenden Adressen entsprechen den Anforderungen der beiden Bildschirmansteuerungseinheiten Z80-KIT/V und Z80-KIT/VZ. Beide Einheiten können direkt verwendet werden. Die adressenmäßige Auswahl erfolgt in Abhängigkeit vom aufrufenden Betriebsprogramm und kann nachstehender Aufstellung entnommen werden:

Betr.prog.	Schnittst.	I/O-Adressen
Z80-KIT/TV1	PIO auf ECB/K	PORT A: 08H/ 0AH PORT B: 09H/ 0BH
Z80-KIT/TV2	PIO auf Z80-KIT/VZ	PORT A: 0FCH/0FEH PORT B: 0FDH/0FFH
Z80-KIT-M2	PIO auf Z80-KIT/VZ	PORT A: 0FCH/0FEH PORT B: 0FDH/0FFH

Neben der PORT-Auswahl werden außerdem die Einsprungpunkte für die Routinen LOAD und STORE entsprechend dem jeweiligen Betriebsprogramm festgelegt. Dadurch ist die Benutzung eines Kassettengerätes über ZBASIC in allen Fällen gesichert.

3.1.2. RAM-Initialisierung

Beim Starten von ZBASIC wird eine Abfrage des verfügbaren RAM-Speichers durchgeführt. Der im Lieferumfang enthaltene Bereich umfaßt 4 KByte, kann aber auf insgesamt 28 KByte erweitert werden. ZBASIC stellt sich automatisch auf die vorhandene Speichergröße ein.

Bedingung: zusätzlicher Speicher muß kontinuierlich an den standardmäßig gelieferten anschließen und darf keine Lücken aufweisen!

Wie im Kapitel 6.1. Standardeinstellung zu lesen sein wird, liegen RAM und ROM in folgenden Bereichen:

RAM	5000H bis 5FFFH
ROM	C000H bis D7FFH

Der Bereich dazwischen ist für den Aufbau von zusätzlichem RAM für ZBASIC geeignet. Er umfaßt folgenden Adreßraum:

RAM zusätzlich	6000H bis BFFFH

und hat einen Umfang von 24 KByte (siehe auch Kapitel 6.1.).

3.2. I/O-Treiber

Die Verbindung zwischen dem Benutzer und ZBASIC wird durch zwei in ZBASIC enthaltene I/O-Routinen hergestellt. Es handelt sich dabei um Programme, die den Datentransfer über parallele Schnittstellen abwickeln. Die Auswahl des entsprechenden Bausteins (PIO) erfolgt in Abhängigkeit vom KIT-Betriebsprogramm (siehe Kapitel 3.1.).

3.2.1. OUTPUT-Treiber

Die Ausgaben von ZBASIC entsprechen in ihrer Form den Anforderungen der Bildschirmansteuereinheiten Z80-KIT/V bzw. Z80-KIT/VZ. Das Datenformat beträgt 8 Bit. (Eine Interpretation als ASCII-Zeichen erfolgt durch die Bildschirmansteuereinheit).
Da die Bildschirmansteuereinheiten Z80-KIT/V bzw. Z80-KIT/VZ maximal 120 Zeichen pro Sekunde aufnehmen können, enthält der OUTPUT-Treiber eine automatische Zeitablaufsteuerung. Nach Ausgabe jedes Zeichens wird ein HALT-Zustand eingenommen. Hat die Bildschirmansteuereinheit das Zeichen verarbeitet, löst sie über die Handshakeleitung $\overline{BSTB}$ einen Interrupt aus, was die Fortsetzung des Ausgabeprogrammes bewirkt (siehe auch KIT/V, KIT/VZ).

3.2.2. INPUT-Treiber

Der INPUT-Treiber entspricht den Anforderungen einer interruptgesteuerten Tastatur. Das Datenformat beträgt 8 Bit. (Eine Interpretation als ASCII-Zeichen erfolgt im ZBASIC). Jeder Tastendruck der Tastatur löst unabhängig vom augenblicklichen Zustand des Programmes einen Interrupt aus. Mit Hilfe der Interrupt-Service-Routine (ISR) wird das entsprechende Zeichen in einen 1 Byte großen Zwischenspeicher gebracht, von wo es dann weiterverarbeitet wird.

3.3. Einsprungadressen von ZBASIC

ZBASIC kann unter zwei Voraussetzungen gestartet werden:

a) Im RAM-Speicher steht bereits ein ablauffähiges BASIC-Programm.
b) Es existiert noch kein BASIC-Programm; es soll eine neue, ‚leere' Datei eröffnet werden.

Abhängig von obiger Bedingung ist der jeweilige Einsprungpunkt zu wählen:

Bedingung	Einsprungadresse
Neu-Einstieg in ZBASIC Aufbau einer leeren Datei	C7B0H
Einstieg in ZBASIC mit einer bereits im Speicher stehenden BASIC-Datei	C7B3H

3.4. RAM-Bedarf von ZBASIC

ZBASIC benötigt für Zwischenspeicherungen etc. einen gewissen Teil des zur Verfügung stehenden Schreib-Lese-Speichers. Dieser Bereich darf vom Anwender nicht benützt werden!

Bereich	Zweck
3C50H bis 3C57H	Bereiche für
5000H bis 53FFH	Zwischenspeicherungen
3C60H bis 3C65H	Interrupt-Tabelle

Die von 3C60H bis 3C65H bestehende Interrupt-Tabelle kann für eigene Zwecke erweitert werden.

Der dem Anwender für eigene Programme zur Verfügung stehende Speicher umfaßt in der Standardausführung folgenden Bereich:

5400H	bis	5FFFH

Vom Anwender erstellte BASIC-Programme werden automatisch bei Adresse 5400H beginnend aufgebaut.

4. ZBASIC-Beschreibung

4.1. Übersicht

BASIC ist eine leicht zu erlernende Programmiersprache und hat eine weite Verbreitung gefunden. Sie wurde für den den interaktiven Betrieb mit dem Computer entwickelt, bei dem der Programmierer im direkten Dialog mit dem Computer steht, der ihn schon bei der Eingabe auf Syntaxfehler aufmerksam macht. Das eingegebene Programm kann sofort gestartet werden, da es nicht in Maschinensprache übersetzt wird, sondern nur Anweisungen an den Computer gibt, der es „interpretiert". Entsprechen die Ergebnisse des Programmes nicht den Vorstellungen, können beliebig Zeilen (Statements) ersetzt, gelöscht oder eingeschoben werden und danach ein neuer Versuch gestartet werden. Diese Eigenschaften verkürzen die benötigte Zeit für die Programmentwicklung ganz wesentlich.

Das im folgenden beschriebene ZBASIC-Interpreter-Programm wurde für den Z80-Mikroprozessor geschrieben. Es ist ebenso komfortabel in der Bedienung wie die für die größeren Computerbrüder geschriebenen Interpreter und steht in der Genauigkeit nicht, und in der Geschwindigkeit nur wenig seinen „großen Brüdern" nach.

Hauptbestandteile der BASIC-Programme sind im Allgemeinen numerische Berechnungen. Deshalb stellt ZBASIC eine Gleitkommaarithmetik, einen Algorithmus zur Lösung algebraischer Ausdrücke und die wichtigsten transzendentalen Funktionen zur Verfügung.

Die verwendete Syntax ist allgemein unter dem Namen Darthmouth-Basic bekannt (nach dem Namen des Ortes an dem sie entwickelt wurde). Erweitert wurde die Sprache durch Eingabe, Ausgabe- und Unterbrechungs-Befehle, die es gestatten, das vorliegende Programm zur Steuerung in Echtzeit zu benutzen („Prozess-Basic"). Damit eröffnet sich die Möglichkeit, das BASIC-Programm auch in Geräten oder Anlagen zu verwenden. Besonders dann, wenn im großen Umfang numerische Daten zu verarbeiten sind.

Werden in ZBASIC-Programmen Fehler festgestellt, können Unterbrechungspunkte gesetzt werden. Im „Direkt-Mode" lassen sich dann sehr einfach Speicherstellen lesen und evtl. verändern.

Ist das Programm ausgetestet, kann es in PROMs oder EPROMs abgespeichert werden und auch in einem anderen Gerät abgearbeitet werden. Die vom Interpreter aufgebaute Programmdatei ist adress-unabhängig! Auch mehrere BASIC-Programmteile können also unabhängig voneinander geprüft und anschließend zusammengesetzt werden, solange sich die Zeilennummern nicht überschneiden.

Soll das fertige BASIC-Programm nur noch ausgeführt werden, so sind die Editing-Funktionen, die Syntaxprüfung und der LIST-Teil des Interpreters überflüssig. Es ist nur noch ein relativ kurzer „RUNTIME" Programmteil der BASIC-Datei selbst notwendig.

Die im Interpreter verwendete Gleitkommaarithmetik stellt bei der gewählten Auflösung ein Optimum bezüglich der entwickelten Verarbeitungsgeschwindigkeit dar.

Die Vorteile der sofortigen Übersetzung des BASIC-Programms in eine neue Datei, der „quasi Compilierung", wie sie der ZBASIC-Interpreter durchführt, sollen noch einmal zusammengefaßt werden:

1. der benötigte Speicherplatz ist gering,
2. der Ablauf ist schnell und
3. das Programm ist mit einem kleinen RUNTIME-Modul lauffähig.

Zusammenstellung

1. Verwendbare Zeichen

- ☐ Einfache Variable
  ein Buchstabe von A bis Z oder
  ein Buchstabe mit einer nachfolgenden Ziffer

  z.B. D, F5, E2, A, Z, I9, W0, etc.

- ☐ Zahlen aus dem Bereich

  $\pm 1{,}7\ 10^{38}$ bis $\pm 2\ 10^{38}$

  mit sieben signifikanten Dezimalstellen
  Eingabe in beliebigem Format, mit oder
  ohne Dezimal-Exponent.

2. Implementierte Mathematik

- ☐ Addition
- ☐ Subtraktion
- ☐ Multiplikation
- ☐ Division
- ☐ Exponenzierung
- ☐ Exponenzierung zur Basis e
- ☐ Logarithmus zur Basis e
- ☐ Quadratwurzel
- ☐ SINUS
- ☐ COSINUS
- ☐ TANGENS
- ☐ Arcustangens
- ☐ Hyperbelsinus
- ☐ Absolutbetrag
- ☐ Integer
- ☐ Signum
- ☐ Zufallszahl

3. Funktionen

- ☐ AND, OR, XOR, NOT
- ☐ INPUT
- ☐ READ, DATA, RESTORE
- ☐ PRINT, TAB
- ☐ GOTO
- ☐ GOSUB . . . RETURN
- ☐ FOR . . . NEXT, STEP
- ☐ IF . . . THEN
- ☐ ON . . . THEN
- ☐ REM
- ☐ IN, OUT
- ☐ PEEK, POKE
- ☐ CALL
- ☐ ON I(nterrupt)
- ☐ END, STOP
- ☐ QUIT

4. Kommandos

- ☐ NEW
- ☐ LIST
- ☐ RUN
- ☐ CONT
- ☐ LOAD
- ☐ STORE

4.2. Eigenschaften von ZBASIC

4.2.1. Programmeingaben und Berichtigungen

Jede BASIC-Anweisung beginnt mit einer Nummer (im Beispiel 30):

```
30 LET A=5
```

Diese Nummer wird im folgenden Zeilennummer genannt.
Die Zeilennummern werden von den Programmierern üblicherweise mit 10 Stellen Abstand gewählt, damit leicht zusätzliche Zeilen eingeschoben werden können. Jede Nummer zwischen 1 und 4094 kann benutzt werden.

Der Computer ordnet die Zeilen in ihrer numerischen Reihenfolge an, gleichgültig in welcher Reihenfolge sie eingegeben wurden.
ZBASIC ignoriert Leerzeichen und benötigt keine Leerzeichen, um Teile einer Zeile zu trennen. Die folgenden Beispiele sind gleichwertig:

```
30 IF A=B+C   THEN 50
30IFA=B+CTHEN50
```

Zur besseren Lesbarkeit fügt das LIST-Programm auch fehlende Leerzeichen wieder ein.
Bei Zeichenfolgen (z. B. bei Textausgaben) wird jedes Zeichen, natürlich auch das Leerzeichen, ein- und ausgegeben.

Mit dem Zeichen CTRL-H („Backspace") können eine oder mehrere fehlerhafte Eingaben wieder gelöscht werden.

```
30  GOTO ←← SUB 80 ist gleichwertig mit
30  GOSUB   80
```

RUBOUT und NUL-Zeichen werden ignoriert. Ein RETURN-Zeichen („ret") beendet die Eingabe. Der Computer antwortet mit einem Zeilenvorschub.

Wenn bereits folgendes Programm im Computer abgespeichert ist:

```
10  INPUT A
20  LET X = A + 3
30  PRINT X
```

dann ist nach der Eingabe von

```
20  IF A = 10 THEN 40 ret  (Zeile 20 wird überschrieben)
30  ret                    (Zeile 30 wird gelöscht)
15  INPUT B                (Zeile 15 wird eingeschoben)
```

folgendes Programm vorhanden:

```
10  INPUT A
15  INPUT B
20  IF A=10 THEN 40
```

4.2.2. Variablen

Als Variablen dürfen die Buchstaben A bis Z oder ein Buchstabe mit einer nachfolgenden Ziffer verwendet werden. Beispiele:

Variable
B0, B, M6, D

4.2.3. Zahlen

Alle Zahlen werden intern mit 32 bit im Gleitkommaformat dargestellt. Das ergibt einen Zahlenbereich von etwa $\pm 1{,}7 \cdot 10^{38}$ bis $\pm 2 \cdot 10^{-38}$ mit sieben signifikanten Dezimalstellen.

Bei der Eingabe (auch INPUT oder READ) dürfen beliebig viele Vorkomma- und Nachkommadezimalstellen verwendet werden (jedoch insgesamt nicht mehr als 32). Das Dezimalexponent-Zeichen E kann vorhanden sein oder nicht und der Dezimalexponent eine oder zwei Stellen enthalten. Zahl und Exponent können mit Vorzeichen versehen sein.

Im folgenden sind erlaubte Zahleneingaben aufgeführt:

```
1
+0.00000005
-20000.003E-22
-3.21 E 3
```

Bei der Ausgabe werden Zahlen im Bereich von 0.1 bis 999999 ohne Dezimalexponent geschrieben, wobei unwichtige Nachkomma-Nullen unterdrückt werden. Zahlen außerhalb dieses Bereichs werden mit 6 Dezimalstellen zwischen 1.00000 und 9.99999 und mit Dezimalexponent dargestellt.

Zahlenformate bei Ausgabe

```
^^-^
-33333.3^                    ^ ist ein Leerzeichen
^^1.23456E±06^
```

4.2.4. Mathematische Ausdrücke

Eine Variable oder Konstante oder eine beliebige Kombination von Variablen, Konstanten und Operatoren, die sich zu einer Zahl errechnen lassen, werden im folgenden als Ausdruck („Expression") bezeichnet.

Die folgende Zeile berechnet den Wert des Ausdrucks (−A3·2)+SIN(P/2) und speichert ihn als neuen Wert der Variable Y ab:

```
40 LET Y = (-A3*2)+SIN(P/2)
```

Den Variablen A 3 und P muß vorher ein Wert zugewiesen worden sein, andernfalls erfolgt eine Fehlermeldung.

Die Zeile LET x=x+1 wird in ZBASIC Interpretiert als: Erhöhe den Wert von x um 1 und speichere ihn wieder als x ab.

Als Operatoren finden die folgenden Symbole Verwendung:

↑	Exponenzierung
*,/	Multiplikation, Division
+,−	Addition, Subtraktion

aufgeführt in Reihenfolge ihrer Priorität. **Funktionen** haben darüber hinaus die höchste Rangordnung.

Klammern dürfen an beliebiger Stelle und bis zu 8-fach ineinandergeschachtelt in einem Ausdruck verwendet werden. Es kommen bei der Berechnung ohne Einschränkung die Regeln der Algebra zur Anwendung.

Der Wert von A↑B wird von BASIC durch die Gleichung $e^{B*\log A}$ errechnet. Das kann eine kleine Ungenauigkeit in der 6. Dezimalstelle ergeben. Auch wegen der Rechengeschwindigkeit ist es deshalb besser bei ganzzahligen Exponenten die mehrfache Multiplikation anzuwenden. Bei negativer Mantisse (−A) wird geprüft, ob B ganzzahlig ist und andernfalls eine Fehlermeldung abgeben.

In einem Ausdruck werden auch folgende Funktionen errechnet:

EXP (x)	errechne e^x
LN (x)	errechne den Logarithmus des positiv gesetzten Wertes von x zur Basis e
LOG (x)	identisch mit LN (x)
SQR (x)	errechne die Quadratwurzel von x
SIN (x)	errechne die Winkelfunktion von x (mit x im Bogenmaß)
COS (x)	
TAN (x)	
ATN (x)	errechne den arctan von x (Ergebnis im Bogenmaß)
SINH (x)	errechne den hyperbel sin von x
ABS (x)	setze x positiv
INT (x)	verwende nur die Vorkommastellen von x
SGN (x)	setze x=+1 bei positiven Werten von x setze x=−1 bei negativen Werten von x setze x=0 wenn x = 0 war

Die Pseudofunktion RND (1) erzeugt eine Zufallszahl zwischen 0 und 1, wobei der Inhalt des Klammerausdrucks ohne Bedeutung ist.

4.3. Funktionen

4.3.1. AND, OR, XOR, NOT

Mit diesen Logischen („Booleschen" Operatoren werden Zahlen bitweise verknüpft. In einem mathematischen Ausdruck können diese Operatoren uneingeschränkt verwendet werden. Die benutzten Zahlenwerte werden jedoch vom Fließkomma-Format zunächst zu einer 16-bit Zahl im Bereich von −32768 bis 32767 gewandelt. Bereichsüberschreitungen führen zu einer Fehlermeldung.

```
10  LET A = OR 12                    0101 ≙  5
                                   OR 1100 ≙ 12
                                   ---------------
              → A = 13             1101 ≙ 13
```

```
20  LET B = A AND 6                 01101 ≙ 13
                                  AND 00110 ≙  6
                                  ---------------
              → B = 4              00100 ≙  4
```

```
30  LET C = NOT A          A      1101 ≙ 13
        → C = −14 NOT A   1111111111110010 ≙ −14
```

```
40  LET D = 7  XOR A                00111 ≙  7
                                  XOR 01101 ≙ 13
                                  ---------------
              →  D = 10            01010 ≙ 10
```

```
50  LET X = 5 + (7 AND B*(C OR NOT D))
```

Die Rangordnung bei der Berechnung von mathematischen Ausdrücken wird durch die Booleschen- und Vergleichs-Operatoren auf folgende Hierarchie erweitert:

1. Berechnung von Klammerausdrücken
2. Funktionen (SIN, LOG usw.)
3. ↑
4. *,/
5. +,−
6. Invert. z. B. −SIN oder −(. . . .)
7. Vergleichsoperationen (=, >, <, > = *usw.)*
8. NOT
9. AND
10. OR
11. XOR

Die bitweise Manipulation von Werten ist besonders wichtig im Zusammenhang mit Ein-Ausgabe-Operationen. Im folgenden Beispiel wird der Zustand einer Eingabeleitung (bit 3, Wertigkeit 8) abgefragt und in Abhängigkeit davon verzweigt.

```
20  IN 244, E
30  IF E AND 8 GOTO 100
```

Ist die ausgewählte Leitung 0, so wird der Ausdruck 0 und damit falsch. Nur wenn die Leitung 1 ist, erfolgt die Verzweigung.
In ähnlicher Weise können mit den booleschen Verknüpfungen ausgewählte Leitungen bei der Ausgabe gesetzt und rückgesetzt werden. Beispielsweise soll die Leitung 4 (Wertigkeit 16) auf 1 gesetzt werden, ohne den Zustand der anderen zu beeinflussen. Der augenblickliche Zustand sei unter dem Variablennamen T abgelegt.

```
50  LET T =T OR 16
60  OUT 245, T
```

Mit der Zeile 70 sollen anschließend nur die Leitungen 1 (Wertigkeit 2) und 4 (Wertigkeit 16) auf 0 gesetzt werden.

```
70  LET T = T AND NOT 2+16
80  OUT 245, T
```

Häufig werden die logischen Verknüpfungen auch bei bedingten Verzweigungen eingesetzt.

```
80  IF A>2 AND C>=2 GOTO 200
90  IF C=3 OR D=3 OR E=3 GOTO 300
```

Nur wenn beide Bedingungen erfüllt sind, erfolgt die Verzweigung zur Zeile 200 bei dem Beispiel in Zeile 80. Zur Verzweigung in Zeile 90 reicht es, wenn nur eine Bedingung erfüllt ist.

4.3.2. INPUT

Die Zeile im Beispiel weist der Variablen a und x jeweils einen über die Tastatur eingegebenen Wert zu.

```
10 INPUT, a, x
```

Wird im Programmablauf die Zeile mit einer Input-Anweisung erreicht, erscheint ein „?" auf dem Bildschirm. Das Programm erwartet Eingaben von Zahlen. Mehrere Zahlen in einer Zeile dürfen durch Komma getrennt werden. RET schließt die Zeile ab. Sind noch mehr Eingaben notwendig, meldet sich der Computer mit einem weiteren „?".
Auch während einer Eingabe kann durch ESC der Programmablauf abgebrochen werden.

4.3.3. READ, DATA, RESTORE

```
 20  READ K, L, M    weist den Variablen die Werte
                     3, 4 und 5 zu
 30  LET X=K+L+M
110  DATA 3, 4
120  DATA 5
130  PRINT X
140  RESTORE         die DATA-Werte stehen für eine
                     neue READ-Anweisung zur
                     Verfügung.
```

Nach der ersten READ-Anweisung sucht BASIC im Programm die erste DATA-Zeile, um den Wert zu lesen und holt eventuell benötigte weitere Werte in der Reihenfolge der Zuweisungen. DATA-Zeilen dürfen an beliebiger Stelle innerhalb des Programms stehen. Nach RESTORE wird mit dem folgenden READ wieder der erste DATA-Wert gelesen.

4.3.4. PRINT und TAB

```
10  PRINT  A;7
20  PRINT  B,C,
30  PRINT  D;E;
40  PRINT  „DIE SUMME IST:";
50  PRINT  SQR(X*X+Y*Y)
60  PRINT  A,TAB (A);"*"
70  PRINT
```

Zeile 10 druckt den Wert von A aus und anschließend „7", beendet mit einem Wagenrücklauf und Zeilenvorschub.
Das **Komma** erzeugt vor jeder neuen Ausgabe einen Vorschub zur nächsten Kolumne, die jeweils mit der Druckstelle 0, 13, 26, 39, 52 (und evtl. 65) beginnt (Zeile 20).
Die Druckanweisung in Zeile 30 bewirkt durch das **Semikolon**, daß die Werte von D und E und der nachfolgende Ausdruck unmittelbar aneinandergeschoben werden. Bei Zahlen bleiben jedoch die Leerzeichen nach jeder vollständigen Zahl erhalten (siehe Zahlausgabe).
Steht am Ende einer Printanweisung ein Komma oder Semikolon wie in Zeile 20 und 30, so wird der Wagenrücklauf und Zeilenvorschub unterdrückt und die nächste Ausgabe in der Zeile fortgesetzt.
Alle Zeichen, die in Printanweisungen zwischen den Anführungszeichen (") stehen, werden unverändert ausgegeben (Zeile 40).
Steht in einer Printanweisung ein mathematischer Ausdruck wie in Zeile 50, so wird erst sein Wert berechnet und dann ausgegeben.
Ein einfaches PRINT wie in Zeile 70 erzeugt einen Wagenrücklauf und Zeilenvorschub.

Die TAB-Funktion gibt soviel Leerzeichen aus, daß der Wagen an der Position stehen bleibt, die durch den nachfolgenden Ausdruck bestimmt ist. Für die graphische Ausgabe kann dann wie in Zeile 60 ein Symbol gedruckt werden. Ist der Wert größer als die Zeilenbreite oder die Position schon überschritten, so wird ein Wagenrücklauf und Zeilenvorschub erzeugt.

4.3.5. GOTO

Der unbedingte Sprung hat folgende Form:

```
 70  GOTO 200 (oder 70 GO TO 200)

200  REM AUSGABE ZWISCHENWERTE
210  PRINT A
```

Das Sprungziel muß eine vorhandene Zeile sein, andernfalls wird eine Fehlermeldung ausgegeben.

Als Sprungziel muß eine Zahl verwendet werden.

4.3.6. GOSUB . . . RETURN

Mit GOSUB-Befehl wird ohne Bedingungen eine Unterroutine aufgerufen. Um das Ende dieser Unterroutine anzuzeigen, muß eine RETURN-Zeile angefügt werden.

```
 70  GOSUB 200
      ⋮
200  PRINT A
      ⋮          } Unterroutine
250  RETURN
```

4.3.7. FOR . . . NEXT und STEP

Programmschleifen lassen sich mit der FOR . . . NEXT Anweisung besonders einfach aufbauen.

```
100  FOR  P1=2 TO 5
110  FOR  A =N-1 TO X STEP -3
120  FOR  S =0 TO (C*B)/2 STEP C/2

200  NEXT S
210  NEXT A
220  NEXT P1
```

Die Schleife wird auf alle Fälle mit dem Anfangswert (in Zeile 100 ist es 2) einmal durchlaufen. Ist die NEXT-Zeile erreicht, wird geprüft, ob die Schleifenvariablen übereinstimmen, anschließend der Wert des STEP errechnet und zur Variablen addiert. Auch negative STEPs sind zulässig. Fehlt dieser Ausdruck, wird als Schrittweite I eingesetzt.
Durch Vergleich mit dem Endwert der Schleife (in Zeile 100 ist es 5) wird festgestellt, ob die Schleife ein weiteres Mal durchlaufen werden soll oder die auf NEXT folgende Zeile abgearbeitet wird.
Als Anfangswert, Endwert und Schrittweite können auch mathematische Ausdrücke verwendet werden.
Die Schleifen dürfen sich nicht überschneiden, es können aber Schleifen innerhalb einer Schleife, wie bei dem obigen Beispiel, benutzt werden.

4.3.8. IF . . . THEN (GOTO oder GOSUB)

Bei dieser Anweisung erfolgt der Sprung nur, wenn die nachfolgende Bedingung erfüllt („wahr") ist. Auch eine einzelne Variable oder Zahl ist wahr, wenn sie nicht Null ist. Mehrere Bedingungen oder Werte lassen sich durch Boolesche Operatoren verknüpfen (siehe NOT, AND, OR).

```
10  IF K =N THEN 200
10  IF K =N GOTO 200      gleichwertig

20  IF A*C-2=SQR(A-2) GOSUB 270
```

Die beiden ersten Zeilen unterscheiden sich nur in der Schreibweise, da THEN und GOTO beide in gleicher Weise zu dem nachfolgenden Sprungziel verzweigen.
Wie in Zeile 20 kann auch eine Unterroutine bedingt aufgerufen werden.

Erlaubte Bedingungen sind:

```
=
>= (bzw. =>)
<= (bzw. =<)
>
<
<> (bzw. ><)
```

Da die Zahlen im Computer nur mit endlicher Genauigkeit verarbeitet werden, sollte die „=" Bedingungen mit Vorsicht benutzt werden und wenn möglich, durch >= und <= ersetzt werden.
Ebenso wie Zahlen können auch Ausdrücke mit den Bedingungen geprüft werden, wie in Zeile 20 des obigen Beispiels. Für eine eventuell weitere Verknüpfung wird als Ergebnis der Prüfung eine 1 gesetzt, wenn die Bedingung erfüllt ist. Auch jeder einzelne Wert ist wahr und wird 1 gesetzt, wenn er nicht Null ist.

```
30  IF A THEN 200
```

Nur wenn A=0 ist (also „falsch" ist), wird die nachfolgende Zeile ausgeführt. In allen anderen Fällen erfolgt die Verzweigung zu Zeile 200

In einer IF-Zeile dürfen die Bedingungen auch mit den Booleschen Operatoren weiter verknüpft werden.

```
10  IF A=1 OR B=1 THEN 200
20  IF A=1 AND B=1 THEN 200
30  IF B>A OR NOT(5-C) AND SQR(C)<2 THEN 200
```

Wahrheitstabelle der Booleschen Operatoren:

A	0	0	1	1	"1" ≙ "wahr"
B	0	1	0	1	
A AND B	0	0	0	1	
A O R B	0	1	1	1	
NOT A	1	1	0	0	

Die Rangordnung bei der Berechnung von Booleschen Ausdrücken ist:

1 Berechnung von arithm. Ausdrücken
2 Prüfung der Bedingungen =><
3 NOT
4 AND
5 OR

4.3.9. ON . . . THEN (GOTO oder GOSUB)

Soll in Abhängigkeit von einem Wert zu einem von mehreren verschiedenen Sprungzielen verzweigt werden, oder eine von mehreren Unterroutinen aufgerufen werden, so wird eine ON-Zeile verwendet.

```
10  ON A THEN 200, 350, 400
20  ON C-B GOSUB   300, 450
```

Hat A in Zeile 10 den Wert 1, springt das Programm zu Zeile 200. Ist der Wert 2, dann zu Zeile 350 und bei 3 zu Zeile 400. Wenn A einen Wert von <1 oder> 3,9999 hat, wird das Programm mit der nachfolgenden Zeile 20 fortgesetzt. Der Wert der Variablen oder des Ausdrucks wird mit der INT-Funktion ganzzahlig gesetzt, bevor die Verzweigung geprüft wird.
In gleicher Weise wird bei Zeile 20 die Unterroutine 300 aufgerufen, wenn C-B einen Wert von 1 hat.

```
30  ON 2+SGN(X) GOTO 100, 200, 300
```

Bei dieser Zeile wird bei negativem X zu Zeile 100 und bei positivem X zu Zeile 300 verzweigt. Ist X=0 wird das Programm mit Zeile 200 fortgesetzt.

4.3.10. REM

Um Kommentare und Erklärungen in einem Programm einzufügen, wird die REM-Zeile benutzt.

```
100  REM DAS IST EINE REM-ZEILE!
```

Während des Programmablaufs wird sie vollständig ignoriert. Sie belegt jedoch Speicherplatz; deshalb sollten bei Speicher-Kapazitätsproblemen die Verwendung dieser Kommentare gering gehalten werden.

4.3.11. IN, OUT

Die Aufgabe dieser Befehle ist es, eine Verbindung zwischen dem ZBASIC Interpreter und seiner „Außenwelt" herzustellen; also Ein- und Ausgaben zu ermöglichen, die nicht über die Console laufen, sondern z.B. über Ein-/Ausgabekanäle, die mit Z80-PIO, Z80-SIO usw. realisiert sind und direkt allgemeine Prozeßperipherie ansteuern.
Zunächst müssen diese Ein-Ausgabe-Tore adressiert werden. Die Zahl 128 in Zeile 10 im Beispiel wird als Adresse (80H in Hexadezimalschreibweise) aufgefaßt und auf den Adreßbus ausgesandt. Anschließend wird der Wert 7 übergeben.

```
10  OUT 128, 7
20  OUT 248, (B+16)/2
30  IN 246, D
```

Wie in Zeile 20 kann auch der Wert einer Variablen oder eines mathematischen Ausdrucks von einem Output-Port übernommen werden. Die Zahl muß jedoch kleiner als 255 sein, andernfalls wird eine Fehlermeldung abgegeben. Als Adresse muß immer eine Dezimalzahl zwischen 0 und 255 verwendet werden.
Umgekehrt wird beispielsweise mit der Zeile 30 der an dem Eingabebaustein mit der Adresse 246 anliegende Wert der Variablen zugeordnet. Das Bitmuster an der Schnittstelle wird als Zahl zwischen 0 und 255 interpretiert und unter einem Variablennamen abgespeichert.
Ein-Ausgabe-Bausteine müssen zunächst initialisiert (in einen bestimmten Modus gebracht) werden oder es müssen Zustandsregister abgefragt werden. Mit den IN- bzw. OUT-Anweisungen ist auch dies in einfacher Weise möglich. In diesem Zusammenhang sind die Booleschen Operatoren zur bitweisen Manipulation besonders wichtig. Siehe AND, OR, XOR und NOT.

4.3.12. PEEK, POKE

Der direkte Zugriff zu Speicherplätzen wird mit den Befehlen PEEK und POKE möglich.

```
30  PEEK   50000, L
40  PEEK   50000 + 1, M
50  POKE   50100, A/10
```

Mit Zeile 30 wird der Inhalt des Speicherplatzes 50000 (0C350H in Hexadezimalschreibweise) geholt und der Variablen L zugewiesen. Der 8-bit Wert unter dieser Adresse wird als Zahl zwischen 0 und 255 interpretiert. Als Adresse kann auch ein mathematischer Ausdruck verwendet werden wie in Zeile 40. Überschreitungen des Zahlenbereichs von 0 bis 65535 werden mit einer Fehlermeldung beantwortet.
Umgekehrt kann mit POKE ein beliebiger Speicherplatz mit einem Wert belegt werden. Mit Zeile 50 im Beispiel wird zunächst der Wert des Ausdrucks A/10 errechnet und anschließend auf den Speicherplatz 50100 abgespeichert.
Mit dem anschließenden Beispiel soll demonstriert werden, daß z. B. auch eine Subroutine in Maschinensprache mit BASIC aufgebaut werden kann, die anschließend mit CALL 50200, B; B gerufen wird. Sie soll mit dem Maschinenbefehl DAD H die Zahl verdoppeln und besteht aus folgenden Befehlen:

Adresse	Befehl	Mnemonischer Code
50200 D	225 D	POP H
50201 D	41 D	DAD H ; verdoppeln
50202 D	229 D	PUSH H
50203 D	195, 0, 100 D	JMP 100 D ; Rücksprung

Mit den folgenden BASIC-Befehlen wird diese Routine in den Speicherbereich übertragen:

```
100  DATA 225, 41, 229, 195,0, 100
110  FOR N = 50200 TO 50205
120  READ X
130  POKE N, X
140  NEXT N


300  CALL 50200, B; B
```

Eine weitere wichtige Aufgabe der Befehle PEEK und POKE ist die Ein- und Ausgabe von Informationen bei der Anwendung der „memory mapped I/O" Technik. Bei ihr werden Ein-Ausgabeschnittstellen wie Speicherplätze adressiert und Daten übergeben oder übernommen. Ob diese Technik verwendet wird oder mit IN-OUT-Befehlen die Ein-/Ausgabe-Schnittstellen angesprochen werden müssen, ist durch die Hardware festgelegt.

4.3.13. CALL

Ein CALL-Befehl erlaubt den Aufruf eines Unterprogramms in Maschinensprache.

```
200  CALL 50200
250  CALL N + 10
```

Die Zahl nach CALL wird als Adresse des Unterprogramms in Dezimalschreibweise aufgefaßt. Auch ein mathematischer Ausdruck kann als Adresse verwendet werden. Überschreitungen des erlaubten Zahlenbereichs von 0 bis 65535 führen zu einer Fehlermeldung.

Zur Übergabe von Werten wird in beiden Richtungen der Stack benutzt. Die entsprechenden Variablen werden beim Aufruf (CALL) direkt angegeben. Dabei werden Werte, die an das Maschinenprogramm übergeben werden sollen, mit je einem führenden Komma an die Einsprungadresse angefügt. Werte die vom Maschinenprogramm zurückgegeben werden sollen, folgen mit je einem Semikolon versehen.

```
300  CALL 50200, A,B;X;Y
```

Das Komma informiert den Interpreter, daß der Wert der nachfolgenden Variablen auf den Stack gepusht werden soll. Dazu wird der Wert zu einer 16-bit Zahl im Bereich von −32768 bis 32767 gewandelt; im Beispiel wird die Variable A und dann die Variable B im Stack abgelegt.
Das Maschinenprogramm holt sich die Werte aus dem Stapelspeicher ("= Stack"), im Beispiel zuerst B und verarbeitet sie.

Der Stack darf bis zu einer Tiefe von 20 Byte benutzt werden.

Die Routine in Maschinensprache endet nicht mit RET sondern mit JP:

JP C7B6H C3B6C7 (im Maschinencode)

Nach Rückkehr aus dem Maschinenprogramm holt der Interpreter mittels POP-Befehl pro angegebener Variable (Kennzeichen: Semikolon) einen Wert aus dem Stack und legt ihn unter dem angegebenen Variablennamen ab. Im Beispiel wird der erste Wert der Variablen X der zweite Wert der Variablen Y zugewiesen.
Wurden fehlerhafte Stackoperationen durchgeführt (Anzahl der PUSH- und POP-Operationen stimmen nicht überein) wird eine Fehlermeldung abgegeben.

4.3.14. ON I . . . GOSUB, GOTO, RETURN

Der Interpreter kann während des Programmablaufs durch einen Hardware-Interrupt dazu veranlaßt werden, einen bestimmten Programmteil aufzuführen und anschließend im normalen Ablauf fortzufahren. Der Buchstabe I ist in der ON-Zeile als Pseudovariable reserviert, um eine Unterbrechung zu ermöglichen oder den Interrupt auszuschließen.

```
100  ON I GOSUB 1000
120  PRINT A,B
130  ON I RETURN
```

Die Zeile 100 löst nicht sofort eine Aktion aus, sondern informiert den Interpreter nur, daß er ab jetzt im Falle eines Interrupts die Unterroutine ab Zeile 1000 abarbeiten soll. Diese Unterbrechungsmöglichkeit wird mit der Zeile 130 zurückgenommen; ab dieser Zeile werden Interrupts ignoriert.

```
1000  IN 246, P
1010  ON P GOSUB 1300, 1400, 1500
1020  RETURN
```

Als Beispiel einer Routine, die bei Interrupt ausgeführt werden kann, wird in Zeile 1010 in Abhängigkeit von einer Eingangsgröße weiter verzweigt. – Jede mit GOSUB gerufene Interruptroutine muß wie in Zeile 1020 mit RETURN abschließen.
Mit dem folgenden Beispiel ist es möglich, den Programmablauf erst nach einem erfolgten Interrupt fortzusetzen:

```
800  ON I GOTO 1000
```

Der Interpreter wartet in der Zeile 800 bis ein Interrupt erfolgt und springt dann zum Programm ab Zeile 1000. Auch durch Drücken der ESC-Taste kann diese Warteschleife durchbrochen werden.
Um im Rahmen von KIT/ZBASIC eine möglichst große Zahl von Einsatzmöglichkeiten zu bieten, ist die Interruptsteuerung nicht an einen bestimmten PORT gebunden.
Für einen Einsatz mit Interruptsteuerung sind folgende I/O-Kanäle geeignet:

a) paralleler I/O-Port (PIO); sämtliche Betriebsarten
b) Zähler/Zeitgeber-Kanal (CTC).

In allen Fällen ist die Wahl der PORT-Adresse beliebig und kann entsprechend den hardwaremäßigen Gegebenheiten festgelegt werden.
Um eine einwandfreie Verarbeitung externer Interrupts sicherzustellen, ist es notwendig, die Interruptfähigkeit des entsprechenden PORTs von ZBASIC aus beeinflussen zu können. Diese Steuerung erfolgt mit Hilfe der entsprechenden Anweisungen an den jeweiligen PORT. Dazu notwendig ist jedoch

a) die Adresse des verwendeten PORTs
b) die Steueranweisung zur Beeinflussung der Interruptfähigkeit des PORTs (z. B. PIO: 03H/83H).

Da, wie oben erwähnt, Adresse und Art des PORTs nicht festliegen, muß die entsprechende Information vom Anwender an ZBASIC übergeben werden. Dies geschieht durch

a) Ablegen der PORT-Adressen in Speicherzelle 3C56H
b) Ablegen des Interrupt-Steuerwortes in Speicherzelle 3C57H.

Beim Interrupt-Steuerwort ist der Zustand des entsprechenden BITs (= BIT D7) ohne Belang, da ZBASIC dieses BIT selbst nach Bedarf setzt oder rücksetzt.
Neben den bisher genannten Vorbereitungen ist eine weitere Maßnahme erforderlich. Da im Interrupt-Mode 2 gearbeitet wird, ist für den gewählten PORT der niederwertige Teil des Interruptvektors anzugeben. Dieser ‚Vektorteil' entspricht dem niederwertigen Teil der entsprechenden Adresse der Interrupt-Tabelle und hat den Wert:

64H (siehe Kapitel 3.4.)

Dieser Wert ist in das I-Register des gewählten I/O-PORTs zu laden.
Selbstverständlich ist zusätzlich die übrige Initialisierung des PORTs (Mode-Wahl etc.) vorzunehmen.
Die gesamte Initialisierung kann entweder durch den Aufruf eines Assembler-Programmes (siehe CALL, Kapitel 4.7.3) erfolgen, oder unter Einsatz der BASIC-Befehle IN, OUT, PEEK und POKE (siehe Kapitel 4.3.11. und 4.3.12.).

4.3.15. END und STOP

Die letzte Zeile in jedem Programm muß eine END-Zeile sein. Wird sie während des Programmablaufs erreicht, beendet der Interpreter seine Arbeit und wartet auf neue Eingaben. Auch ohne diese Zeile wird nach der Ausführung der letzten (höchsten) Zeile der Ablauf beendet.

```
200  IF B>0 AND D>0 THEN 210
205  STOP
210  PRINT X
220  END
- - - - - - - - - -
250  REM SUBROUTINE
240  LET X=B ↑ C+D ↑ E
250  RETURN
```

Die END-Zeile muß gesetzt sein, wenn das Programm nicht an der letzten Zeile auch logisch zu Ende ist. Im Beispiel folgt auf das Ende noch eine Subroutine von Zeile 230 bis 250.

Eine typische Anwendung der STOP-Zeile ist die Einfügung nach einer Prüfung, die bei negativem Ergebnis auf einen Fehler aufmerksam macht. Wird sie erreicht, meldet sich der Computer mit der Mitteilung

STOP IN LINE 205

und wartet auf neue Eingaben. Auf diese Weise kann ein Fehler leicht eingekreist werden.

4.3.16. QUIT

Durch das Kommando

QUIT

erfolgt ein Rücksprung in das Betriebsprogramm des Z80-KIT. Der Einsprung erfolgt bei 0000H.
Ein ab Adresse 5400H im RAM stehendes BASIC-Programm bleibt vollständig erhalten und kann nach erneutem Einsprung in ZBASIC wiederverwendet werden (siehe Kapitel 3.3. Einsprungadressen von ZBASIC).

4.4. Kommandos

Kommandos sind in diesem Zusammenhang definiert als Anweisungen an das Betriebssystem, wie das Programm behandelt werden soll. Sie sind also nicht Bestandteil des Programms.

4.4.1. NEW

Bevor ein neues Programm eingegeben wird, ist mit dem NEW-Kommando ein eventuell vorhandenes altes Programm zu löschen. NEW wird automatisch ausgeführt, wenn ein Einsprung in ZBASIC mit Startadresse C7B0H erfolgt (siehe 3.3.).

4.4.2. LIST

Ist ein Programm geladen, kann es mit dem LIST-Kommando auf der Console aufgelistet werden.

NEW 10LETP1=4 ATN(1) 20IFA BGOSUB 100	Eingabe
LIST 10 LET P1=4 ATN(1) 20 IF A B GOSUB 100 READY	Antwort des Computers

Im Beispiel wurde Zeile 10 und 20 als neues Programm eingegeben und anschließend mit LIST abgerufen. Der ZBASIC-Interpreter schreibt das Programm und meldet sich anschließend mit READY als Mitteilung, daß er neue Eingaben erwartet.
Bei der Eingabe werden Leerzeichen ignoriert, bei der Ausgabe nach LIST jedoch an den Stellen eingefügt, an denen es für bessere Lesbarkeit notwendig ist.
(Die Zeile 10 weist übrigens der Variablen P1 den Wert $\pi = 3.14159$ zu.)
Soll ein Programm nicht vollständig, sondern erst ab Zeile 210 z.B. gelistet werden, so wird das mit folgender Zeile erreicht:

```
210 LIST
```

Eine LIST-Ausgabe kann jederzeit durch Drücken der ESC-Taste abgebrochen werden. Der Interpreter meldet sich mit STOP IN LINE XXXX. Sollte die Ausgabe damit nur unterbrochen werden, um z.B. ein Detail des Programms näher zu analysieren, so kann anschließend mit XXXX LIST das Listing fortgesetzt werden.

4.4.3. RUN, CONT

Das Programm wird mit dem RUN-Kommando gestartet. Bevor das Programm jedoch abgearbeitet wird, setzt der Interpreter alle von ihm benutzten Speicherplätze auf ihren Anfangswert.
Wird während des Programmablaufs ein Fehler erkannt, wird eine Fehlermeldung ausgedruckt. Alle Fehler, außer dem numerischen Überlauf (ERROR 0), beenden den Programmablauf und lassen den Interpreter auf neue Eingaben warten.
Nach einem vollständigen Durchlauf meldet sich der Computer mit READY. Wird jedoch eine STOP-Zeile erreicht, oder der Ablauf durch Drücken der ESC-Taste abgebrochen, wird als Meldung STOP IN LINE NNN ausgedruckt.
Soll ein Programm nicht mit der niedrigsten Zeilenzahl oder ein anderes, oberhalb liegendes Programm gestartet werden, so wird dafür das folgende Kommando benutzt:

```
200 RUN
```

Der Programmablauf beginnt mit der Zeile 200. Wird das Programm mit CONT gestartet, so werden im Gegensatz zum RUN-Kommando alle intern benutzten Speicher nicht verändert. Sinnvoll ist dieses Kommando, wenn z.B. die Werte der Variablen eines Programms für ein zweites Programm übernommen werden sollen. Doch besonders wichtig ist das CONT-Kommando, um ein unterbrochenes Programm fortzusetzen. Hat sich der Interpreter mit STOP IN LINE NNN noch ESC oder einer STOP-Zeile oder auch mit einer ERROR X IN LINE NNN Ausgabe gemeldet, kann mit folgendem Kommando der Programmablauf an beliebiger Stelle fortgesetzt werden (Beispiel mit Zeile 300):

```
300 CONT
```

Es ist natürlich Aufgabe des Programmierers, hierbei nur plausible Eingriffe vorzunehmen; z.B. keine FOR-NEXT-Schleifen zu verlassen oder anzuspringen.
Anmerkung: Es ist jedoch nicht möglich, eine Zeile durch eine Eingabe zu verändern und anschließend das Programm fortzusetzen. In diesem Fall gibt es undefinierte Fehler.
Eine besondere Eigenschaft des ZBASIC-Interpreters ist die Möglichkeit, auch nach dem im folgenden beschriebenen Direkt-Modus das Programm mit NNN CONT fortzusetzen.

4.4.4. LOAD

Mit dem Kommando LOAD kann ein auf Kassette befindliches BASIC-Programm in den Speicher zurückgeholt werden. ZBASIC bedient sich dabei der im KIT-Betriebsprogramm enthaltenen Routine.
Nach Aufruf des Kommandos erscheint auf dem Schirm die Frage:

TAPE OK?

(Das Kommando kann an dieser Stelle durch ESC rückgängig gemacht werden.)
Stellen Sie nun Ihr Bandgerät auf Wiedergabe und starten Sie es so, daß der bei STORE aufgenommene Anfangsdauerton wiedergegeben wird. Drücken Sie dann die Taste RETURN. Damit beginnt der Ladevorgang, dessen Ende durch ein

READY

auf dem Bildschirm angezeigt wird. Schalten Sie jetzt Ihr Kassettengerät ab. Bei Auftreten eines Übertragungsfehlers (ERROR Lampe am KIT brennt) ist der Ladevorgang zu wiederholen. Das Programm wird grundsätzlich Adresse 5400H beginnend geladen.

4.4.5. STORE

Mit dem Kommando STORE kann ein BASIC-Programm auf Kassette abgespeichert werden. ZBASIC bedient sich dabei der im KIT-Betriebsprogramm enthaltenen Routine.
Nach Aufruf des Kommandos erscheint auf dem Schirm die Frage:

TAPE OK?

(Das Kommando kann an dieser Stelle durch ESC rückgängig gemacht werden.). Schalten Sie nun Ihr Bandgerät auf Aufnahme, regeln Sie die Aussteuerung ein (bzw. Automatik benutzen) und lassen Sie das Gerät laufen. Nach ca. 30 Sekunden (Anfangsdauerton) drücken Sie die Taste RETURN. Damit beginnt der Abspeichervorgang, dessen Ende durch ein

READY

auf dem Bildschirm angezeigt wird. Schalten Sie jetzt Ihr Kassettengerät ab. Die Abspeicherung beginnt grundsätzlich bei Adresse 5400H.

4.5. DIREKT-MODUS

Wird ein BASIC-Befehl (Statement) ohne Zeilennummer eingegeben, z. B.

```
PRINT A,B,C
```

so antwortet der Computer sofort, indem er die Werte von A, B und C ausdruckt. Bei der Fehlersuche ist das eine sehr hilfreiche Eigenschaft. Ebenso können auch direkt Werte von Variablen verändert werden, z. B. mit

```
LET A=7
INPUT B,C
```

Nach diesen Tests oder Eingriffen kann das Programm mit NNN CONT fortgesetzt werden.
Der Direkt-Modus erlaubt auch eine effektive Benutzung des Interpreters als Tischrechner.

Bei Eingabe von
PRINT SQR(3*3+4*4)
wird als Antwort ausgedruckt:
5

Alle bei BASIC erlaubten Ausdrücke werden bei dieser Anweisung errechnet und deren Zahlenwert ausgedruckt.
Dies setzt natürlich ein Programm voraus, in dem diese Variablen verwendet werden, d. h. diese Variablen sind zum Zeitpunkt des Befehls „bekannt" und haben einen definierten Wert. Ist diese Voraussetzung nicht erfüllt, erfolgt eine Fehlermeldung.

4.6. Fehlercode

B Zuviele GOSUBs oder FOR . . . NEXT-Schleifen ineinander geschachtelt
C kein TO in FOR-Zeile oder Schleife unterbrochen
E zu viele verschiedene Variablen verwendet
G GOSUB ohne RETURN oder FOR ohne NEXT verwendet
H Zeilenzahl ist größer als 4094
I unbekannte Zahl nach GOSUB, bzw. THEN oder ON (oder auch Zeilenzahl >4094)
J FOR-NEXT-Schleife hat unterschiedliche Variablen (vielleicht weil sich Schleifen überschneiden)
K Klammerfehler weil) und (nicht im Gleichgewicht sind oder mehr als 8 Klammern ineinander geschachtelt wurden
L Kein Kommando oder Basic-Anwendung am Zeilenanfang
M Speicherüberlauf durch Eingabe eines zu umfangreichen Programms oder bei RUN durch zu viele Variablen
N undefinierte Variable verwendet
O mathematischer Zahlenüberlauf (nur Warnung; das Programm rechnet mit dem größtmöglichen Wert weiter)
Q bei M↑E ist Mantisse negativ und Exponent nicht ganzzahlig
R RETURN ohne GOSUB oder NEXT ohne FOR verwendet
S Syntaxfehler
T Stack nach Aufruf von Routinen mit CALL nicht in Ordnung
V unbekannter Befehl oder Operator
W zu wenig Zahlen mit DATA definiert
X Zahlenbereichsüberschreitung bei Umwandlung in 16-bit Zahlenwert
Z Es fehlt die Klammer nach der Funktion.

4.7. Rechenzeiten

Die angegebenen Zeiten sind Mittelwert je Rechenoperation im Gleitkommaformat. Je nach Zahlengröße können Abweichungen bis etwa ±20% auftreten.

Addition	2 msec
Multiplikation	3,6 msec
Division	5,2 msec
SIN, COS	36,8 msec
TAN	78,4 msec
ATN	72 msec
LOG	32 msec
SQR	20,8 msec
EXP	56 msec
SINH	60 msec
X↑Y	88 msec
ABS, SGN	0,4 msec
INT	0,8 msec

Die Zeiten gelten für die Z 80-CPU mit 2.5 MHz Taktfrequenz.

5. Zusammenbau

ACHTUNG!

Bei den gelieferten ICs handelt es sich in einigen Fällen um MOS-Bausteine, die trotz ihrer integrierten Schutzstrukturen mit Vosicht gehandhabt werden sollen!
Bitte nehmen Sie die Bausteine erst bei Bedarf aus der leitenden Verpackung!

5.1. Stückliste zu KIT/ZBASIC

Stand 08. 02. 1979

Lfd.-Nr.	Bezeichnung	Name	Anzahl
1	Leiterplatte ECB/E16 Nr.: 119		1
2	Messerleiste 64-polig VG 95324 incl. Befestigung	STA	1
3	Federleiste 64-polig lötbar VG		1
4	IC-Sockel 24-polig	G.1 G.2 E.1 E.2 C.1 C.2 A.1 A.2	8
5	IC-Sockel 18-polig	G.3 G.4 F.3 F.4 E.3 E.4 E.3 D.4	8
6	Umschalter 2 × UM 74C02	C.4	1
7	Umschalter 4 × UM 74C04	C.5	1
8	Widerstand 4,7 K	RI	1
9	Kondensator ELKO 4,7 μ/16 V	C1–C15	15
10	Diode 1N4002	D1	1
11	IC 74LS08	G.5	1
12	IC 74LS14	F.5	1
13	IC 74LS30	D.5	1
14	IC 74LS18	C.3	1
15	IC 74LS139	E.5	1
16	IC 74LS174	B.3 B.4	2
17	IC 74LS175	B.5	1
18	IC 74LS245	A.4	1
19	IC 2114 RAM statisch 1 K × 4	G.3 G.4 F.3 F.4 E.3 E.4 D.3 D.4	8
20	IC i2716 oder i2758 bzw. vergleichbarer Typ EPROM 2 K × 8 single 5 V, progr. mit ZBASIC	G.1 G.2 E.1	2
21	Dokumentation		1

5.2. Aufbau der Platine

Bitte beachten Sie beim Aufbau von KIT/ZBASIC die Hinweise die im Kapitel 5.5. von Teil I des Z80-KIT-Anwenderhandbuches niedergelegt sind!
Der Zusammenbau ist in mehrere Schritte unterteilt, um eine möglichst große Sicherheit zu erreichen. Bitte stecken Sie die einzelnen Teile jeweils in die vorgesehenen Plätze und vergleichen ihre Lage mit dem Bestückungsplan. Erst nach dieser Kontrolle soll mit dem Einlöten begonnen werden. Der Bestückungsplan liegt der Verpackung bei.

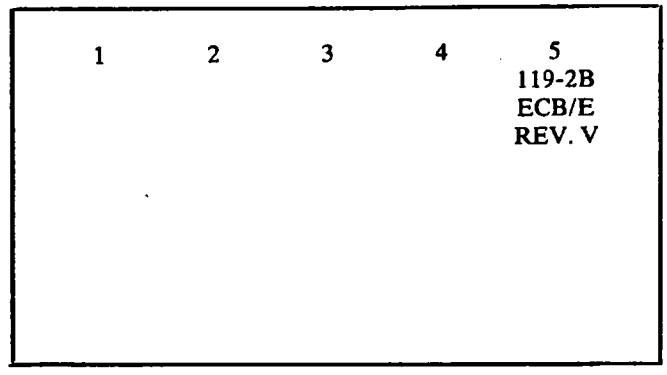
Hinweis: Bauteilebezeichnungen, die das Format

Buchstabe . Zahl

besitzen, sind gleichzeitig eine Koordinatenangabe bezüglich der Lage dieses Bauteils auf der Platine. Die Koordinaten sind am Rande der Platine aufgebracht.

Schritt 1:

Nehmen Sie die Platine (160 × 100 mm) aus der Verpackung und legen sie so, daß am oberen Rand die Ziffern 1 bis 5 von links nach rechts in aufsteigender Reihenfolge zu lesen sind.



In der rechten oberen Ecke muß dabei die im obigen Bild dargestellte Bezeichnung stehen.
Die jetzt sichtbare Seite ist die Bestückungsseite.

Schritt 2:

Nehmen Sie jetzt die Kondensatoren C1 bis C15 (ELKOs 4,7 μ/16 V)und löten sie sie an den entsprechenden Stellen ein. Beachten Sie dabei unbedingt die Polarität der Bausteine. Sowohl im Bestückungsplan als auch auf der Platine ist jeweils mindestens ein Anschluß entsprechend mit + oder – gekennzeichnet.

Schritt 3:

Löten Sie nun den Widerstand R1 (4,7 K) und die Diode D1 (1N4002) in ihre vorbestimmten Plätze. Bitte kontrollieren Sie die Lage der Diode an Hand des Bestückungsplanes. Eine falsch gepolte Diode führt zu einem Kurzschluß der Versorgungsspannung und zur Zerstörung der Diode selbst.

Schritt 4:

Setzen Sie jetzt die acht 18-poligen IC-Fassungen in die Platine ein. Wichtig ist die Lage der Markierung von PIN 1. Sie besteht aus einer kleinen Abschrägung bzw. Nase im Innenteil der Fassung. Stellen Sie sicher, daß diese Markierung in Übereinstimmung mit dem Bestückungsplan zu liegen kommt. Im Plan ist die Lage der Markierung mit einem kleinen Pfeil kenntlich gemacht.
Löten Sie anschließend diese Fassungen ein. Dabei empfiehlt es sich, erst zwei an den Ecken diagonal gegenüberliegenden Anschlüsse zu befestigen. Dadurch kann der Sitz der Fassung leicht korrigiert werden. Erst wenn ein gleichmäßig ‚satter' Sitz erreicht ist, sollten die restlichen PINs verlötet werden.

Schritt 5:

Verfahren Sie nun mit den acht 24-poligen Fassungen wie im vorigen Schritt. Achten Sie auch hier auf die Lage von PIN 1!

Schritt 6:

Nehmen Sie nun die ICs, deren Bezeichnung laut Stückliste mit 74LS beginnt der Reihe nach zur Hand und setzen sie an der entsprechenden Stelle in die Platine ein. Auch hier ist größte Sorgfalt bezüglich der Lage von PIN 1 geboten! Im Bestückungsplan ist PIN 1 durch eine 1, auf der Platine durch ein kurzes Stück Leiterbahn gekennzeichnet.

Hinweis: Alle ICs, ob auf Fassung oder nicht, haben eine einheitliche Ausrichtung.

Löten Sie die Bausteine nur nach eingehender Prüfung ein!

Hinweis: Sollten Sie keine oder nur geringe Erfahrung mit Lötarbeiten besitzen, empfiehlt es sich, alle ICs auf Sockel zu setzen! Eine evtl. Fehlersuche wird dadurch wesentlich erleichtert! IC-Sockel sind in jedem Elektronik-Geschäft oder in entsprechenden Versandfirmen erhältlich. Ein großes Angebot derartiger Firmen findet man in jeder Elektronikzeitschrift.

Schritt 7:

Löten Sie jetzt die beiden Umschalter C.4 (2-polig) und C.5 (4-polig) in die vorgesehenen Plätze.

******* ACHTUNG! ********

Bei diesen beiden Bauteilen liegt PIN 1 entgegen der übrigen Ausrichtung. Im Bestückungsplan ist dies durch eine 1 an entsprechender Stelle angedeutet. Auf den Schaltern ist PIN 1 durch einen Punkt gekennzeichnet.

Schritt 8:

Als letztes Teil ist nun noch die 64-polige Steckerleiste einzulöten. Spannen Sie dazu Stecker und Platine möglichst in eine entsprechende Vorrichtung ein, um einen festen Sitz zu erreichen.
Sollte kein Steckplatz Ihres Z80-KIT frei sein, kann mit der beigelegten 64-poligen Buchsenleiste ein neuer Steckplatz geschaffen werden.

Schritt 9:

Nehmen Sie jetzt die acht 18-poligen RAM-Bausteine (2114) und stecken Sie sie vorsichtig und unter Beachtung der Lage von PIN 1 in die Fassungen.
Anschließend verfahren Sie in gleicher Weise mit den drei 24-poligen ROM-Bausteinen. Beachten Sie dabei folgende Zuordnung:

ROM-Baustein	Steckplatz
Nr. 1	G.2
Nr. 2	G.1
Nr. 3	E.1

Damit ist Ihre Erweiterung KIT/ZBASIC komplett aufgebaut.

6. Inbetriebnahme

Nachdem Sie Ihre Erweiterung KIT/TBASIC nun komplett aufgebaut haben, steht der Inbetriebnahme nichts mehr im Wege.

6.1. Standard-Einstellung (Adressen)

Für den Betrieb des BASIC-Interpreters ist es notwendig, beiden Speicherbereichen (RAM und ROM) eine definierte Lage innerhalb des maximal adressierbaren Speichers von 64 KByte zuzuordnen.
Folgende Einstellung ist bindend vorgeschrieben:

	Speicheranf. adresse	Schalterstellung			
		C1	C2	C3	C4
ROM	C000H	EIN	EIN	–	–
RAM	5000H	EIN	AUS	EIN	AUS

Durch diese Einstellung ergibt sich folgende Speicherbelegung Ihres Z80-KIT:

Speicher-bereich	Größe	belegt durch
0000H 07FFH	2 KB	Betriebsprog. M2 oder TV
0800H 0FFFH	2 KB	evtl. PDT
1000H 3BFFH	11 KB	frei
3C00H 3CFFH	¼ KB	Z80-KIT RAM
4000H 4FFFH	4 KB	frei
5000H 5FFFH	4 KB	KIT/ZBASIC RAM
6000H BFFFH	24 KB	frei
C000H D7FFH	6 KB	ZBASIC-Interpreter
D800H FFFFH	10 KB	frei

6.2. Wahl des Steckplatzes

Wie schon unter 3.2. beschrieben, verwenden die beiden I/O-Treiber eine Interruptsteuerung. Interruptfähig ist der jeweils vom BASIC-Interpreter bestimmte Parallelschnittstellenbaustein (PIO). Dieser Baustein hat einerseits auf Grund seiner Lage auf der Platine, und andererseits auf Grund der Lage dieser Platine im Gesamtsystem (BUS!) eine definierte Stellung innerhalb der Z80-Prioritätskette (dasy-chain).
Sollen außer den beiden genannten noch weitere Interruptebenen zugelassen werden, ist vom Anwender zu entscheiden, ob diese eine höhere oder geringere Priorität erhalten sollen.
Die für die Prioritätskette notwendige Verdrahtung ist im BUS des Z80-KIT bereits realisiert. Steckplatz 5 hat dabei die höchste Wertigkeit. Die restlichen Plätze schließen sich in der Reihenfolge 4 3 2 und 1 an.

Hinweis: Wird ein Steckplatz nicht verwendet, so muß zur Aufrechterhaltung der Kette eine Zusatzverbindung eingefügt werden! Dies kann durch Aufbringen von etwas Lötzinn auf zwei nebeneinanderliegende Leiterbahnhalbkreise erreicht werden. Diese Verbindungsstelle liegt zwischen den Anschlußreihen der entsprechenden Buchsenleiste, auf der Vorderseite der Grundplatine (ECB/M) und ist für diesen Zweck vorgesehen.

6.3. Starten von ZBASIC

Setzen Sie dazu den Befehlszähler (PC) auf einen der beiden möglichen Einsprungadressen und drücken Sie anschließend die Taste START.

Bedingung	Einsprungsadresse
Neu-Einstieg in ZBASIC Aufbau einer leeren Datei	C7B0H
Einstieg in ZBASIC mit einer bereits im Speicher stehenden BASIC-Datei	C7B3H

Als Reaktion wird der angeschlossene Bildschirm gelöscht, in der obersten Zeile erscheint die Meldung

'READY'

und darunter der, als blinkender Strich dargestellte Cursor. ZBASIC ist damit betriebsbereit.

6.4. Fehlersuchhinweise

Für den Fall, daß Ihr ZBASIC in der angegebenen Weise reagiert, schalten Sie bitte den Ştrom ab und führen folgende Kontrollen aus:

- sind RAM- und ROM-Bereich richtig gewählt?
- steht der normale Z80-KIT RAM-Bereich (3C00H bis 3CFFH) vollständig zur Verfügung?
- ist die Interrupt-Prioritäts-Kette richtig aufgebaut (Wahl des Steckplatzes, evtl. Störung durch höherwertige, falsch oder nicht initialisierte I/O-Bausteine)?
- sind alle RAM-Bausteine (2114) richtig eingesetzt?
- ist die Lage der drei ROM-Bausteine korrekt?
- ist bei einem der auf Fassung sitzenden Bausteine ein PIN geknickt, oder neben der Fassung?
- sind alle übrigen ICs in ihrer Einbaulage korrekt?
- sind die Schalter C.4 und C.5 richtig eingebaut (Ausrichtung entgegen übriger Orientierung!)?
- haben alle Kondensatoren richtige Polarität?
- ist die Diode D1 richtig gepolt?
- sind alle Lötungen einwandfrei (kalte Lötstellen!)?
- bestehen evtl. Lötkurzschlüsse (z. B. zwischen benachbarten Leiterbahnen)?

Sollten diese Punkte keinen Fehler erkennen lassen, ist es ratsam zu versuchen, ob eine Benutzung des KIT/ZBASIC-RAMs (Adresse 5000H bis 5FFFH) möglich ist. Dazu kann ein beliebiger Bereich über den INPUT-Mode des Z80-KIT mit willkürlichen Kombinationen belegt werden. Der anschließende Lesevorgang (IDM) muß bei intaktem Speicher die eingegebenen Kombinationen ergeben. Im gegenteiligen Fall ist die obige Liste erneut durchzugehen!
Sollten Sie hingegen erfolgreich sein, empfiehlt es sich, den in Teil II, KIT/P, Kapitel 7 stehenden RAM-Test durchzuführen. Evtl. hier auftretende Meldungen können Aufschluß über die Lage der Art des Fehlers geben.
Sollte der RAM-Test keine Beanstandung ergeben, kann versucht werden, den Inhalt der ROM-Bausteine = ZBASIC-Interpreter zu lesen. Da KIT/ZBASIC ohne Listing ausgeliefert wird, sind im Anschluß einige Auszüge angegeben, die verifiziert werden können.

Ergibt sich eine Diskrepanz zwischen Ihren PROM-Inhalten und dem oben Gedruckten, so ist die Checkliste erneut durchzugehen. Im anderen Fall ist ein erneuter Startversuch, evtl. auf einem anderen Steckplatz im Z80-KIT zu unternehmen. Drücken Sie vor einem erneuten Startversuch in jedem Fall mehrmals die RESET-Taste!

Adresse	Inhalt															
C000	C6	00	5F	7A	CE	00	57	79	CE	00	C3	59	C2	D6	00	7D
C010	DE	00	6F	7C	DE	00	67	7B	DE	00	5F	3E	00	C9	C6	00
C020	6F	7C	CE	00	67	7B	CE	00	5F	3E	00	C3	C2	C2	00	2E
C030	2F	26	C0	5E	26	50	73	2D	F8	18	F6	73	2C	77	2C	70
C040	2C	71	2C	72	C9	26	50	2E	30	AF	77	C9	3E	80	0E	AF
C050	21	31	50	A6	EE	80	77	26	50	2E	30	7E	A7	28	E6	5F
C060	2C	7E	2C	AE	2C	4E	2C	56	C3	68	C1	7E	A7	28	D6	5F
C070	2C	7E	2C	4E	2C	56	6F	F6	80	47	AD	21	30	50	CD	3B
C080	C0	A8	47	F6	01	7B	C9	7E	A7	C4	81	C1	28	B7	38	2F
C090	CD	34	C2	78	A7	FA	A2	C0	2E	30	7E	DE	01	77	C8	CD
C0A0	A7	C1	CD	17	C2	38	18	47	F6	01	7B	C9	AF	96	FE	01
C0B0	D4	81	C1	38	0A	28	90	4F	CD	74	C2	26	50	38	E3	21
C0C0	2E	50	3E	FF	77	07	C9	00	3E	80	0E	AF	5E	2C	AE	47
C0D0	2C	4E	2C	56	21	30	50	7E	2D	77	7B	A7	CA	59	C0	68
C0E0	78	F6	80	47	AD	2E	31	AE	2E	35	77	2E	30	7E	A7	CA
C0F0	74	C1	93	38	0A	FA	59	C0	FE	19	38	21	C3	59	C0	F2

7. Programmbeispiele

7.1. Sinus-, Cosinus- und Tangens-Berechnung

Das unten gezeigte Programm ist dazu geeignet, die Werte der drei Winkelfunktionen über einen beliebigen Wertebereich zu berechnen und auszugeben. Durch den Befehl INPUT werden die Parameter

erster Wert
letzter Wert
und Schrittweite

eingegeben.
Die Eingabe muß dabei im Bogenmaß erfolgen! Als Ausgabe erhält man das Argument (in Grad), sowie die entsprechenden Winkelfunktionswerte in Tabellenform.

Programm:

```
10 INPUT X,Y,Z
20 LET P=3.1415927
30 FOR I=X*P TO Y*P STEP Z*P
40 LET A=SIN(I)
50 LET B=COS(I)
60 LET C=TAN(I)
70 LET D=I*(180/P)
80 PRINT K,A,B,C
90 NEXT I
```

Soll dem Ausdruck der Werte eine Überschrift hinzugefügt werden, so kann man wie folgt Zwischenzeilen einbauen:

```
12 PRINT "I","SIN(I)","COS(I)","TAN(I)"
14 PRINT "-----------------------------------------------"
```

Die Zeilennummern 12 und 14 sind frei aus dem ‚Vorrat' zwischen 10 und 20 gewählt.
Als Gesamtprogramm ergibt sich:

```
10 INPUT X,Y,Z
12 PRINT "I","SIN(I)","COS(I)","TAN(I)"
14 PRINT "-----------------------------------------------"
20 LET P=3.1415927
30 FOR I=X*P TO Y*P STEP Z*P
40 LET A=SIN(I)
50 LET B=COS(I)
60 LET C=TAN(I)
70 LET D=I*(180/P)
80 PRINT K,A,B,C
90 NEXT I
```

Beispiel:
Das obige Programm wird mit den Werten

Y = 0 → 0 Grad Anfangsargument
Y = 2 → 360 Grad Endargument
Z = .25 → 45 Grad Schrittweite

durchlaufen. Es ergeben sich die SINUS-, COSINUS- und TANGENS-Werte über den Bereich 0 . . . 360 Grad im Abstand von 45 Grad.