

Realisierung einer Speicher-  
erweiterung in einem Klein-  
rechnersystem und Implemen-  
tierung einer Strichgraphik

# I n h a l t

	Seite
1. Einleitung	1
 <u>Teil I</u>	
2. Das bestehende System	3
2.1 Das Video-Genie	3
2.2 Das CP/M-Betriebssystem	4
2.3 CP/M von OMIKRON	6
3. Aufgabenstellung	8
4. Problemlösung	9
4.1 Banking-Verfahren	9
4.2 Einschaltvorgang	12
5. Hardware	13
5.1 Beschreibung sämtlicher Signale	13
5.2 Schaltungsbeschreibung	14
5.2.1 Eingriffe in Grundeinheit	14
5.2.1.1 1. Block abschalten	14
5.2.1.2 NMI-Taster	15
5.2.2 Eingriffe in Expander	16
5.2.2.1 FDC-Adressen verlegen	16
5.2.2.2 Expansion-RAM2 nicht enablen	16
5.2.2.3 Datenbus für zusätzlichen Speicher enablen	17
5.2.3 Zusätzliche Platine	18
5.2.3.1 Blockschaltbild	19
5.2.3.2 Das Signal $\overline{\text{CPM}}$	20
5.2.3.3 Kalt- und Warmstart	20
5.2.3.4 Verlegung der FDC-Adressen	21
5.2.3.5 Expansion-RAM2 nicht enablen	22
5.2.3.6 Ports	22
5.2.3.7 RAM-/Eprom-Ansteuerung	24
5.2.3.8 Datenbus für zusätzlichen Speicher enablen	26
5.2.3.9 Eprom und zusätzlicher Speicher	26
5.2.3.10 Stromversorgung	27
5.3. Praktische Ausführung	29

	Seite
6. Software	30
6.1 Speicherbelegung der Zeropage	30
6.2 Beschreibung der Systemprogramme	31
6.2.1 Systemstart durch Eprom und Cold-Start-Loader	31
6.2.2 Unterprogramme des Eproms (CBIOS)	33
6.2.2.1 CMDFDC	33
6.2.2.2 DRVSEL	33
6.2.2.3 WRPTXT	33
6.2.2.4 PREAD	34
6.2.2.5 PWRITE	34
6.2.2.6 WRPROT	35
6.2.2.7 PCONOU	35
6.2.2.8 ESCAPE	36
6.2.2.9 PCONST	36
6.2.2.10 PCONIN	37
6.2.2.11 INKEY	37
6.2.2.12 DSPLMS	38
6.2.2.13 PREADY	38
6.2.3 BIOS	39
6.3. Beschreibung der geänderten transienten Programme	40
6.3.1 SETUP63.COM	40
6.3.2 MFORM63.COM; LFORM63.COM	41

## Teil II

7. Strichgraphik	43
7.1 Zeichendarstellung auf dem Bildschirm	43
7.2 Strichgraphik-Darstellung auf dem Bildschirm	44
7.3 Blockschaltbild	45
7.4 Hardware	47
7.4.1 Datenbuslatch	47
7.4.2 Schaltzeichendecoder	48
7.4.3 Zeilendecoder (horizontaler Strich)	49
7.4.4 Zeichendecoder (vertikale Striche)	49
7.4.5 Spaltendecoder	50
7.4.6 Flipflops	51
7.4.6.1 Zeichengeneratorumschaltung	51
7.4.6.2 Horizontaler Strich	52
7.4.7 Mischer	52

	Seite
7.5 Software	53
7.5.1 Ausgabe auf Bildschirm	53
7.5.2 Ausgabe auf Drucker	54
7.6 Benutzungsbeispiel	55
8. Zusammenfassung	56
9. Literaturverzeichnis	57

### Anhänge

Original-Video-Genie mit Änderungen	
- Schaltplan CPU-Platine (Blatt 1-3)	59
- Schaltplan Interface-Platine (Blatt 1-2)	62
- Schaltplan Expander-Platine (Blatt 1-4)	64
- Layout CPU-Platine	68
- Layout Interface-Platine	69
- Layout Expander-Platine	70
Speichererweiterung	
- Schaltplan Speichererweiterung (Blatt 1-3)	71
- Layout Speichererweiterung	74
- Pinbelegung Steckerleiste (Speichererweiterung)	75
Strichgraphik	
- Schaltplan Strichgraphik	77
- Layout Strichgraphik	78
- Anschlußbelegung Strichgraphik-Platine	79
Listings sämtlicher Programme	
- Cold-Start-Loader	80
- BIOS	83
- CBIOS	97
- SETUP63.COM	109
- MFORM63.COM	119
- LFORM63.COM	128

## 1. Einleitung

Die Umrüstung eines Home-Computers auf das CP/M-Betriebssystem unter Ausnutzung des gesamten Speicherbereichs ist Thema des ersten Teils dieser Arbeit.

Als Grundlage dienten das Home-Computer-System Video-Genie EG 3003, das von sich aus nur mit einem Basic-Interpreter ausgestattet ist, sowie eine Umrüstschaltung (OMIKRON-Mapper), die Software und Hardware für das CP/M-Betriebssystem enthält, jedoch nicht den gesamten verfügbaren Speicher ausnutzt.

Dem Entwurf der Hardware ging eine Analyse des Home-Computers voraus. Es mußten die Stellen der Schaltung gefunden werden, die einen Umbau mit möglichst wenig Änderungen an der Original-Platine ermöglichten. Anschließend wurde die Software der Umrüstschaltung disassembliert und analysiert, um sie der neuen Hardware anpassen zu können. Die nächsten Abschnitte der Arbeit waren der Aufbau der zusätzlichen Platine, sowie die Änderungen der transienten Programme, die in dem neuen System nicht mehr lauffähig gewesen wären.

Alle geänderten Programme wurden mit Kommentar versehen, um evtl. spätere Änderungen möglichst schnell und bequem vornehmen zu können.

Der zweite Teil der Arbeit besteht aus der Implementierung einer Strichgraphik. Es sollte die Möglichkeit geschaffen werden, auf dem Bildschirm einzelne Wörter unterstreichen bzw. einrahmen zu können, um damit das Schriftbild übersichtlicher zu gestalten.

Zunächst mußte das Video-Interface analysiert und eine Schaltung zur Generierung der Graphik entworfen werden.

Schließlich war aber noch eine Änderung der Software (Video-Treiber) erforderlich, um die Graphik-Steuerzeichen überhaupt auf den Bildschirm durchzuschalten.

T e i l I

2. Das bestehende System

2.1 Das Video-Genie

Es handelt sich um einen Home-Computer der Bezeichnung "Video-Genie EG 3003", der überwiegend baugleich mit dem TRS-80 von Tandy ist. In der ausgebauten Version besteht er aus Grundeinheit, Erweiterung (Expander/Expansion), Bildschirm, Drucker und bis zu vier Diskettenlaufwerken.

In der Grundeinheit befinden sich die CPU, 12kbyte ROM (Basic-Interpreter), Bildschirm- und Tastaturspeicher sowie 16kbyte RAM. Der Expander enthält weitere 32kbyte RAM und die Ansteuerung für Drucker und Diskettenlaufwerke. Die Z-80-CPU des Systems kann 64kbyte Speicher direkt adressieren. Dieser Speicherbereich ist im Video-Genie wie folgt belegt:

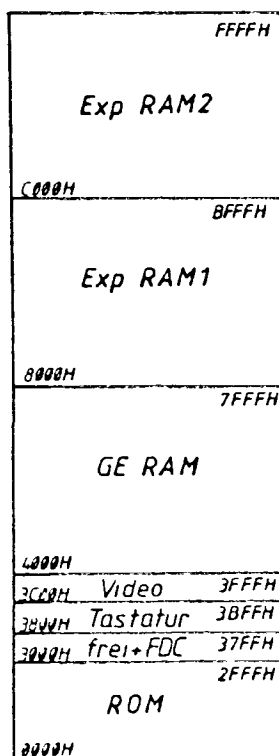


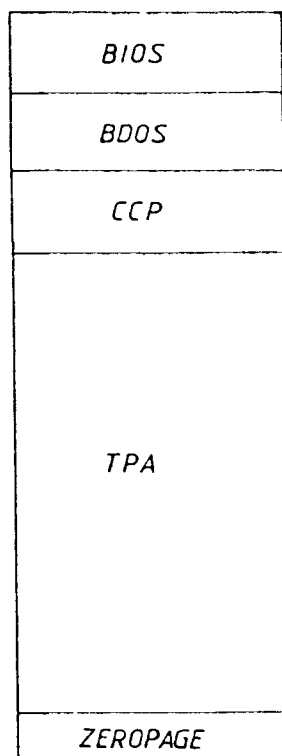
Fig. 2.1

## 2.2 Das CP/M-Betriebssystem

Das CP/M ist ein Betriebssystem zur Steuerung und Verwaltung von Files auf Disketten. Das System selbst enthält nur einige Befehle zum Anzeigen, Löschen und Abspeichern von Files. Es lassen sich jedoch andere Programme (sog. COM-Files) laden und starten, die dann den Befehlssatz erweitern (Files können kopiert, ausgedruckt, geändert usw. werden). Ferner gibt es zum CP/M Assembler, Texteditor sowie Pascal-, Fortran- und Cobol-Compiler. Diese sogenannten transienten Programme sind völlig hardwareunabhängig und benötigen nur einen (hier Z-80-) Rechner, auf dem CP/M lauffähig ist.

CP/M ist aufgeteilt in:

- BIOS: Basic I/O-System  
hardwareabhängiges Ein-/Ausgabesystem
- BDOS: Basic Disc Operating System  
Disketten-Steuersystem
- CCP: Console Command Processor  
Console-Steuersystem
- TPA: Transient Program Area  
Speicher für auszuführendes Programm
- Zeropage: Programmsprünge ins CP/M; vom CP/M benutzte Daten  
Vorstehende Programmteile befinden sich wie folgt im Speicher:



— Beginn des Speichers  
(Adr 0000H)



Auch hier sind alle Segmente mit Ausnahme des BIOS hardwareunabhängig und laufen auf jedem Z-80- bzw. 8080-Rechner.

Die auszuführenden Programme (wie z.B. Fortran-Compiler, Assembler, Texteditor) werden jeweils in die TPA geladen und benutzen die CP/M-Routinen durch Einsprünge über die Zeropage. So sind sie unabhängig von der im System vorhandenen Speichergröße und dem vorhandenen Mikroprozessorsystem. Demzufolge gibt es inzwischen eine beträchtliche Anzahl an CP/M-Programmen.

### 2.3 CP/M von OMIKRON

Wie man aus Fig. 2.1 und Fig. 2.2 erkennt, ist das Video-Genie von sich aus nicht für CP/M geeignet, da von Adresse 0000H an das Basic-ROM liegt, während das CP/M dort einen RAM-Bereich verlangt. Zudem befinden sich die Tastatur- und Bildschirmadressen mitten im adressierbaren Bereich.

Die Firma OMIKRON bietet für den TRS-80 und damit auch für das Video-Genie einen sog. "OMIKRON-Mapper" an: Ein Teil (der eigentliche Mapper) wird in die Grundeinheit eingebaut und rechnet hardwaremäßig die von der CPU erzeugten Adressen so um, daß der gesamte Speicherbereich um einen Block von 4000H nach unten verschoben wird. Dadurch liegt das RAM ab Adresse 0000H; ROM, Tastatur und Bildschirm liegen im letzten Block und somit am Ende des adressierbaren Speicherbereichs.

Die Speicherverteilung ergibt sich damit zu:

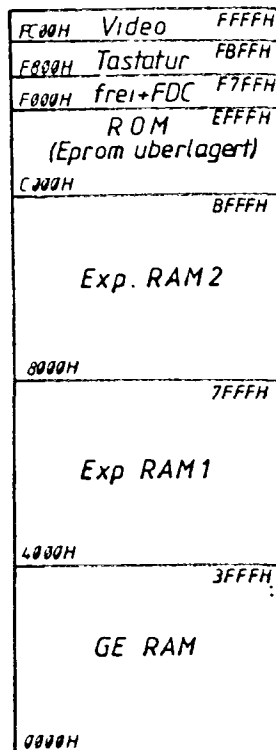


Fig. 2.3

Der Nachteil dieses "Mappers" ist, daß nur 3/4 des gesamten adressierbaren Speicherbereichs für das CP/M zur Verfügung stehen (48k-CP/M-Version).

Der zweite Teil (der Mapper II) wird in den Expander eingebaut und ermöglicht die Verwendung zweier verschiedener Laufwerkarten:

- die vom Video-Genie benutzten Mini-Laufwerke und
- die vom Standard-CP/M benutzten 8"-Laufwerke.

Ohne Einbau des Mappers II kann auch das CP/M nur Mini-Laufwerke ansteuern.

Im Rahmen dieser Arbeit soll nur mit Mini-Disketten gearbeitet werden. Die Software jedoch ist für 8"-Laufwerke vorbereitet und nach Einbau des Mappers II sind diese auch anschließbar.

### 3. Aufgabenstellung

Das Video-Genie soll nun so auf CP/M umgerüstet werden, daß nachstehende Forderungen erfüllt sind:

- Speicher so groß wie möglich
- Wahl zwischen TRS-80-Mode und CP/M-Mode mittels Umschalter
- Möglichkeit eines CP/M-Warmstarts mittels Taster
- Übernahme der Software für 8"-Laufwerke von OMIKRON
- Möglichst Übernahme aller weiteren Funktionen von OMIKRON
- Möglichst wenig Eingriffe in das bestehende System

## 4. Problemlösung

### 4.1 Banking-Verfahren

Den größtmöglichen freien Speicher erreicht man durch das "Banking-Verfahren": In der ersten Bank (das ist der normal adressierte Speicherbereich) liegen 64kbyte RAM. Alle anderen Adressen (Tastatur, Bildschirm und Floppy-Disc-Controller (FDC)) werden in die zweite Bank verlegt und nur dann angesprochen, wenn (über ein Port) ein Flipflop eingesetzt wurde, welches den entsprechenden Teil des RAMs abschaltet. Das bedeutet also, daß jedesmal, wenn z.B. ein Zeichen auf den Bildschirm geschrieben werden soll, von der ersten auf die zweite Bank umgeschaltet, dann das Zeichen auf den Video-Speicher gegeben, und anschließend wieder in die erste Bank zurückgeschaltet wird. Dabei darf das Umschalten natürlich nur aus dem Bereich erfolgen, der nicht abgeschaltet wird.

Durch das Einfügen der Umschaltbefehle dauern die Ein- und Ausgabe-Routinen geringfügig länger als ohne Banking. Jedoch reicht diese Verzögerung schon aus, um keinen sicheren Datentransfer zwischen Speicher und Diskette mehr zu ermöglichen. Daraus folgt, daß die FDC-Adressen in der ersten Bank verbleiben müssen.

Das Video-Genie besitzt keine intelligenten Ein-/Ausgabeeinheiten. So muß z.B. die CPU die Tastaturmatrix abfragen, entscheiden, welche Taste gedrückt ist und diese entprellen. Auch die Ausgabe auf den Bildschirm geschieht durch Schreiben eines Zeichens auf den entsprechenden Speicherplatz im RAM des Videospeichers. Die dafür benötigten Routinen beanspruchen mehr Speicherraum als im CP/M für das BIOS vorgesehen ist. Sie gehören zum hardwareabhängigen Teil des CP/M und müssen als Ein-/Ausga-

beroutinen in der ersten Bank verbleiben. Deshalb teilt sich das Video-Genie-BIOS in das eigentliche BIOS (1kbyte); das sich an der vorgesehenen Stelle im CP/M befindet und von Diskette eingelesen wird und das CBIOS (1kbyte), das in einem Eprom am Ende des Speichers liegt. Dort befinden sich auch die Adressen des Floppy-Disc-Controllers. Insgesamt geht 1kbyte verloren und die freie Speichergröße ergibt sich zu 63kbyte (63k-Version).

Nach dem Einschalten ist der Program-Counter der CPU zurückgesetzt und erwartet ab Adresse 0000H ein Programm. Dieses findet er in einem Eprom, das nach einem Kaltstart auf Adresse 0000H liegt und anschließend an das Ende des Speichers weggeschaltet wird und dort als CBIOS verbleibt.

Somit wurde im CP/M-Mode folgende Speicherverteilung gewählt:

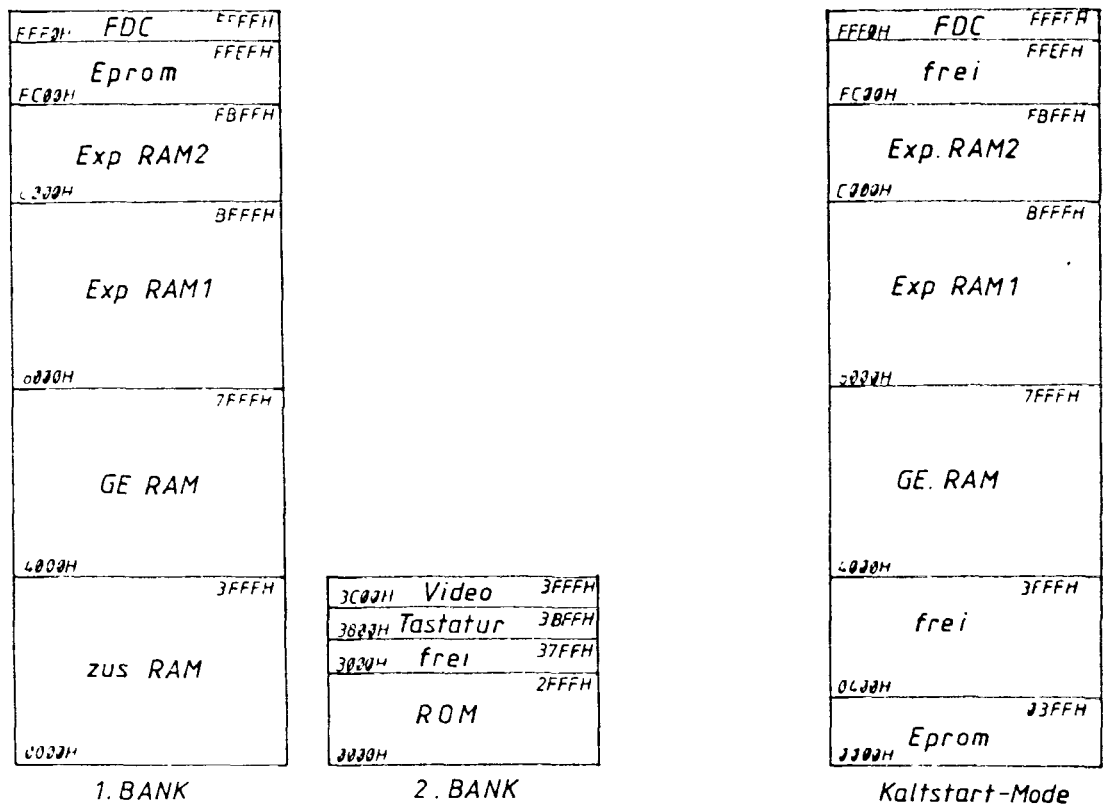


Fig. 4.1

Die einzelnen Teile des CP/M befinden sich dann wie folgt im Speicher:

FFE0H	FDC	F4FFH
FC00H	CBIOS	FFEFH
F600H	BIOS	F8FFH
E800H	BDOS	F5FFH
E000H	CCP	E7FFH
		DFFFH
TPA		
D100H		
0000H	Zeropage	00FFH

Fig. 4.2

#### 4.2 Einschaltvorgang

Das Einschalten des Systems im CP/M-Mode spielt sich demnach folgendermaßen ab:

- Verlegen des Eproms nach Adresse 0000H
- Sprung nach 0000H
- Laden der Umschaltroutine ins RAM  
(nach Adresse F000H)
- Sprung zur Umschaltroutine:  
Eprom nach FCXXH (X: beliebig)  
RAM ab 0000H
- Sprung ins Eprom
- Laden des Cold-Start-Loaders ab Adresse 0000H
- Sprung zum Cold-Start Loader
- Einlesen des CP/M-Systems
- Sprung zum BIOS und damit Ausführung des  
eigentlichen CP/M

Ein Kaltstart mittels des NMI-Tasters legt ebenfalls das Eprom ab Adresse 0000H. Der Einsprung erfolgt dann bei Adresse 0066H im Eprom, wo die gleiche Routine liegt, die auch beim Einschalten durchlaufen wird.



## 5. Hardware

### 5.1 Beschreibung sämtlicher Signale

- $\overline{\text{CPM}}$ : System im CP/M-Mode
- $\overline{\text{EPROM}}$ : Adressen FC00H...FFFFH  
(A10·A11·A12·A13·A14·A15)
- $\overline{\text{COM}}$ : Adressen FFF0H...FFFFH  
(A4·A5·A6·A7·A8·A9·A10·A11·A12·A13·A14·A15)
- $\overline{\text{EPR0}}$ : Eprom liegt auf Adressen 0000H...03FFH
- $\overline{\text{MMRQ}}$ : modifiziertes  $\overline{\text{MREQ}}$ :  
identisch mit  $\overline{\text{MREQ}}$ , es sei denn,  $\overline{\text{MEMDIS}}$  ist aktiv
- $\overline{\text{CS}}$ : Eprom enablen (Chip-Select)
- $\overline{\text{RAMSEL}}$ : zusätzliches RAM enablen
- $\overline{\text{DBUS}}$ : Datenbus des Eproms bzw. zusätzlichen RAMs wird  
auf den System-Datenbus durchgeschaltet
- $\overline{\text{BANKE}}$ : 2. Bank ist eingeschaltet (Adressen 0000H...3FFFH  
in der Grundeinheit)  
Ist im TRS-80-Mode immer eingeschaltet
- $\overline{\text{MEMDIS}}$ : Memory zu Prüfzwecken disablen
- $\overline{\text{37EX}}$ : com-Adressen (FDC,Drucker) adressiert  
( $\overline{\text{37EX}}$  hier ist identisch mit 37EX im Service-  
Manual)
- $\overline{\text{PRESET}}$ : setzt Prozessor auf 0000H zurück
- $\overline{\text{PR}}$ : setzt alles in Kaltstart-Mode
- $\overline{\text{RCS}}$ : zusätzliches RAM bzw. Eprom angesprochen (ohne  $\overline{\text{RD}}$ )

## 5.2 Schaltungsbeschreibung

### 5.2.1 Eingriffe in Grundeinheit

In diesem Abschnitt werden die Änderungen beschrieben, die an der Grundeinheit vorgenommen werden müssen.

#### 5.2.1.1 1. Block abschalten

In der Grundeinheit muß der gesamte erste Block (ROM, Videospeicher, Tastatur usw.) abschaltbar sein und darf nur als zweite Bank adressiert werden können: Dieser Block ist dann angesprochen, wenn an Z25 der Grundeinheit der Pin 12 auf logisch "0" liegt. Auf der Zusatzplatine wird ein Signal  $\overline{\text{BANKE}}$  generiert, das logisch "0" ist, wenn das System im TRS-80-Mode ist oder im CP/M-Mode die zweite Bank eingeschaltet ist.

Damit gilt:

$$\overline{\text{BLOCK1}'} = \overline{\text{BLOCK1}} \cdot \overline{\text{BANKE}} \quad (\text{BLOCK1}' = \text{BLOCK1} + \text{BANKE})$$

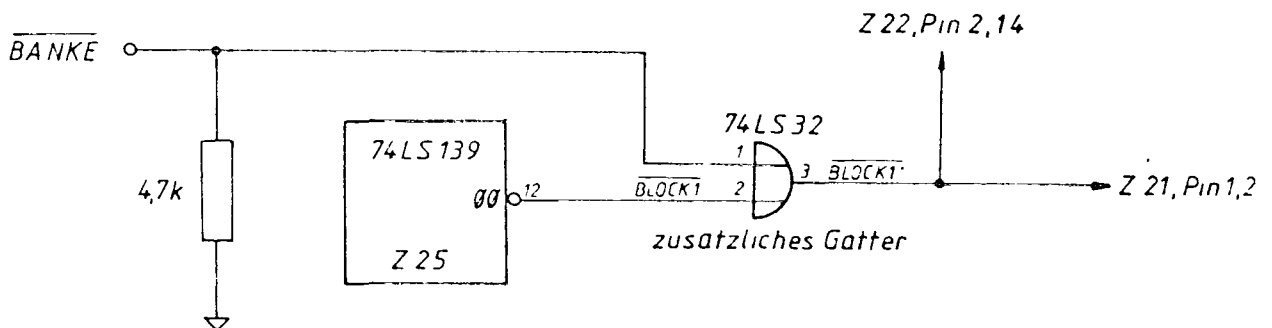


Fig. 5.1

Der 4,7 k $\Omega$  Widerstand hält den einen Eingang des Gatters auf Masse (logisch "0"), so daß die Grundeinheit ohne Expander oder mit einem nichtmodifizierten Expander arbeitet, als sei kein Eingriff erfolgt.

5.2.1.2 NMI-Taster

Ein Eingriff ist nicht nötig, da das  $\overline{\text{NMI}}$ -Signal bereits über Leitung 47 zum Expander geführt ist.

### 5.2.2 Eingriffe in Expander

In diesem Abschnitt werden die Änderungen beschrieben, die im Expander vorgenommen werden müssen.

#### 5.2.2.1 FDC-Adressen verlegen

Im CP/M-Mode darf das Signal  $\overline{37EX}$  des Expanders nicht bei 37EXH, sondern bei FFFXH aktiviert sein:

Die Leitung  $\overline{37EX}$  wird aufgetrennt und zur Zusatzplatine geführt, wo die Verknüpfung erfolgt.

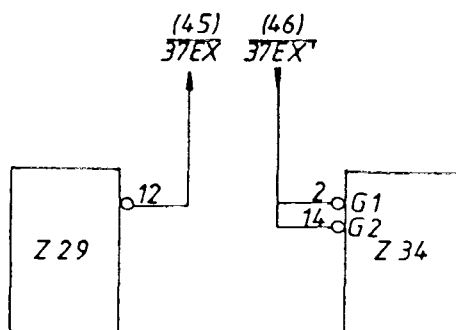


Fig. 5.2

#### 5.2.2.2 Expansion-RAM2 nicht enablen

Im CP/M-Mode darf das Expansion-RAM2 bei FC00H...FFFFH nicht aktiviert werden, da dort das Eprom und die com-Adressen liegen:

Die Leitung  $\overline{48k}$  des Expanders wird an Pin 2, Z32 aufgetrennt und auf der Zusatzplatine logisch verknüpft.

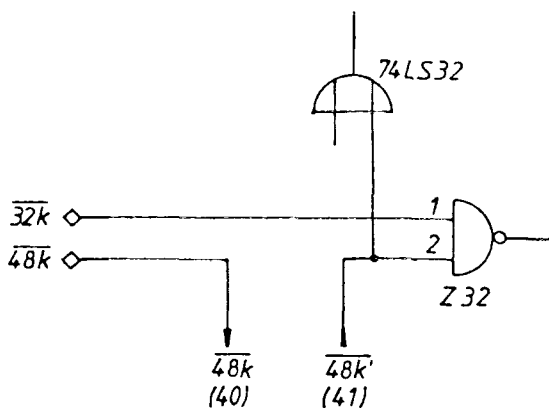


Fig. 5.3

### 5.2.2.3 Datenbus für zusätzlichen Speicher enablen

Ist das System in CP/M-Mode und die Bank nicht eingeschaltet, muß bei den Adressen  $0000H...3FFFH$  der Datenbus des Expanders zur Grundeinheit durchgeschaltet sein. Zu diesem Zweck kann ebenfalls das Signal  $\overline{48k}$  benutzt werden, das auf der Zusatzplatine zum Signal  $\overline{48k''}$  modifiziert wird.

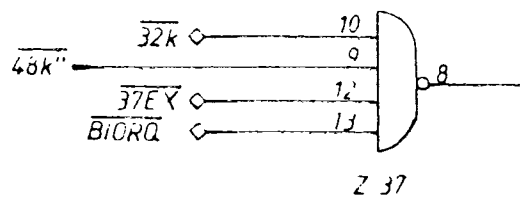


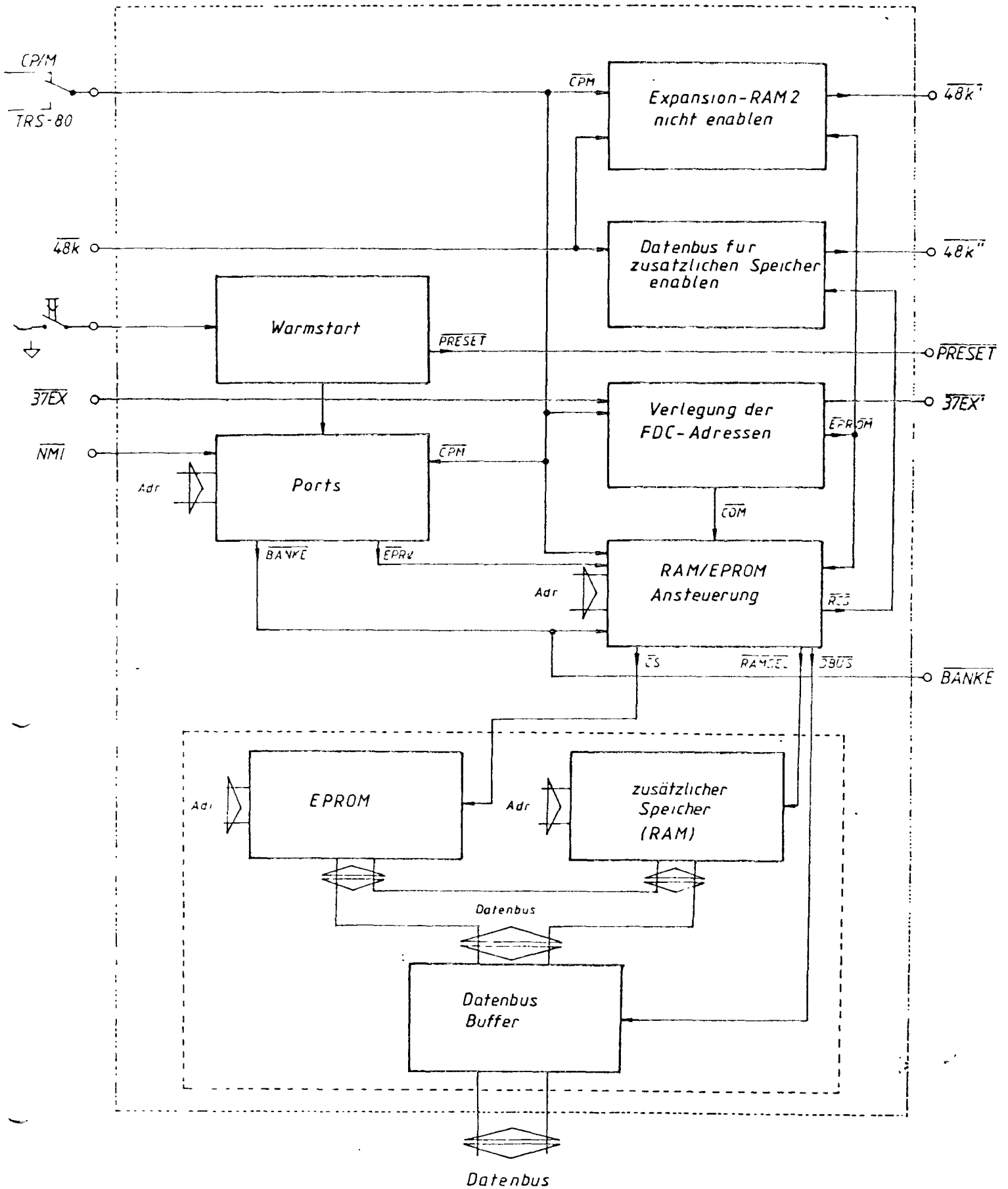
Fig. 5.4

### 5.2.3 Zusätzliche Platine

Die Platine enthält die logischen Verknüpfungen für die Änderungen in Grundeinheit und Expander, sowie den zusätzlichen Speicher, das Eprom, die Bankumschaltung und die dazugehörige Logik.

Im einzelnen handelt es sich um folgende Blöcke:

5.2.3.1 Blockschaltbild



### 5.2.3.2 Das Signal $\overline{\text{CPM}}$

Über den CP/M-Umschalter wird auf die Leitung  $\overline{\text{CPM}}$  0V gelegt. Das entspricht einer logischen "0":

CP/M ist eingeschaltet, falls  $\overline{\text{CPM}}$  = logisch "0"

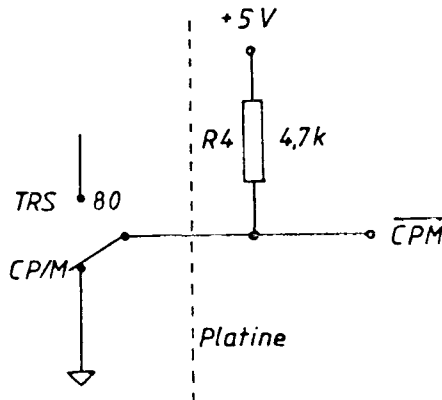
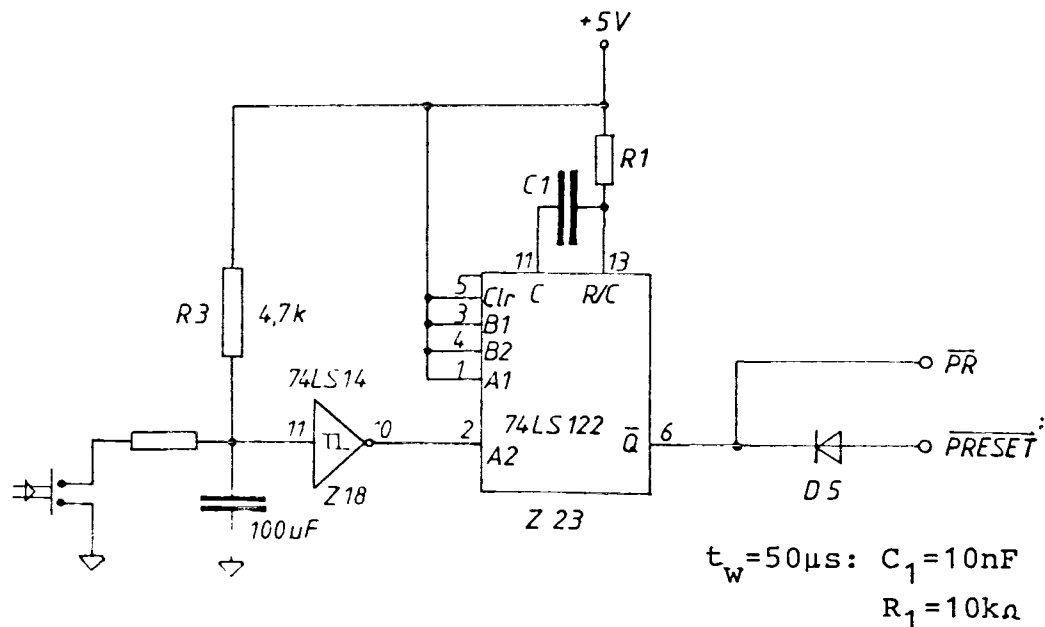


Fig. 5.6

### 5.2.3.3 Kalt- und Warmstart

Der Warmstart-Taster soll den Program-Counter in der CPU auf 0000H zurücksetzen und im CP/M-Mode das zusätzliche RAM auf Adresse 0000H schalten. Um die dynamischen RAMs aber weiterhin zu refreshen, darf  $\overline{\text{PRESET}}$  nur sehr kurz anliegen, weshalb ein Monoflop benötigt wird.





Im TRS-80-Mode löst der Warmstart-Taster einen Kaltstart aus.

Als Kaltstart-Taster wird im CP/M-Mode der in der Grundeinheit vorhandene NMI-Taster benutzt: Das Signal  $\overline{\text{NMI}}$  legt das Eprom auf  $0000\text{H}$  und schaltet die zweite Bank aus. Dies ist in der Portsteuerung (Fig. 5.10) durch Dioden realisiert worden.

#### 5.2.3.4 Verlegung der FDC-Adressen

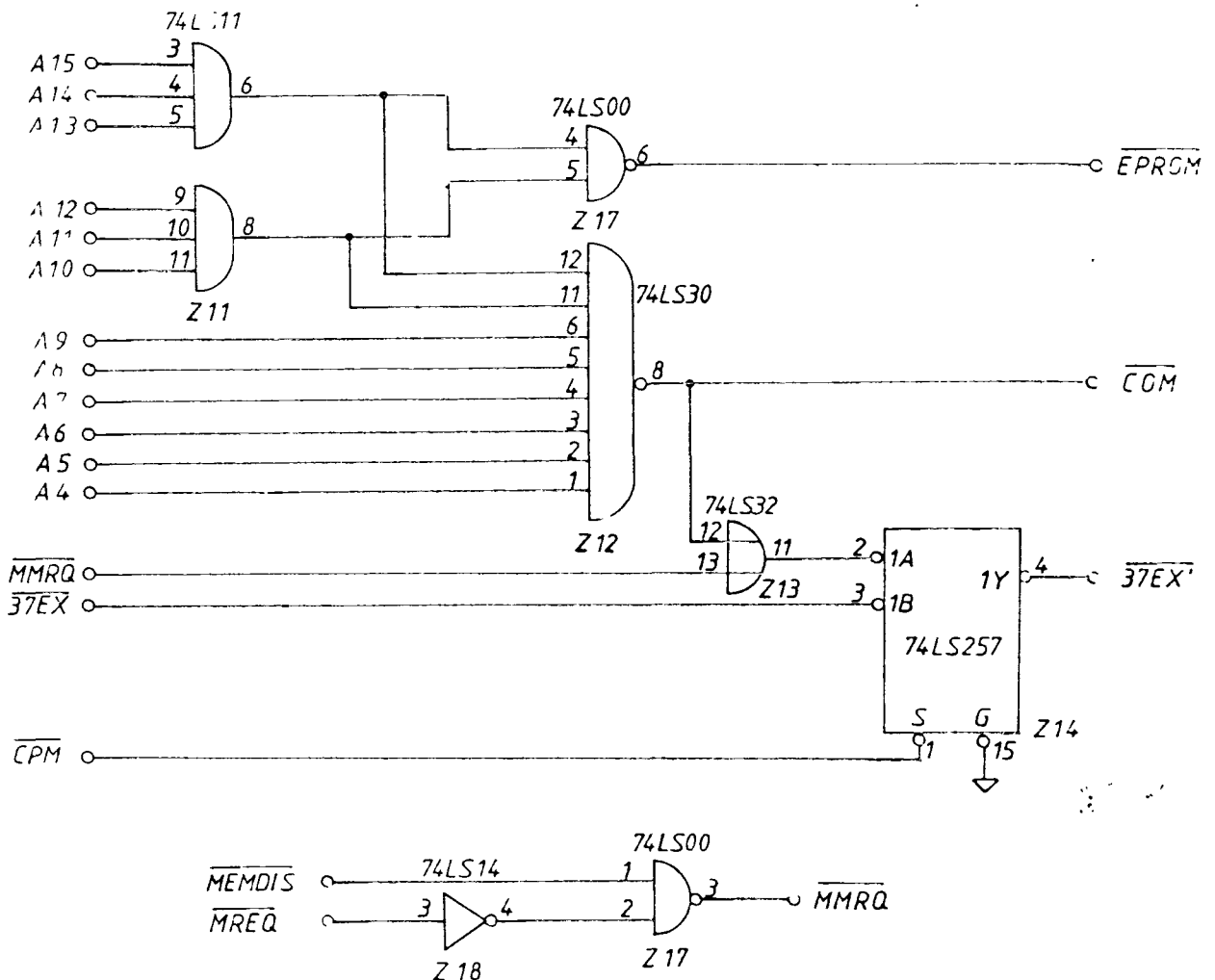
Im CP/M-Mode werden die FDC-Adressen an das Ende des Speichers verlegt ( $37\text{EXH} \rightarrow \text{FFFFH}$ ).

Die Signale  $\overline{\text{EPROM}}$  bzw.  $\overline{\text{COM}}$  sind dann logisch "0", wenn die Adressen  $\text{FCXXH}$  bzw.  $\text{FFFFH}$  anliegen:

$$\overline{\text{EPROM}} = \overline{\text{A15} \cdot \text{A14} \cdot \text{A13} \cdot \text{A12} \cdot \text{A11} \cdot \text{A10}}$$

$$\overline{\text{COM}} = \overline{\overline{\text{EPROM}} \cdot \text{A9} \cdot \text{A8} \cdot \text{A7} \cdot \text{A6} \cdot \text{A5} \cdot \text{A4}}$$

$$\overline{37\text{EX}} = \overline{\text{CPM} \cdot \text{MMRQ} \cdot \text{COM}} \quad \text{mit } \overline{\text{MMRQ}} = \overline{\text{MEMDIS} \cdot \text{MREQ}}$$



Der Multiplexer Z14 (74LS257) schaltet im TRS-80-Mode die B-Eingänge durch, so daß sie nicht verändert werden.

### 5.2.3.5 Expansion-RAM2 nicht enablen

Im CP/M-Mode soll gelten: Das Expansion-RAM2 darf bei den Eprom-Adressen nicht angesprochen werden:

$$\overline{48k'} = \overline{48k} \cdot \text{EPROM}$$

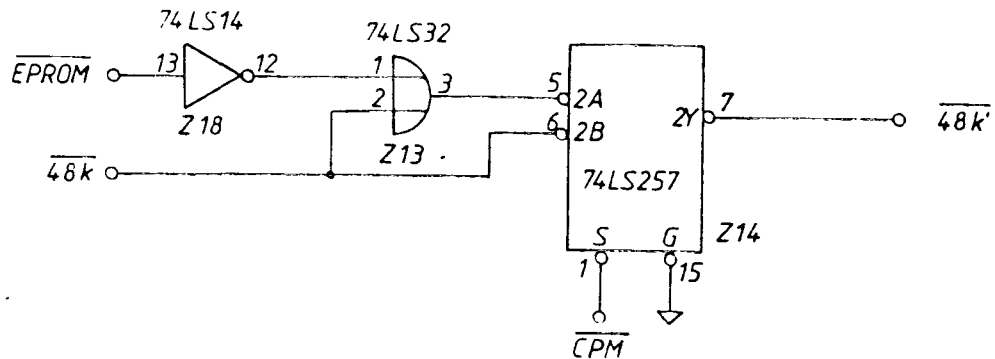


Fig. 5.9

### 5.2.3.6 Ports

Für das Umschalten in die zweite Bank und zum Verlegen des Eproms auf Adresse 0000H werden Ports benötigt. Um jedoch die Software möglichst einfach zu halten, wurden sie ohne Decodierung des Datenbus realisiert. Dadurch sind vier Ports erforderlich.

- $\overline{\text{EPR0}}$ : schaltet Eprom auf - Adr. 0000H (von Clr über D3 bzw. D4 oder Software)
- $\overline{\text{BANKE}}$ : schaltet zweite Bank - ein (von Software)
- Adr. FC00H (von Software oder über Pr (Warmstart))
- aus (von Software oder Pr über D1 (Warmstart))

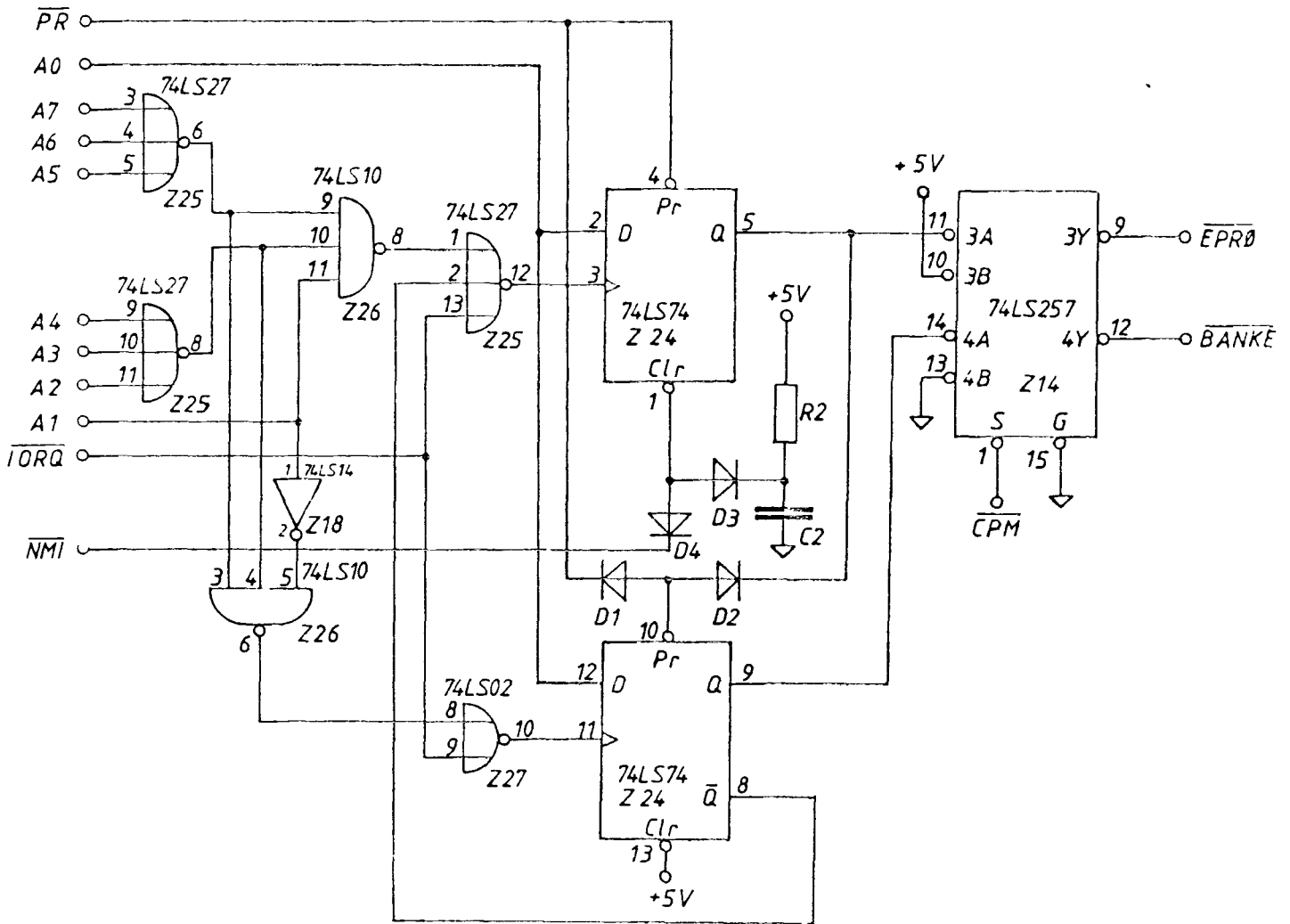


Fig. 5.10

Damit ergibt sich folgende hardwaremäßige Zuordnung:

- |          |                           |                         |
|----------|---------------------------|-------------------------|
| Port 00: | $\overline{\text{BANKE}}$ | Bank einschalten        |
| Port 01: | BANKE                     | Bank ausschalten        |
| Port 02: | $\overline{\text{EPRØ}}$  | Eprom auf Adresse 0000H |
| Port 03: | EPRØ                      | Eprom auf Adresse FC00H |

Diese Umschaltung erfolgt nur im CP/M-Mode. Im TRS-80-Mode ist über Z14 die Bank immer ein- und das Eprom immer ausgeschaltet.

Als Verriegelung wirkt die Diode D2. Sie verhindert, daß die Bank eingeschaltet ist und zugleich das Eprom auf 0000H liegt. Die Kombination C2, R2, D3 setzt das  $\overline{\text{EPRØ}}$ -Flipflop nach dem Einschalten zurück (das Eprom

wird auf Adresse 0000H gelegt und über D2 wird die Bank ausgeschaltet). Ein Rücksetzen dieses Flipflops kann auch durch einen NMI über D4 ausgelöst werden.

### 5.2.3.7 RAM-/Eprom-Ansteuerung

Zur Ansteuerung des RAMs und des Eproms werden drei Signale benötigt:

- Das neue RAM wird adressiert, wenn im CP/M-Mode der erste Block (die ersten 16kbyte) des Speichers ( $\overline{MMRQ}$ ) adressiert wird, das Eprom nicht auf 0000H liegt und die Bank nicht eingeschaltet ist.

$$\overline{RAMSEL} = \overline{A14} \cdot \overline{A15} \cdot \overline{CPM} \cdot \overline{EPR0} \cdot \overline{BANKE} \cdot \overline{MMRQ}$$

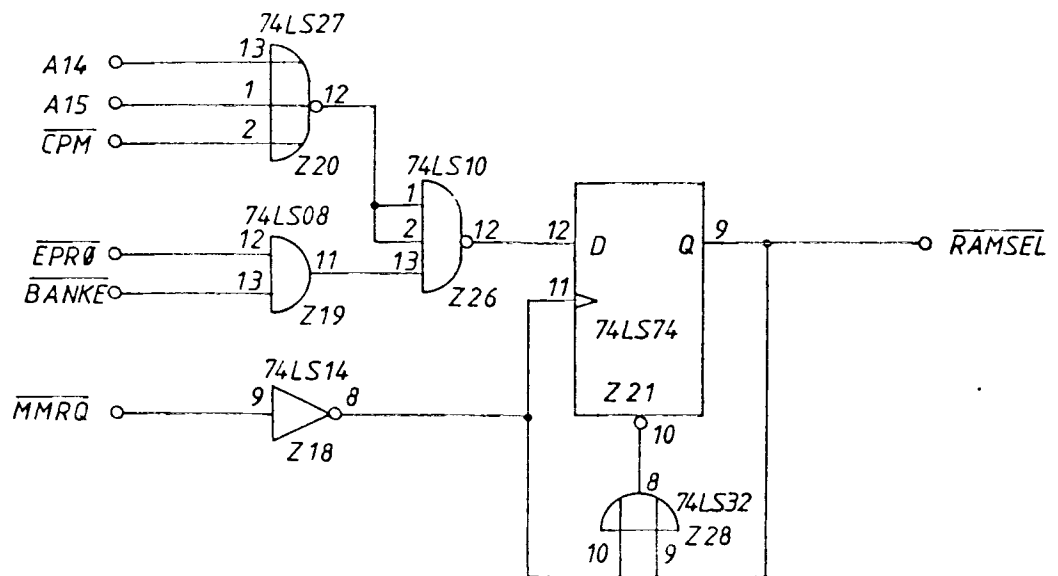


Fig. 5.11

Am Ende des M1-Zyklus der CPU sind die Adressen schon wieder auf "0" zurückgesetzt, obwohl noch  $\overline{MMRQ}$  anliegt, da dies Signal durch Gatterlaufzeiten verzögert ist. So würde kurzzeitig das RAM des ersten Blocks angesprochen. Verhindert wird dies durch das Flipflop Z21, das

nur bei der neg. Flanke des Signals  $\overline{\text{MMRQ}}$  die deco-  
dierten Adressen durchschaltet. Weitere Änderungen,  
(während  $\overline{\text{MMRQ}} = "0"$  ist) werden herausgefiltert.

- Das Eprom soll angeschaltet sein, wenn im CP/M-Mode  
entweder die Speicheradressen  $0000\text{H} \dots 0400\text{H}$  adres-  
siert sind und  $\overline{\text{EPR0}}$  aktiv ist (Eprom auf Adresse  
 $0000\text{H}$ ), oder die Speicheradressen  $\text{FC00H} \dots \text{FFEH}$   
adressiert sind und  $\overline{\text{EPR0}}$  nicht aktiv ("1") ist (Eprom  
ab Adresse  $\text{FC00H}$ ).

Im TRS-80-Mode ist  $\overline{\text{EPR0}}$  immer logisch "1", so daß  $\overline{\text{CPM}}$   
im ersten Fall nicht berücksichtigt zu werden braucht.

$$\overline{\text{CS}} = \overline{\text{MMRQ}} \cdot (\overline{\text{A11}} \cdot \overline{\text{A12}} \cdot \overline{\text{A13}} \cdot \overline{\text{A14}} \cdot \overline{\text{A15}} \cdot \overline{\text{EPR0}}) + (\overline{\text{CPM}} \cdot \overline{\text{EPROM}} \cdot \overline{\text{COM}} \cdot \overline{\text{EPR0}})$$

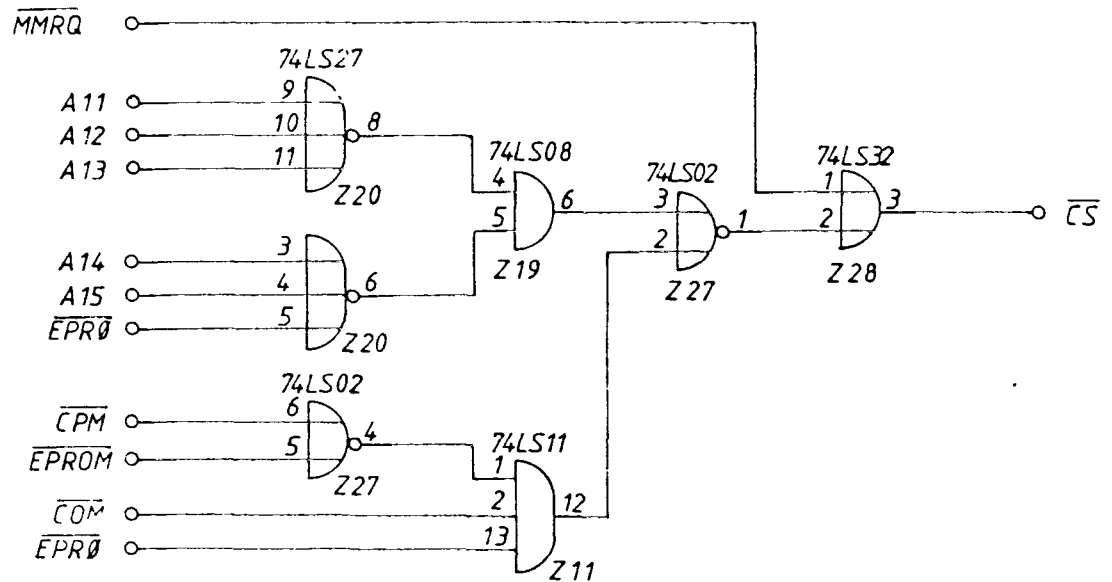


Fig. 5.12

- Der Datenbus wird durchgeschaltet, wenn Eprom oder  
RAM adressiert sind und  $\overline{\text{RD}}$  aktiv ("0") ist:

$$\overline{\text{DBUS}} = (\overline{\text{RAMSEL}} + \overline{\text{CS}}) \cdot \overline{\text{RD}} = \overline{\text{RCS}} \cdot \overline{\text{RD}} \quad \text{mit } \overline{\text{RCS}} = \overline{\text{RAMSEL}} + \overline{\text{CS}}$$

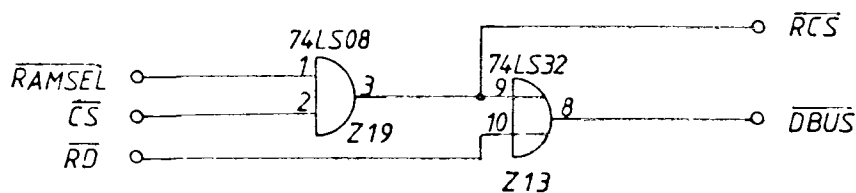


Fig 5.13

### 5.2.3.8 Datenbus für zusätzlichen Speicher enablen

Im CP/M-Mode soll gelten:  $\overline{48k''} = \overline{48k} + \overline{RCS}$

Da im TRS-80-Mode  $\overline{RCS}$  immer logisch "0" ist, wird dann das Signal  $\overline{48k}$  unverändert durchgeschaltet und eine Unterscheidung ist nicht nötig.

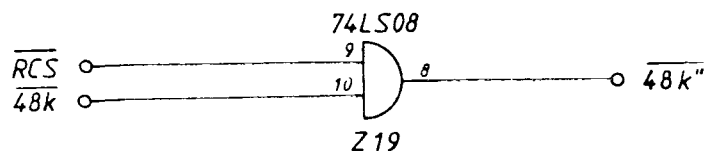


Fig. 5.14

### 5.2.3.9 Eprom und zusätzlicher Speicher

Die Realisierung von RAM und Eprom erfolgt in gewohnter Weise.

Die RAMs erhalten ihre Adressen von den Multiplexern Z15 und Z16. Zunächst liegen die Adressen A0-A6 an und werden mit dem Signal  $\overline{MREQ}$  ( $\hat{=} \overline{RAS}$ ) in die RAMs eingelesen. Mit dem MUX-Signal werden die Adressen A7-A13 durchgeschaltet und nach Anliegen des  $\overline{CAS}$ -Signals ebenfalls eingelesen. Anschließend werden je nach Zustand von  $\overline{MWRITE}$  die Daten ein- bzw. ausgegeben. Fig. 5.15 stellt die zeitlichen Zusammenhänge der Signale  $\overline{RAS}$ ,  $\overline{CAS}$ , und MUX dar.

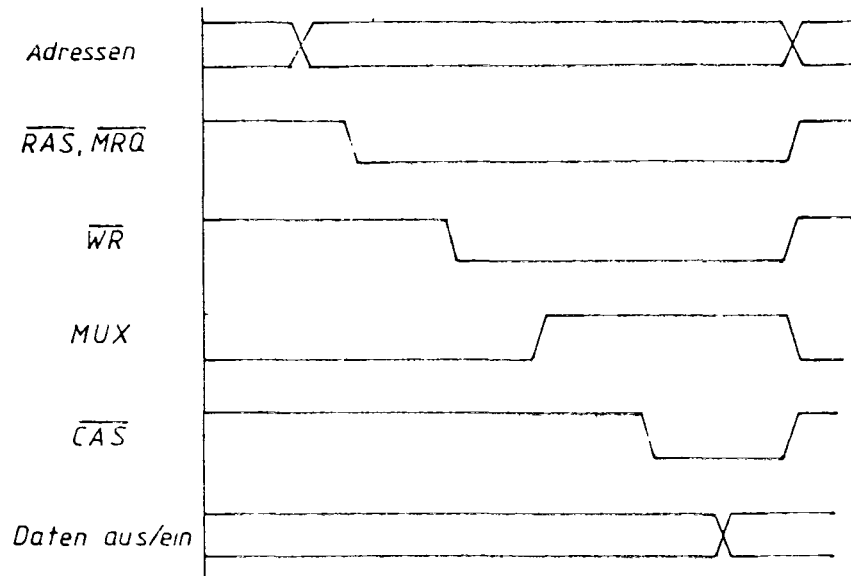


Fig. 5.15

Am Eprom liegen die Adressen unmittelbar an (A10 liegt auf Masse, da vom 2kbyte-Eprom nur 1kbyte benutzt wird). Die ausgelesenen Daten werden über den Datenbustreiber gepuffert und dann auf den Expander-Datenbus gegeben.

Bei der Inbetriebnahme lagen die Pegel der RAM- bzw. Epromausgänge im hochohmigen Zustand genau in Höhe der Schaltschwelle des Datenbustreibers und führten zu Instabilitäten und Spannungsspitzen auf allen Leitungen. Daraufhin wurden alle Ausgänge über 4,7 k $\Omega$ -Widerstände an +5V gelegt.

Der Schaltplan der RAMs und des Eproms befindet sich im Anhang.

#### 5.2.3.10 Stromversorgung

Die Stromversorgung ist mit integrierten Stabilisatoren aufgebaut. Die drei Spannungen +5V, +12V, -5V werden aus den vom Netzteil gelieferten ungestabilisierten Span-

nungen +8V, +16V, -16V abgeleitet und mit den Elkos C3-C5 geglättet. Die Dioden D6-D8 verhindern, daß sich falsch gepolte Spannungen auf den Versorgungsleitungen aufbauen können.

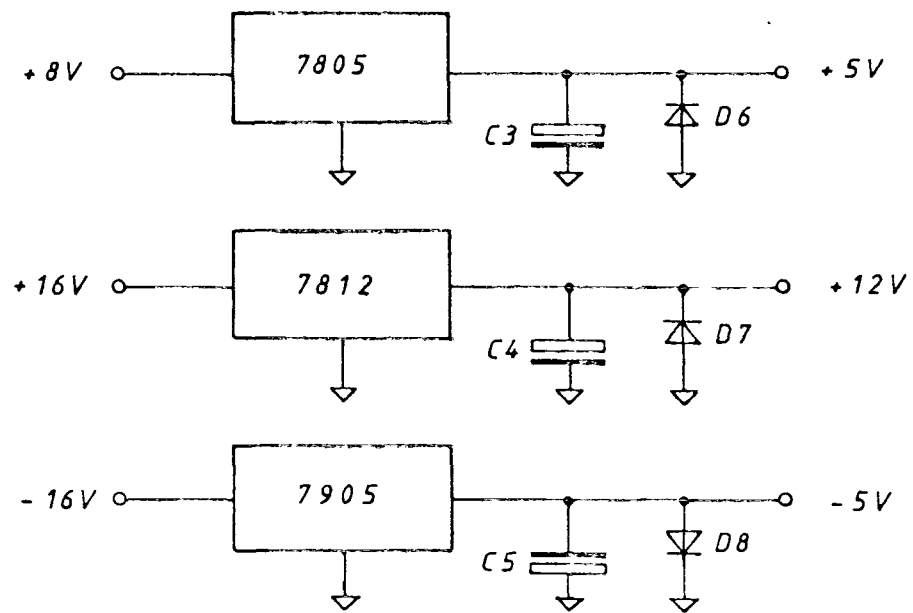


Fig 5.16



### 5.3 Praktische Ausführung

Realisiert wurde die gesamte Schaltung in Fädertechnik auf der zusätzlichen Platine (siehe Anhang "Layout"). Die Verbindung zur Platine im Expander wurde über den im Expander vorhandenen S100-Bus-Connector vorgenommen, wobei einzelne Leitungen geändert werden mußten (siehe Pinbelegung Steckerleiste im Anhang).

In der Grundeinheit wurde das benötigte IC durch Einlöten über ein bereits vorhandenes (Z21) eingefügt.

## 6. Software

### 6.1. Speicherbelegung der Zeropage

Die Speicherplätze von 0000H bis 00FFH (Zeropage) enthalten kein Programm, sondern dienen als Einsprünge und Zwischenspeicher des CP/M. Da diese Adressen in allen Programmteilen auftauchen, werden sie hier für die 63k-Version zusammengefaßt dargestellt:

0000H-0002H:	C3 03 F6	Sprung zur Warmstart-Routine im BIOS
0003H:	xx	
0004H:	xx	aktuelle Drive-Nr.
0005H-0007H:	C3 00 E8	Sprung zum BDOS (als Einsprung für sämtliche Kommunikation zwischen transienten Programmen und dem CP/M)
0008H-0037H:	-	nicht benutzt
0038H-003AH:	-	nur von DDT.COM und SID.COM benutzt
003BH-003FH:	-	nicht benutzt
0040H-0041H:	xx xx	Cursorstand
0042H:	xx	von Cursor verdecktes Zeichen
0043H:	xx	für Escape-String
0044H:	xx	für Strichgraphik
0045H:	xx	Kleinschrift-Großschrift
0046H:	xx	von Tastatur eingegebenes Zeichen (nach PCONST)
0047H-004DH:	xx	Tastaturstatus
004EH:	xx	für "Echo Deleted Characters"
004FH:	xx	von Tastatur eingelesenes Zeichen (nach PCONIN)
0050H-005BH:	-	nicht benutzt
005CH-007CH:	xx	File-Control-Block von CCP für transiente Programme
007DH-007FH:	xx	
0080H-00FFH:	xx	Disk-Buffer, bzw. Kommandozeile eines transienten Programms

## 6.2 Beschreibung der Systemprogramme

### 6.2.1 Systemstart durch Eprom und Cold-Start-Loader

Das Eprom hat zunächst die Aufgabe, das System zu starten und den Cold-Start-Loader einzulesen. Danach wird es als Erweiterung zum BIOS benutzt und enthält Ein-/Ausgaberoutinen, die vom BIOS angesprungen werden.

Das Flußdiagramm zeigt den Ablauf vom Kaltstart bis zum Start des Cold-Start-Loaders.

Kaltstart:

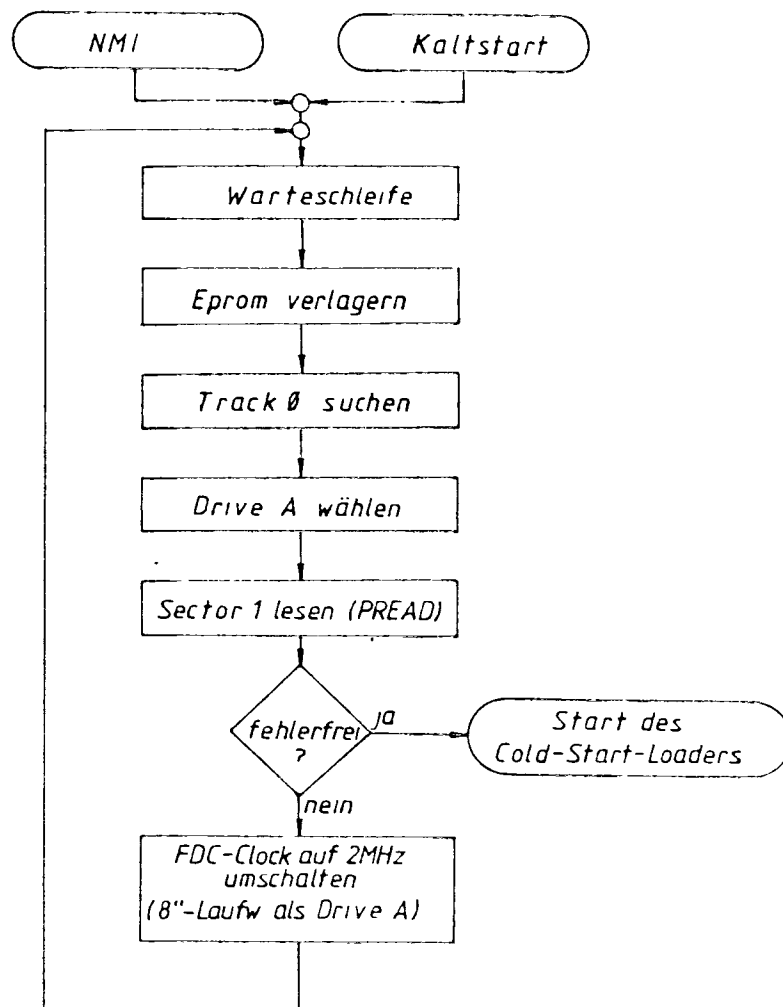


Fig. 6.1

Anschließend liest der Cold-Start-Loader das gesamte CP/M-Betriebssystem von Diskette in den Speicher und startet es. Auch die Funktion des Cold-Start-Loaders wird nachfolgend in einem Flußdiagramm dargestellt:

Cold-Start-Loader:

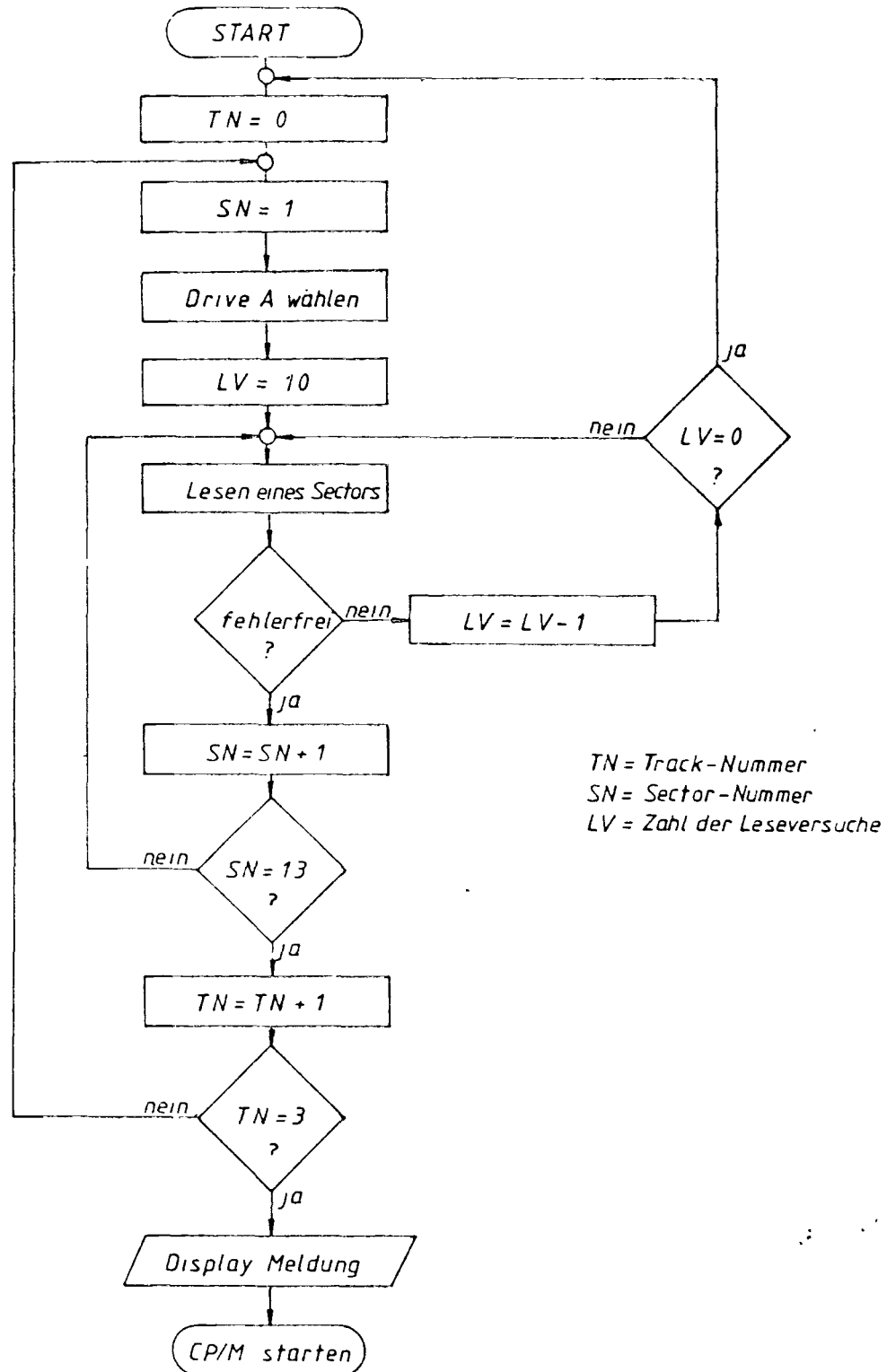


Fig. 6.2

## 6.2.2 Unterprogramme des Eproms (CBIOS)

In diesem Kapitel werden die im Eprom enthaltenen Unterprogramme beschrieben. Aufgeführt werden jeweils Ein- und Ausgabeparameter, aufgerufene Unterprogramme und eine kurze Beschreibung der Funktionsweise. Ein kommentiertes Listing befindet sich im Anhang.

### 6.2.2.1 CMDFDC

Dieses Unterprogramm gibt ein Kommando an den Floppy-Disc-Controller und durchläuft anschließend eine Warteschleife.

Eingabeparameter:	BC-Reg.:	ØFFFCH (Commandreg.)
	A-Reg.:	auszuführender Befehl
Ausgabeparameter:	BC-Reg.:	ØFFFCH (Commandreg.)
	A-Reg.:	ØH
gerufene UP:		keine

### 6.2.2.2 DRVSEL

Dieses Unterprogramm selektiert gewählten Drive.

Eingabeparameter:	E-Reg.:	Drive Nummer (Ø1H, Ø2H, Ø4H, Ø8H für Drive A, B, C, D)
Ausgabeparameter:	BC-Reg.:	ØFFFCH (Statusreg.)
gerufene UP:		LOOP3 (Warteschleife)

### 6.2.2.3 WRPTXT

Dieses Unterprogramm gibt den Text "PROTECTED TYPE C" aus und erwartet eine Eingabe:

- bei C erfolgt RET
- bei CTRL C erfolgt ein Kaltstart

Eingabeparameter:	keine )	alle Register bleiben unverändert
Ausgabeparameter:	keine )	
gerufene UP:		DSPLMS PCONIN

#### 6.2.2.4 PREAD

Dieses Unterprogramm liest einen Sector von Diskette und schreibt die gelesenen Daten in den Speicherbereich, auf den HL zeigt.

Eingabeparameter: BC-Reg.: 0FFFCH (Statusreg.)  
HL-Reg.: Zeiger auf Speicherplatz,  
von dem an die Daten abge-  
speichert werden sollen

Ausgabeparameter: A-Reg.: Fehlerbits (mit Bit 5,6 = 0)  
BC-Reg.: 0FFFCH (Statusreg.)  
DE-Reg.: 0FFFFH (Datenreg.)  
HL-Reg.: Zeiger auf nächsten zu be-  
schreibenden Speicherplatz

gerufene UP: CMDFDC

#### 6.2.2.5 PWRITE

Dieses Unterprogramm liest Daten aus dem Speicherbereich, auf den das HL-Reg. zeigt, und schreibt sie auf einen Sector der Diskette. Ist beim Aufruf das Carry-Flag gesetzt, so muß das adressierte Laufwerk ein 8"-Laufwerk sein, ist es zurückgesetzt, so wird ein Mini-Laufwerk erwartet.

Eingabeparameter: BC-Reg.: 0FFFCH (Commandreg.)  
DE-Reg.: 0FFFFH (Datenreg.)  
HL-Reg.: Zeiger auf Daten im Speicher  
Carry- "0" bei Mini-Laufwerk  
Flag: "1" bei 8"-Laufwerk

Ausgabeparameter: BC-Reg.: 0FFFCH (Commandreg.)  
DE-Reg.: 0FFFFH (Datenreg.)  
HL-Reg.: Zeiger auf nächsten noch  
nicht abgespeicherten  
Speicherplatz

gerufene UP: CMDFDC  
WRPROT

#### 6.2.2.6 WRPROT

Dieses Unterprogramm überprüft das Statusregister des FDC. Ist das "Protected-Flag" nicht gesetzt, erfolgt unmittelbar ein RET, sonst wird zunächst das UP WPRTXT aufgerufen.

Eingabeparameter: BC-Reg.: 0FFFCH (Statusreg.)  
Ausgabeparameter: A-Reg.: Inhalt des Statusreg.  
des FDC  
BC-Reg.: 0FFFCH (Statusreg.)  
gerufene UP: WPRTXT

#### 6.2.2.7 PCNOU

Dieses Unterprogramm gibt das im C-Register enthaltene Zeichen auf dem Bildschirm aus. Es bewegt den Cursor und führt alle Steuerbefehle aus, es sei denn, das vorhergehende Zeichen war ein 01H.

Video-Steuerbefehle sind:

<u>ASCII</u>	<u>Wirkung:</u>
01H	das folgende Zeichen wird in ein Zeichen zwischen 00H und 1FH umgewandelt und auf dem Bildschirm ausgegeben (siehe Teil II)
08H	Backspace und Delete
0AH	Linefeed (LF)
0DH	Carriage Return (CR)
18H	Cursor nach links
19H	Cursor nach rechts
1AH	Cursor runter
1BH	siehe ESCAPE
1CH	Cursor an Bildschirmumfang und Bildschirm löschen

Eingabeparameter: C-Reg.: auszugebendes Zeichen  
Ausgabeparameter: keine  
gerufene UP: ESCAPE

#### 6.2.2.8 ESCAPE

Dieses Unterprogramm testet, ob z.Zt. ein Escape-String in Bearbeitung ist oder das aktuelle Zeichen ein Escape (1BH) ist.

Es gibt zwei Arten von Escape-Strings:

- Escape,\*  
(1BH,2AH) : setzt Cursor an den Bildschirm-  
fang und löscht dann den gesamten  
Bildschirm (=Rückkehr aus ESCAPE  
mit 1EH)
- Escape,=,X1,X2 : positioniert Cursor an eine belie-  
(1BH,3DH,X1,X2) bige Stelle im Bildschirm.  
Dabei ist X1 die Zeilenr. + 20H  
und X2 die Spaltenr. + 20H.

Eingabeparameter: A-Reg.: auszugebendes Zeichen  
Ausgabeparameter: A-Reg.: - 00H, wenn nur der Cursor  
positioniert wurde, bzw.  
mitten im String  
- 1EH, wenn der Bildschirm  
gelöscht werden soll  
DE-Reg.: Cursorposition  
gerufene UP: keine

#### 6.2.2.9 PCONST

Dieses Unterprogramm fragt die Tastatur ab, ob eine Taste gedrückt ist, und speichert das eingegebene Zeichen nach 0046H.

Eingabeparameter: keine  
Ausgabeparameter: A-Reg.: 00H, wenn keine Taste  
gedrückt;  
FFH, wenn Taste gedrückt  
(0046H): eingegebenes Zeichen  
gerufene UP: INKEY



#### 6.2.2.10 PCONIN

Dieses Unterprogramm wartet auf Eingabe eines Zeichens über die Tastatur, toggelt nach Eingabe von CTRL Ø und wandelt je nach "Zustand von CTRL Ø" Kleinschrift in Großschrift um.

Eingabeparameter: keine  
Ausgabeparameter: (ØØ4FH): eingegebenes Zeichen  
A-Reg.: eingegebenes Zeichen  
gerufene UP: PCONST

#### 6.2.2.11 INKEY

Dieses Unterprogramm fragt die Tastaturmatrix ab und generiert daraus den OMIKRON-ASCII-Code, der folgende Besonderheiten aufweist:

<u>Taste</u>	<u>Funktion</u>
Down-Arrow	CTRL
CTRL Ø	CAPS-LOC
Right-Arrow	Ø9H; CTRL I
Shift-(Break)	1BH; Escape
Left-Arrow	7FH; Delete
Shift-(Enter)	ØAH; LF
Break	Ø8H; Backspace
Shift-(Left-Arrow)	5BH; Left Bracket
Shift-(Right-Arrow)	5DH; Right Bracket
Shift-(Clear)	5EH; Up-Arrow
Shift-(Up-Arrow)	5CH; Backslash
Clear	13H; CTRL S
Up-Arrow	1AH; CTRL Z
CTRL 1	5FH
CTRL 2	1CH
CTRL 3	1DH
CTRL 4	1EH
CTRL 5	1FH
CTRL 6	7BH
CTRL 7	7CH
CTRL 8	7DH
CTRL 9	7EH

Eingabeparamter: keine  
Ausgabeparameter: A-Reg.: ASCII-Code  
des eingegebenen Zeichens  
gerufene UP: keine

#### 6.2.2.12 DSPLMS

Dieses Unterprogramm gibt einen Textstring auf dem Bildschirm aus. Die Ausgabe wird beendet, sobald im String ein  $\emptyset H$  auftritt. Es erfolgt dann die Rückkehr zum aufrufenden Programm.

Eingabeparameter: HL-Reg.: Zeiger auf Textanfang  
im Speicher  
Ausgabeparameter: keine  
gerufene UP: PCONOUT

#### 6.2.2.13 PREADY

Dieses Unterprogramm testet, ob im adressierten Laufwerk eine Diskette eingelegt ist und sich dreht. Zwei Fälle sind möglich:

- Diskette dreht sich: RET
- Diskette dreht sich nicht: Ausgabe: "NOT READY TYPE C"  
nach Eingabe von C: neuer Text  
nach Eingabe von CTRL C:  
CP/M-Warmstart

Eingabeparameter: keine  
Ausgabeparameter: keine  
gerufene UP: CMDFDC  
DRVSEL

### 6.2.3 BIOS

Das BIOS wird fast vollständig in der OMIKRON-Version übernommen. Es sind nur die Adressen zu ändern, die Routinen im Eprom (CBIOS) aufrufen, da dies modifiziert wurde und jetzt an anderer Stelle im Speicher liegt. Die im Eprom anzuspringenden Adressen sind am Anfang des BIOS durch Zuweisungen festgelegt (siehe Anhang: BIOS). Sie werden wie nachfolgend dargestellt abgeändert:

MEMSIZ	48	→	63
BIAS	0C000H	→	0000H
PREAD	0C00DH	→	0FC0BH
PWRITE	0C010H	→	0FC0EH
PCONIN	0C004H	→	0FC02H
PCONOUT	0C007H	→	0FC05H
PCONST	0C00AH	→	0FC08H
PREADY	0C01FH	→	0FC14H

### 6.3 Beschreibung der geänderten transienten Programme

Von OMIKRON werden leider nicht alle Programme voll CP/M-kompatibel geliefert. Einige ihrer transienten Programme greifen unmittelbar auf das OMIKRON-Eprom zu, das jetzt an einer anderen Stelle im Speicher liegt und modifiziert ist. So sind diese Programme nun nicht mehr lauffähig und müssen ebenfalls geändert werden.

#### 6.3.1 SETUP63.COM

Dieses Programm benutzt die Ein- und Ausgaberroutinen des Eproms. Es läßt sich jedoch leicht so abändern, daß auf die Ein- und Ausgaberroutinen des CP/M-Systems zurückgegriffen werden kann:

Anstatt unmittelbar DSPLMS oder PCONIN im Eprom aufzurufen, werden gleichnamige Unterprogramme angehängt, deren Flußdiagramme im folgenden dargestellt sind:

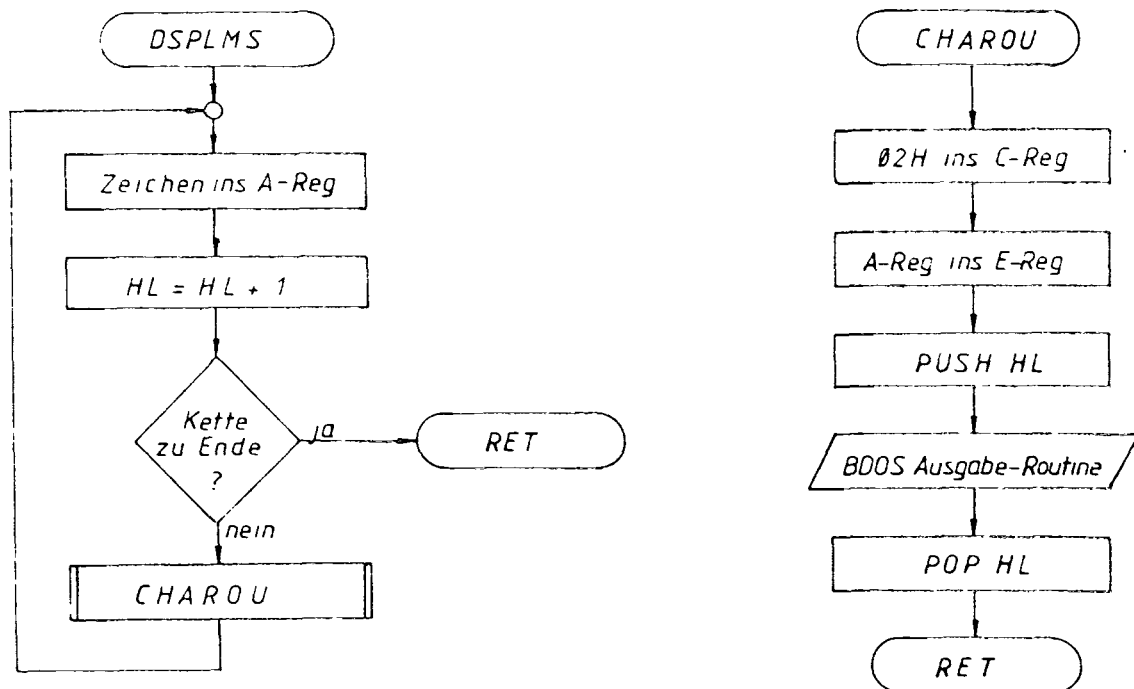


Fig. 6.3

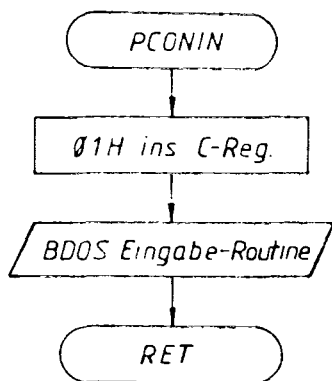


Fig. 6.4

Ein kommentiertes Listing des Programms befindet sich im Anhang.

### 6.3.2 MFORM63.COM; LFORM63.COM

Die Programme MFORMAT und LFORMAT greifen zum einen auf das Diskettentest-Unterprogramm PREADY im Eprom zu, sprechen aber auch den FDC unmittelbar an. Hier müssen sowohl die Adressen zum Eprom, als auch die des FDC geändert werden. Beide Programme sind für die 63k-Version umgeschrieben, durchlaufen aber zunächst eine Routine, die überprüft, ob nicht die 48k-Version vorliegt. Wenn dies der Fall ist, werden alle "falschen" Adressen überschrieben (so daß wieder die Original-Version vorliegt), bevor das eigentliche Programm gestartet wird.

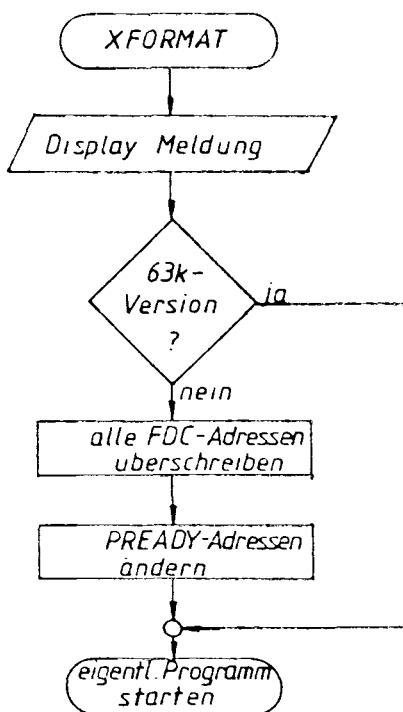


Fig. 6.5

Ein kommentiertes Listing der Programme befindet sich  
im Anhang.

T e i l II

7. Strichgraphik

Der zweite Teil der Arbeit befaßt sich mit der Implementierung einer Strichgraphik für die Bildschirmausgabe.

Zunächst wird das bestehende System analysiert. Es folgt dann eine Darstellung der geplanten Strichgraphik und abschließend die hardwaremäßige Lösung.

7.1 Zeichendarstellung auf dem Bildschirm





Auf dem Bildschirm werden 16 Zeilen à 64 Zeichen dargestellt. Jedes dieser Zeichen ist entweder ein Schrift- oder ein Graphikzeichen. Graphikzeichen füllen die gesamte Matrix von 6 x 12 Punkten aus, während Schriftzeichen aus einer 5 x 7 - Matrix aufgebaut sind. Die übrigen Punkte sind dunkel und dienen zur Abgrenzung zwischen den einzelnen Zeichen bzw. zwischen den Zeilen.

	0	1	2	3	4	5
0				•		
1			•		•	
2		•				•
3		•				•
4		•	•	•	•	•
5		•				•
6		•				•
7						
8						
9						
10						
11						

Fig. 7.1

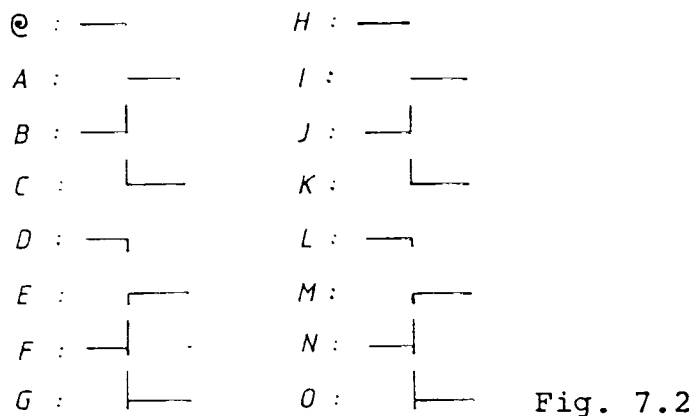
### 7.2 Strichgraphik-Darstellung auf dem Bildschirm

Die Strichgraphik soll es ermöglichen, einzelne Wörter unterstreichen bzw. einrahmen zu können. Sie wird in folgende Segmente aufgeteilt:

- Beginn einer Unterstreichung  bzw. Ende  Bit 0
- Senkrechter Strich nach oben  Bit 1
- Senkrechter Strich nach unten  Bit 2

Nach dem "Beginn einer Unterstreichung" wird ein Strich bis zum Ende der Zeile oder zum Zeichen "Ende" erzeugt.

Aus diesen Segmenten läßt sich durch Bitkombinationen die gewünschte Graphik erzeugen (siehe 7.6 "Benutzungsbeispiel"). Es ergibt sich die in Fig. 7.2 dargestellte Zuordnung:



Im einzelnen müssen folgende Bildpunkte hellgetastet werden:

	0	1	2	3	4	5
0				o		
1				o		
2				o		
3				o		
4				o		
5				o		
6				o		
7				o		
8				o		
9	x	x	x	o	•	•
10				-		
11				-		

Fig. 7.3



### 7.3 Blockschaltbild

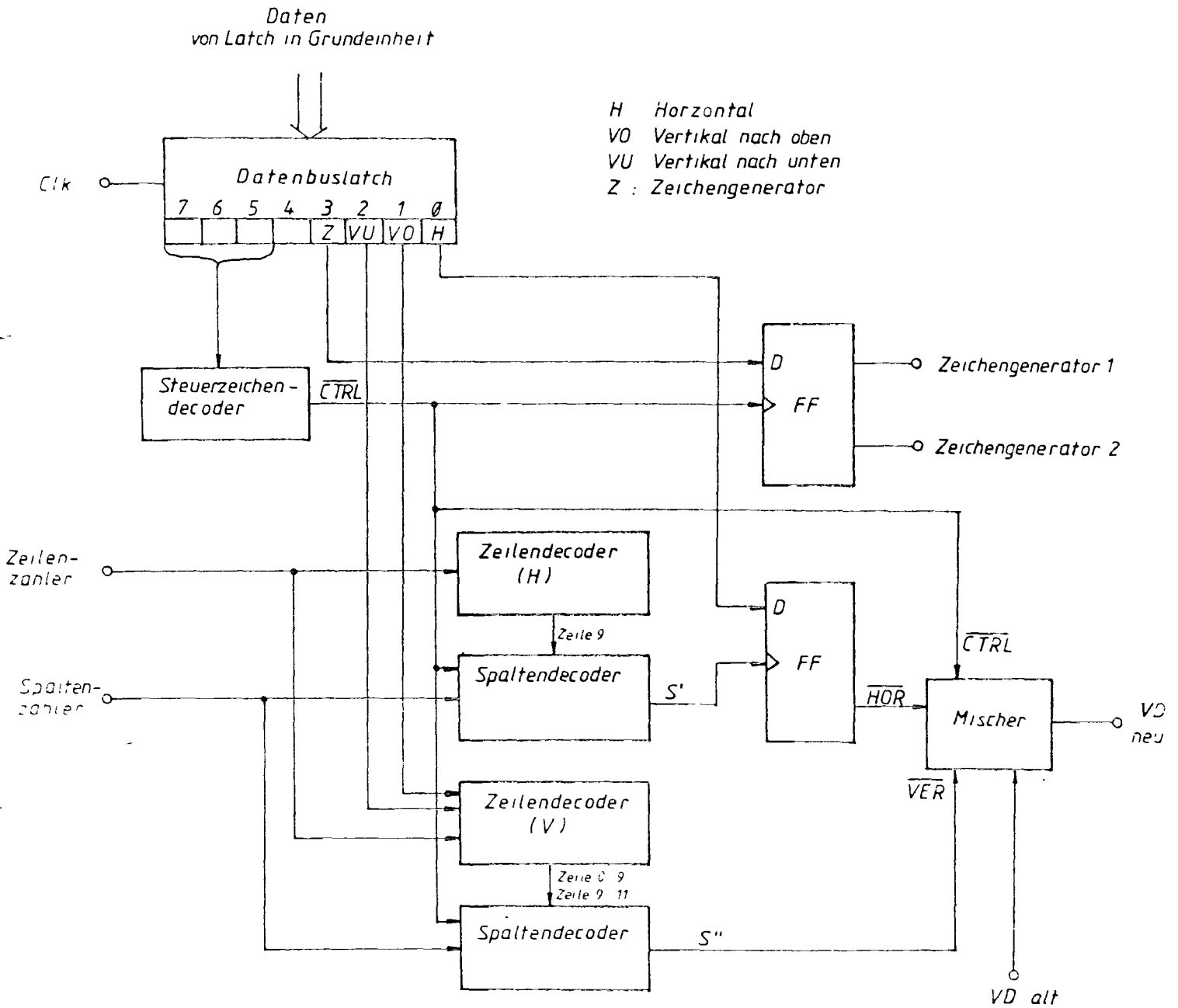


Fig. 7.4

Die Daten des BildwiederholSpeichers werden aus dem in der Grundeinheit vorhandenen Latch übernommen und mit dem Datenbuslatch verzögert, um die Strichgraphik synchron

zur Characterdarstellung auszugeben. Bit 5, 6 und 7 werden zur Decodierung der Strichgraphik-Steuerzeichen herangezogen. Bit 3 dient zum Umschalten auf einen anderen Zeichengenerator. Die Bits 0 - 2 steuern die einzelnen Segmente der Strichgraphik.

Zwei Zeilendecoder decodieren die jeweilige Zeile. Wenn ein Strichgraphik-Steuerzeichen und die richtige Zeile vorliegen, wird auch die Spalte decodiert. Horizontale Striche schalten ein Flipflop, während vertikale unmittelbar auf den Mischer gelangen. Der Mischer mischt vertikale und horizontale Bildpunkte und unterdrückt die Buchstaben des Charactergenerators.

## 7.4 Hardware

### 7.4.1 Datenbuslatch

Der Datenbuslatch verzögert die Daten aus dem Bildwiederholungspeicher um eine Stelle auf dem Bildschirm, damit das Zeichen der Strichgraphik nicht einen Platz zu früh auftritt.

Das nachstehende Diagramm veranschaulicht die zeitlichen Zusammenhänge:

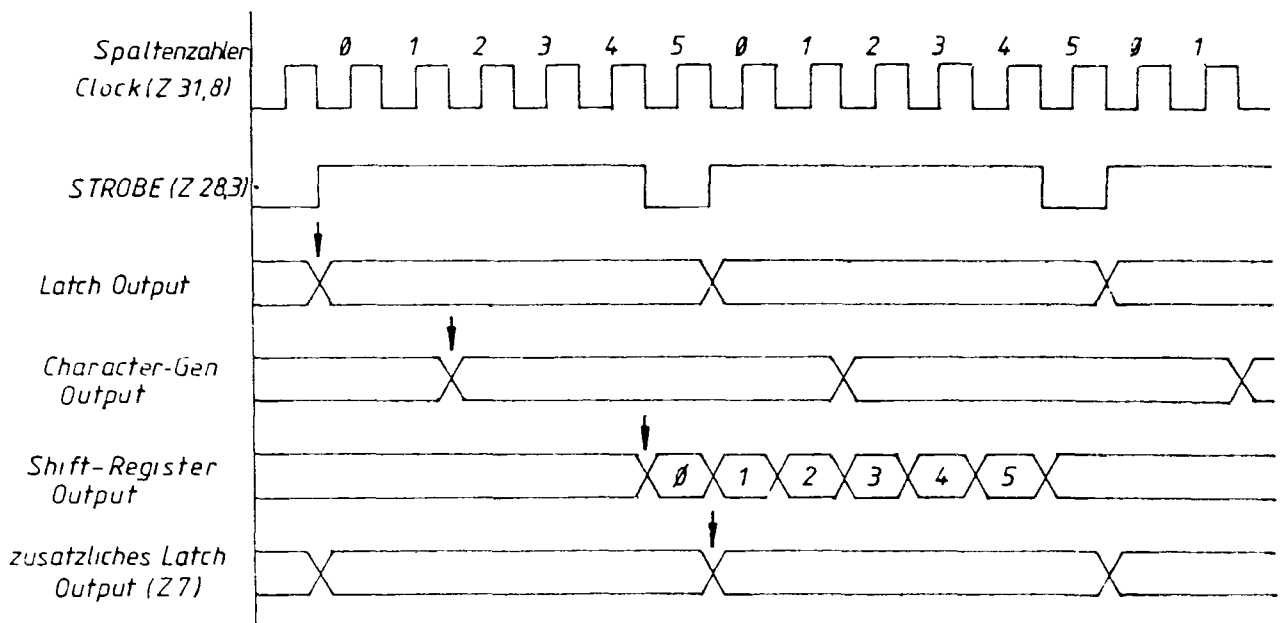


Fig. 7.5

Durch den Einbau des zusätzlichen Latches Z7 liegen die Daten zu der Zeit an, wo das Zeichen aus dem Shiftregister geschoben wird. Die zeitliche Verschiebung um einen Bildpunkt (siehe Fig. 7.4) wird dadurch ausgeglichen, daß die Flanke von der 1. zur 2. Spalte (und nicht die von der 2. zur 3.) zur Steuerung benutzt wird.

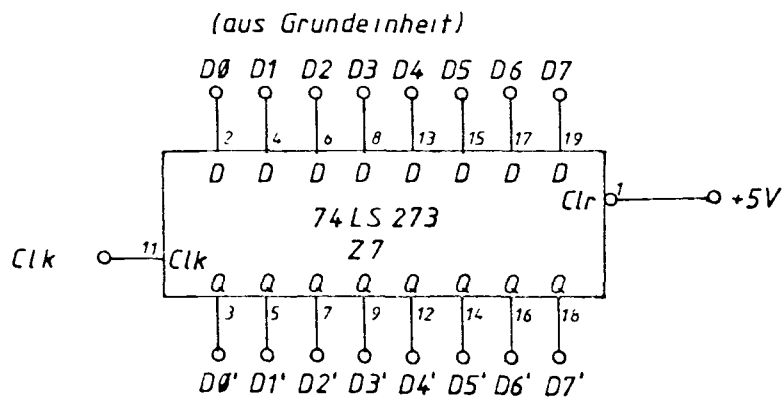


Fig. 7.6

### 7.4.2 Schaltzeichendecoder

Der Schaltzeichendecoder decodiert die am zusätzlichen Latch anliegenden Daten. Er generiert die Signale  $\overline{CTRL}$  und CTRL, die

- die Strichgraphik enablen
- den Zeichengenerator disablen  
(bei einem Zeichen der Strichgraphik soll kein anderes Zeichen ausgegeben werden).

Als Steuerzeichen gelten alle Daten mit Bit 5,6,7 = "0", das sind alle ASCII-Zeichen von 00H bis 1FH.

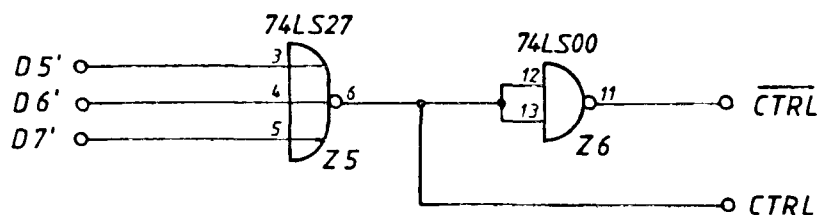


Fig. 7.7

Damit diese Zeichen überhaupt im Bildschirmwiederhol-speicher anliegen, müssen Software und Hardware des Systems geändert werden:

- Das Original Basic-ROM wandelt softwaremäßig alle Buchstaben in Zeichen von 00H bis 1FH um und gibt diese an das Video-RAM.

ROM 1 wird gegen ein Eprom ausgetauscht  
(Speicherstelle 0473H: 18H statt 38H)

Ohne diese Änderung wäre das Original-Basic nach Einbau der Strichgraphik nicht mehr lauffähig.

- Das Bit 5 des Datenbus wird hardwaremäßig invertiert, wenn ein Zeichen von 00H bis 1FH ins Video-Ram geschrieben werden soll. So wird jetzt das

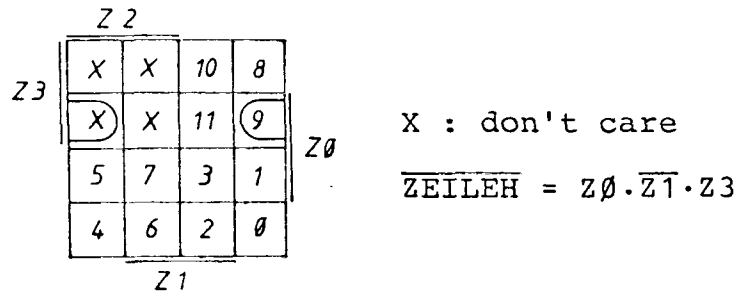
Bit 5 des Datenbus unmittelbar auf das Video-RAM durchgeschaltet.

7.4.3 Zeilendecoder (horizontaler Strich)

Der horizontale Strich soll in Zeile 9 generiert werden. Z33 der Grundeinheit liefert den Code der augenblicklich adressierten Zeile. Dieser braucht dann nur noch decodiert zu werden:

Es treten nur die Zeilen 0...11 auf. Alle anderen sind "don't cares".

Im Karnough-Diagramm gilt dann:



Als Schaltung ergibt sich dann:

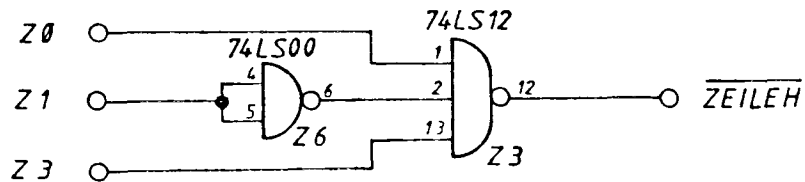


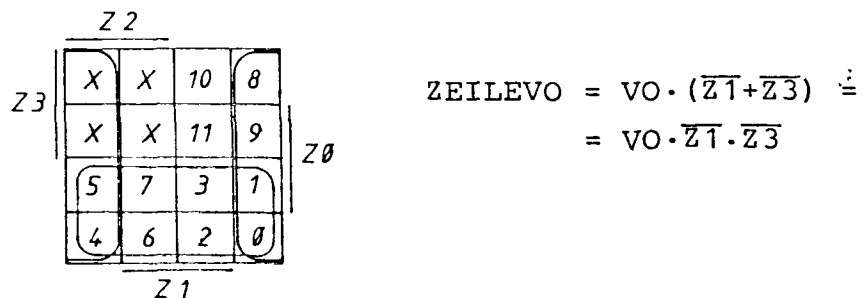
Fig. 7.8

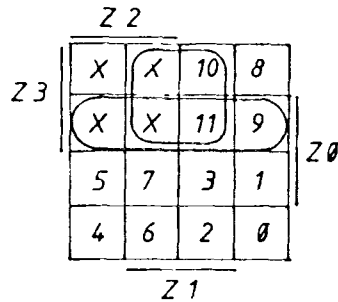
7.4.4 Zeilendecoder (vertikale Striche)

Für die vertikalen Striche sollen

- die Zeilen 0 - 9 für vertikal nach oben
- die Zeilen 9 - 11 für vertikal nach unten

decodiert werden.





$$\begin{aligned} \text{ZEILEVU} &= \text{VU}(\text{Z3} \cdot \text{Z1} + \text{Z3} \cdot \text{Z0}) = \\ &= \text{VU} \cdot \text{Z3} \cdot (\text{Z1} + \text{Z0}) \end{aligned}$$

Damit ergibt sich:

$$\text{ZEILEV} = \text{ZEILEVU} + \text{ZEILEVO}$$

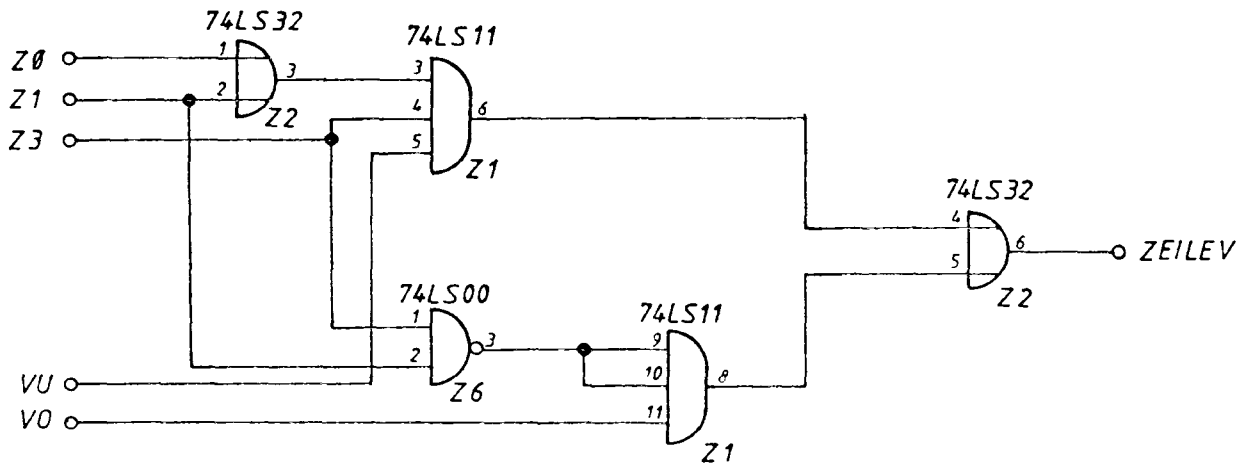


Fig. 7.9

#### 7.4.5 Spaltendecoder

Alle Umschaltungen erfolgen in Spalte 2. Es ist dabei aber zu beachten, daß nach Anlegen der Spaltenadresse die Gatterlaufzeiten bis zur Ausgabe des Video-Signals so kurz wie möglich gehalten werden (die Breite eines Bildpunktes beträgt 65,1 ns, während die Gatterlaufzeiten im Bereich von 20 ns liegen).

Der Spaltendecoder ist nur dann enabled, wenn ein CTRL-Zeichen (00H-1FH) anliegt und der Zeilenzähler die richtige Zeile decodiert hat.

Der Spaltenzähler Z36 zählt von 0 bis 5. Decodiert werden soll Spalte 2:

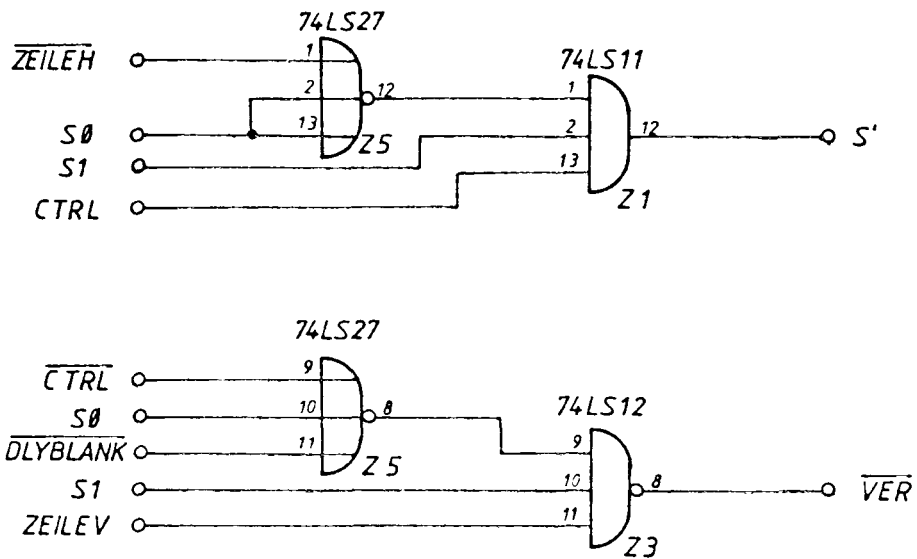
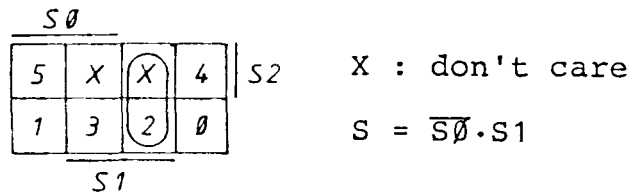
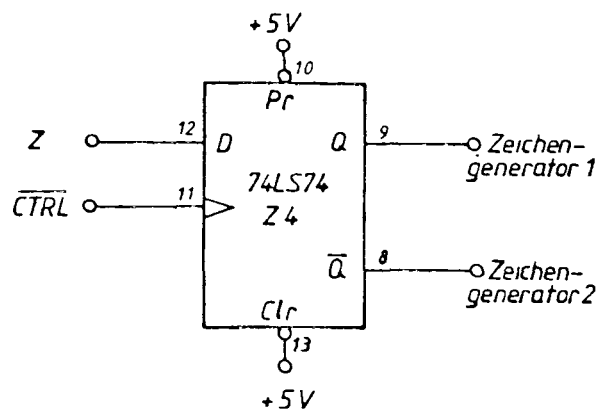


Fig. 7.10

### 7.4.6 Flipflops

#### 7.4.6.1 Zeichengeneratorschaltung

Ist das Z-Bit gesetzt, so wird auf den zweiten Zeichengenerator umgeschaltet:



### 7.4.6.2 Horizontaler Strich

Ist das H-Bit gesetzt, so wird von diesem Bildschirmplatz an ein horizontaler Strich eingeschaltet, ist es zurückgesetzt, so wird er abgeschaltet.

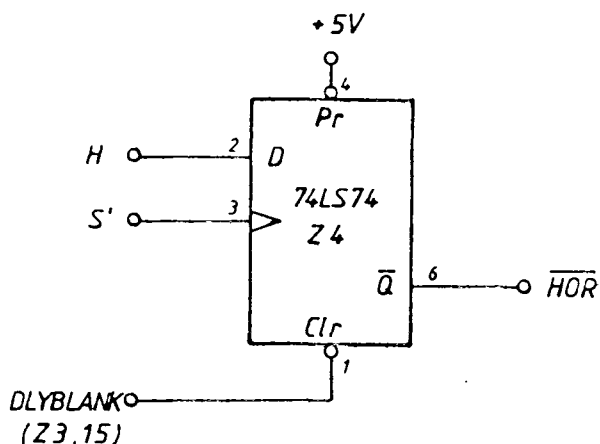


Fig. 7.12

### 7.4.7 Mischer

Die Signale  $\overline{HOR}$ ,  $\overline{VER}$  werden ausgegeben; das Zeichen aus dem Zeichengenerator VD alt wird bei  $\overline{CTRL}$  nicht durchgeschaltet.

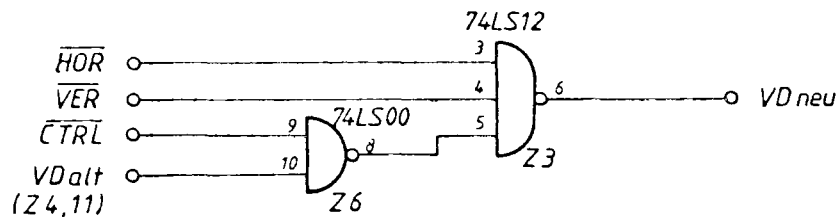


Fig. 7.13



## 7.5 Software

### 7.5.1 Ausgabe auf Bildschirm

Neben dem Einbau der Hardware sind noch einige Änderungen der Software erforderlich. In der OMIKRON-Version filtert das CBIOS bei der Ausgabe auf den Bildschirm alle Zeichen zwischen 00H und 1FH heraus und benutzt sie als Steuerzeichen bzw. unterdrückt sie.

Dies läßt sich umgehen durch die Definition eines Umschaltzeichens (01H). Es bewirkt, daß das folgende Zeichen in ein Steuerzeichen (00H - 1FH) umgewandelt und in den Video-Speicher geschrieben wird. Die dazu erforderlichen Routinen sind als Unterprogramme GRAPH1 und GRAPH2 ins Unterprogramm PCONOU aufgenommen worden.

Eine derartige Ausgabe ist nur möglich, wenn das BDOS über die "Console-Output"-Routine angesprochen wird. Bei unmittelbarer Eingabe von der Tastatur werden die Steuerzeichen, soweit sie nicht Steuerfunktion besitzen, schon im BDOS herausgenommen und in einen "5E-X"-String umgewandelt. Dabei ist X das ASCII-Zeichen des Buchstabens, der als CTRL-X eingegeben wurde:

z.B. 5E - 41(A) für 01 (CTRL A)  
5E - 42(B) für 02 (CTRL B).

Nachfolgend werden kurz die für die Strichgraphik ins CBIOS eingefügten Routinen beschrieben.

Die Speicherstelle 0044H dient als Statuswort für die Strichgraphik. Das Bit 0 ist gesetzt, wenn das letzte Zeichen ein 01H war und damit das folgende als CTRL-Zeichen ausgegeben werden soll. Nach der Ausgabe des zweiten Zeichens wird dieses Bit zurückgesetzt.

Unmittelbar zu Beginn der Ausgaberroutine PCONOU befindet sich die erste Abfrage. War das letzte Zeichen das Um-

schaltzeichen 01H, so wird zunächst die "Delete-Backspace"-Routine übersprungen (Sprung nach L00030).

Die zweite Abfrage erfolgt an Adresse FD35H. War das letzte Zeichen ein 01H, so erfolgt ein Sprung zu GRAPH2, das folgende Eigenschaften aufweist:

#### GRAPH2

Dieses Unterprogramm setzt das Bit 0 des Graphik-Statuswortes zurück, wandelt das im A-Reg. vorhandene aktuelle Zeichen in ein CTRL-Zeichen um, gibt dieses aus und bereitet einen Rücksprung nach RUECK vor, um den Cursorstand zu aktualisieren.

Die dritte Abfrage befindet sich bei der Decodierung der Steuerzeichen an Adresse FDE7H. Hier wird getestet, ob das aktuelle Zeichen ein Steuerzeichen ist. In diesem Fall erfolgt ein Sprung zu GRAPH1 mit folgenden Eigenschaften:

#### GRAPH1

Dieses Unterprogramm setzt das Bit 0 des Graphik-Statuswortes (damit das folgende Zeichen als Steuerzeichen ausgegeben wird) und kehrt - wie alle Steuerzeichendecodiererroutinen - nach RUECK zurück, um den Cursorstand zu aktualisieren.

#### 7.5.2 Ausgabe auf Drucker

Das auf ein 01H folgende Zeichen darf nicht an den Drucker ausgegeben werden, da er es als Steuerzeichen zur Wahl der Schriftbreite o.ä. decodieren würde. Deshalb wird die LIST-Routine des BIOS folgendermaßen erweitert:

Zu Beginn wird abgefragt, ob das letzte oder das aktuelle Zeichen ein 01H war bzw. ist. Ist dies der Fall, erfolgt eine Rückkehr aus LIST, ohne daß etwas ausgegeben wird. Das Umschaltzeichen und das darauf folgende werden von der Druckeroutine unterdrückt.

7.6 Benutzungsbeispiel

Einige Beispiele sollen die Möglichkeiten der Strichgraphik verdeutlichen:

Eine unmittelbare Eingabe über die Tastatur ist, wie in 7.5.1 beschrieben, nicht möglich. Die Strichgraphik kann angesprochen werden über die Console-Output-Routinen des BDOS, oder indem man einen vom Editor (ED) geschriebenen File mit dem "TYPE"-Befehl ausgibt.

In den folgenden Beispielen sind die Bildschirmausgabe und die dazu benötigten Zeichen dargestellt, so wie sie bei Eingabe von der Tastatur auf den Bildschirm zu sehen sind (CTRL A = †A usw.); dabei ist †A als e i n Zeichen (Ø1H) abgespeichert, obwohl auf den Bildschirm z w e i ausgegeben werden.

Unterstreichung:

Es ist <u>sehr</u> warm	Es ist †Asehr †Ahwarm
	od. Es ist †Aasehr †Ahwarm
	od. Es ist †Aisehr †Ahwarm
	od. Es ist †A †Asehr †A †Hwarm

Einrahmung:

<u>UEBERSCHRIFT</u>	†AE	†AD
	†ACUEBERSCHRIFT	†AB

Betragsbildung:

a + b  =  a+b	†AFa †AF+ †AFb †AF = †AFa+b †AF
---------------	---------------------------------

Tabellierung:

Name	Menge	Preis	†AAName	†AG Menge	†AG Preis	†AF
Cola	100	1.50	Cola	†AF 100	†AF 1.50	†AF
Fanta	80	1.40	Fanta	†AF 80	†AF 1.40	†AF

## 8. Zusammenfassung

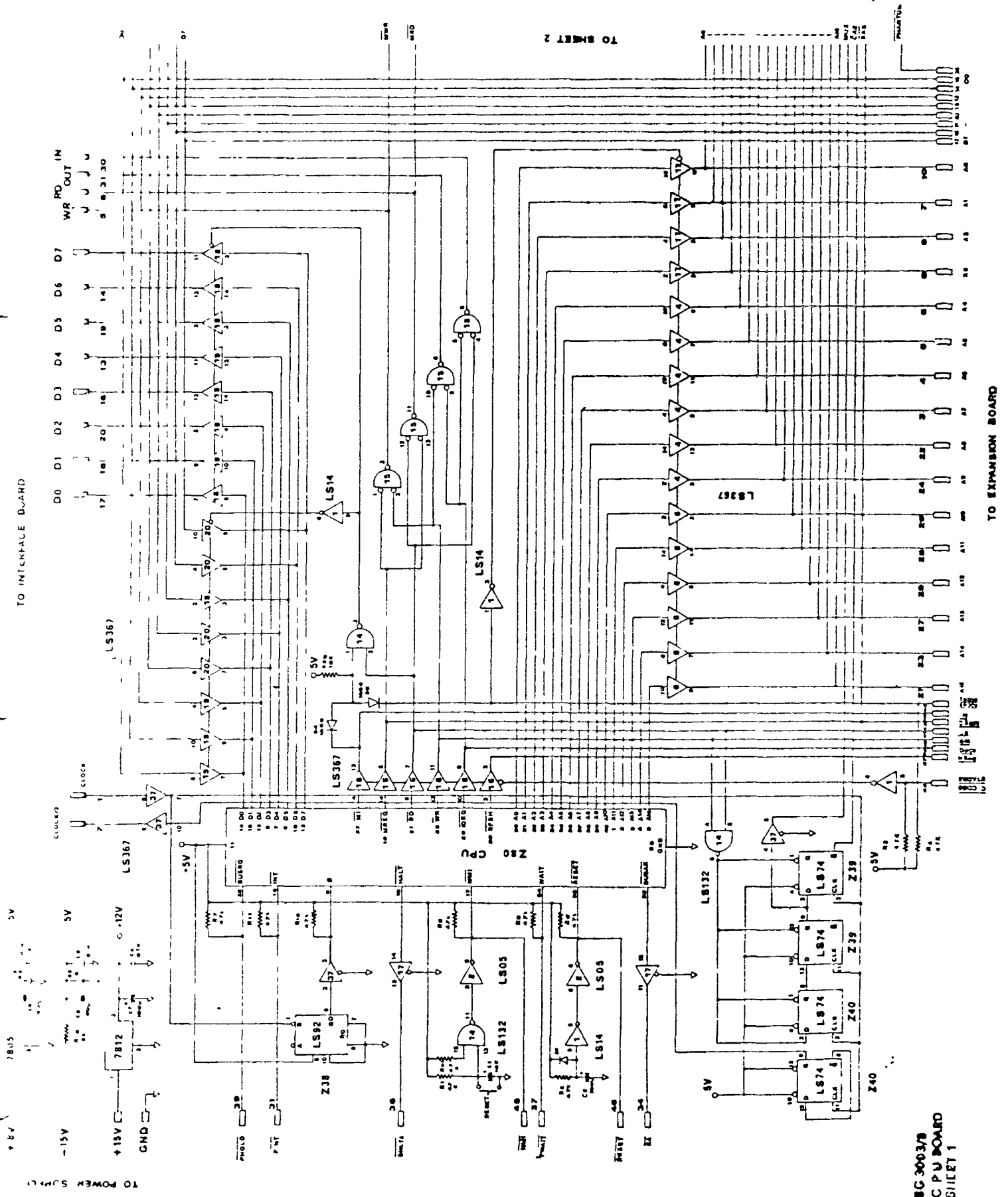
Nach dem Einbau der Speichererweiterung und der Strichgraphik ist auf dem Video-Genie eine 63k-CPM-Version lauffähig, wobei auch die Möglichkeiten der Strichgraphik genutzt werden können. Wenn noch der Mapper II von OMIKRON in den Expander eingebaut wird, können auch 8"-Disketten benutzt werden.

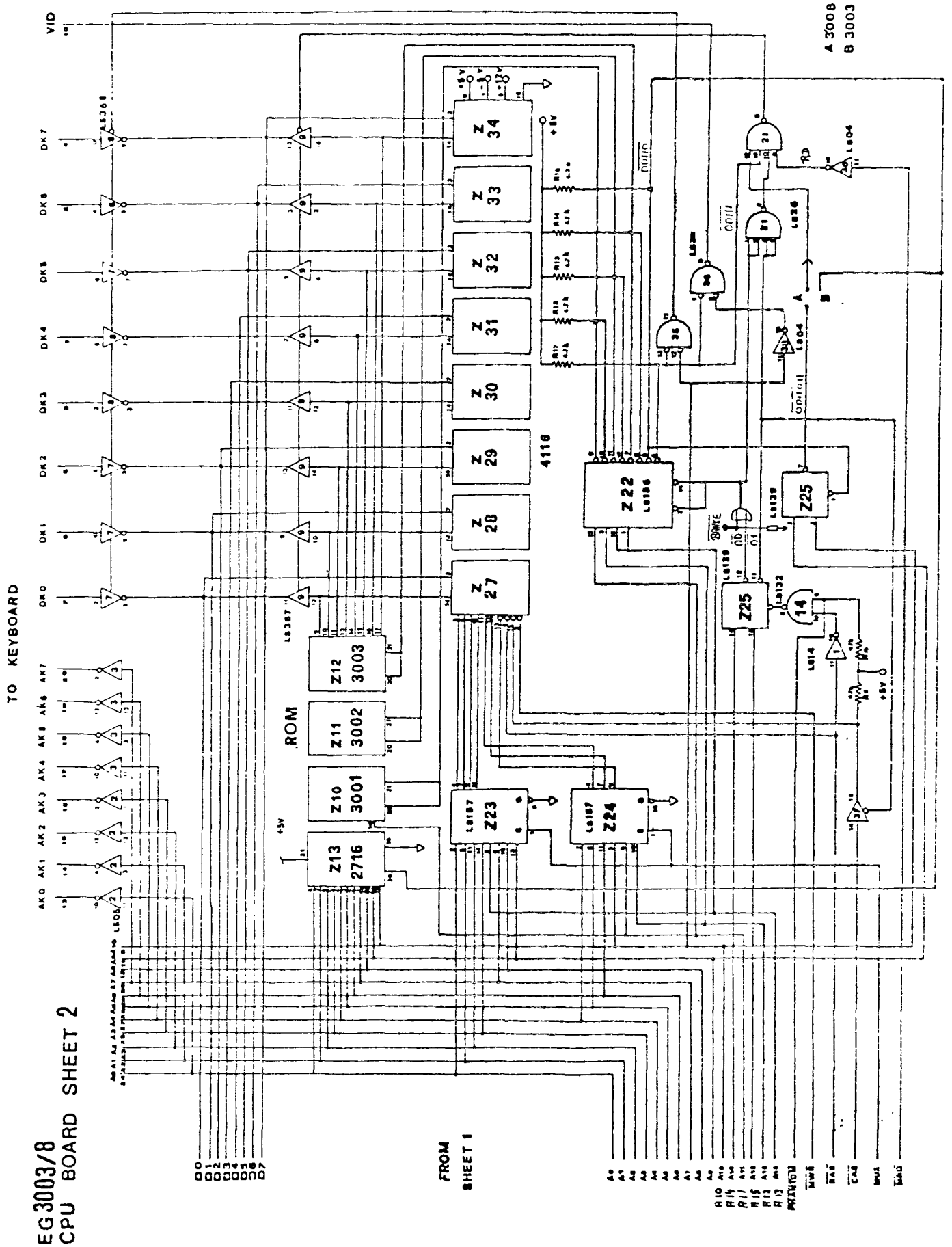
Nach geringen Änderungen des BIOS und Eingriffen in den Drucker läßt sich auch dieser für die Strichgraphik umrüsten.

9. Literaturverzeichnis

1. Video Genie System  
Service Manual  
Trommeschläger Computer GmbH
2. The Omikron Mapper  
Owners Manual, 09.01.80
3. An Introduction to CP/M Features and Facilities  
Digital Research 1976, 77, 78
4. CP/M 2.0 Alteration Guide  
Digital Research, 1979
5. CP/M 2.0 Interface Guide  
Digital Research, 1979
6. The TTL Data Book for Design Engineers, 1981
7. Z-80 Assembler Handbuch  
W. Hofacker, 1980

A N H Ä N G E



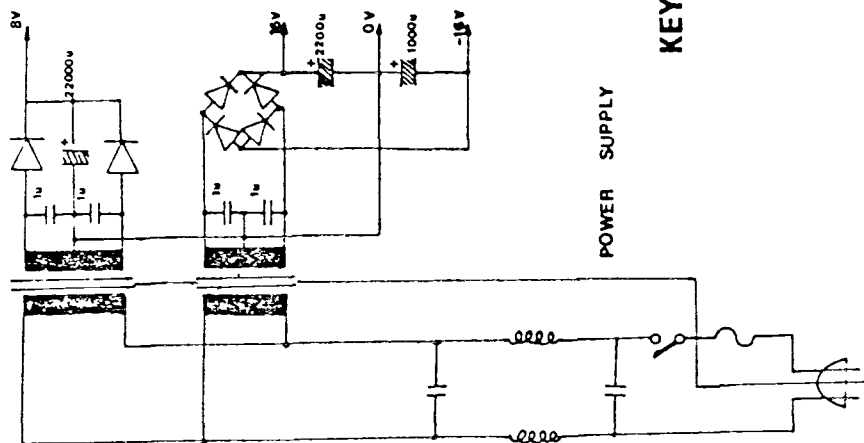
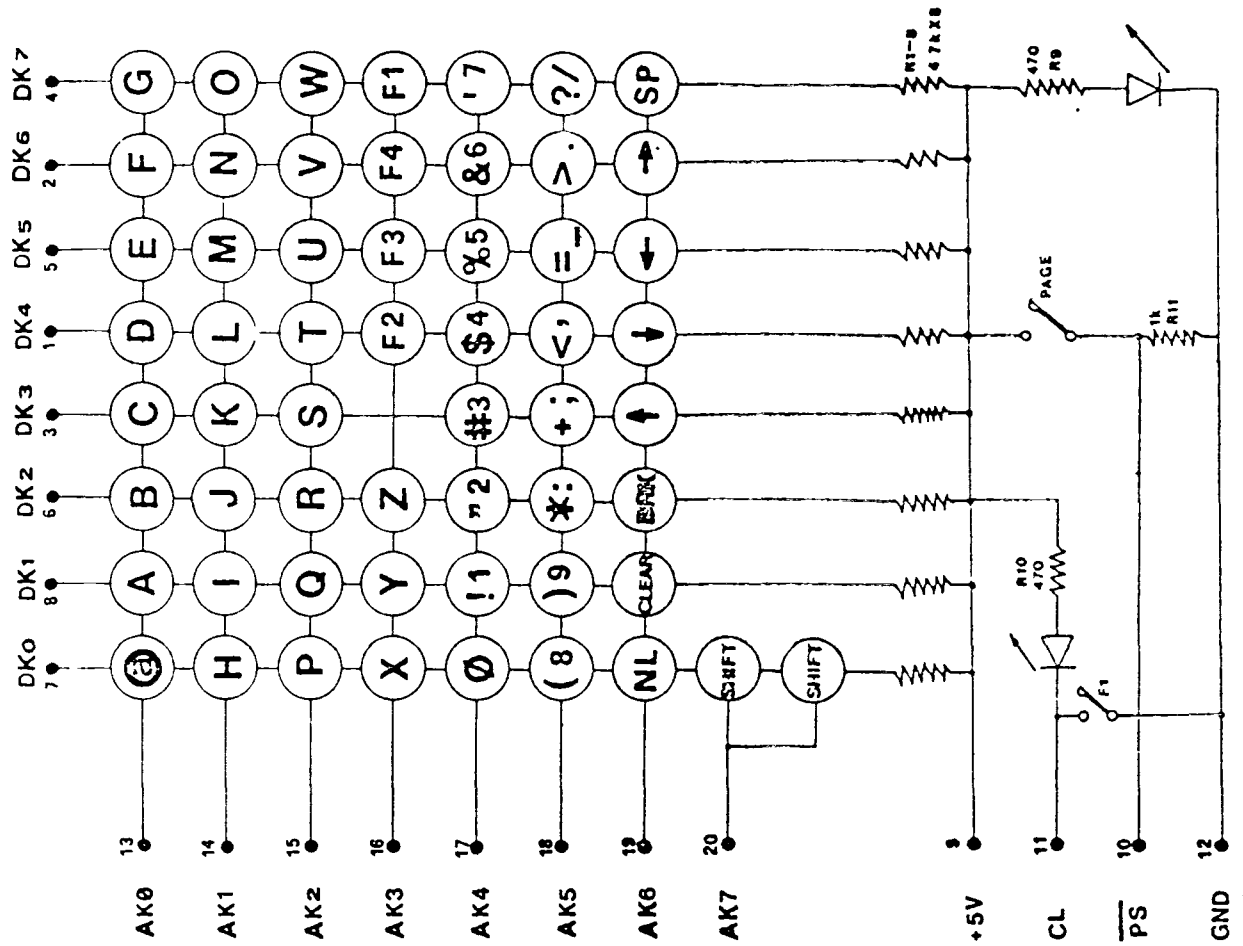


EG3003/8  
CPU BOARD SHEET 2

FROM  
SHEET 1

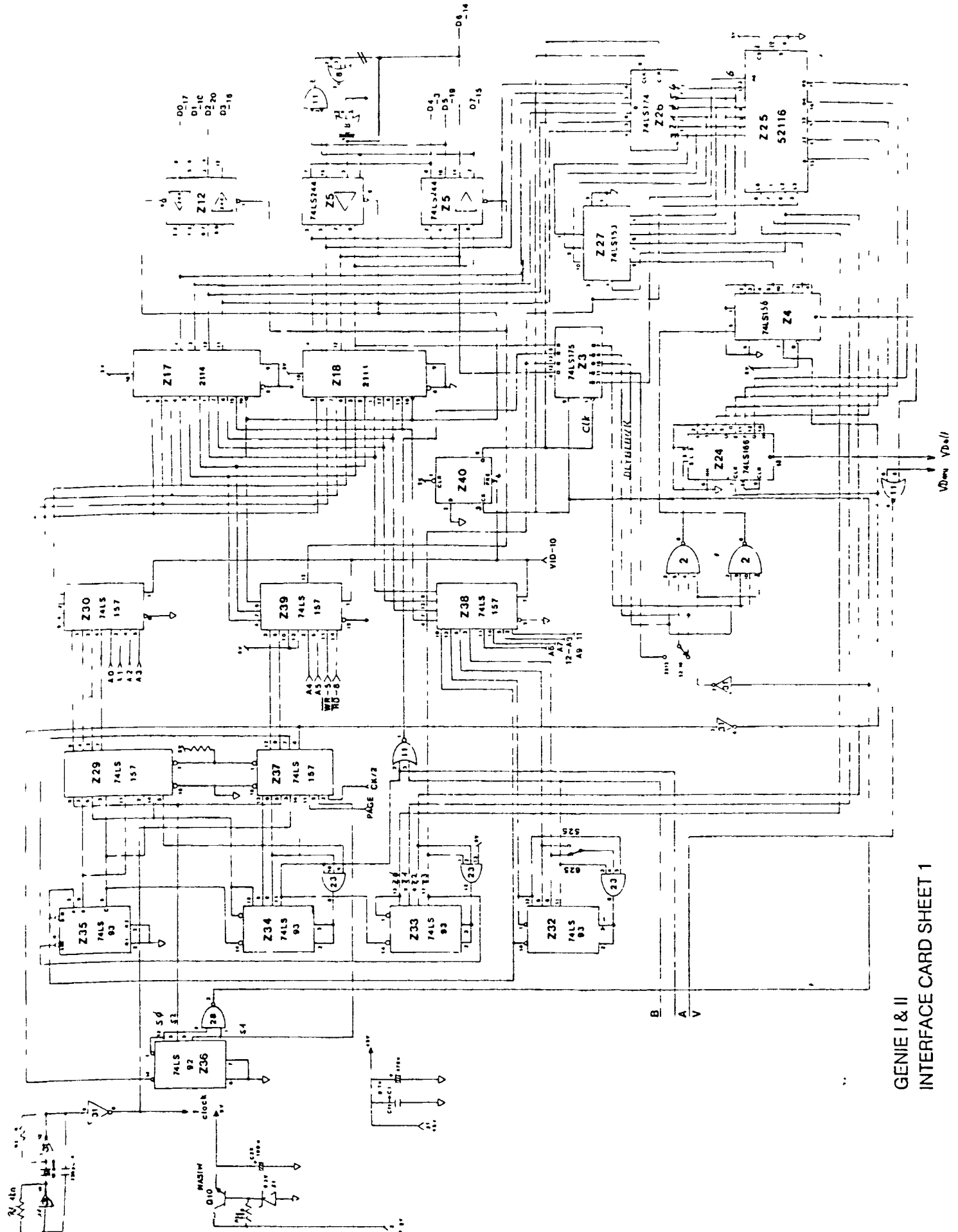
A 3008  
B 3003



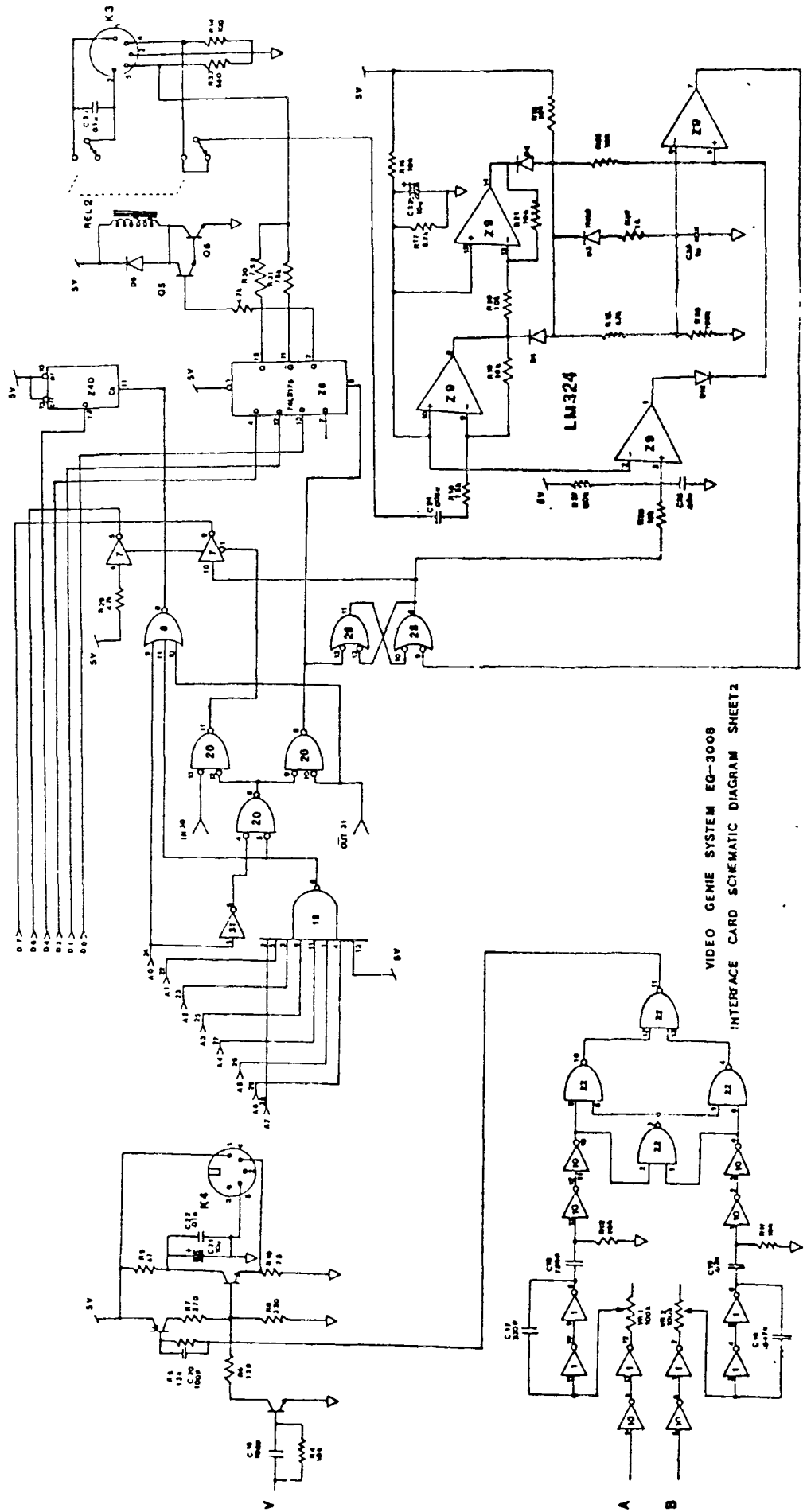


KEYBOARD

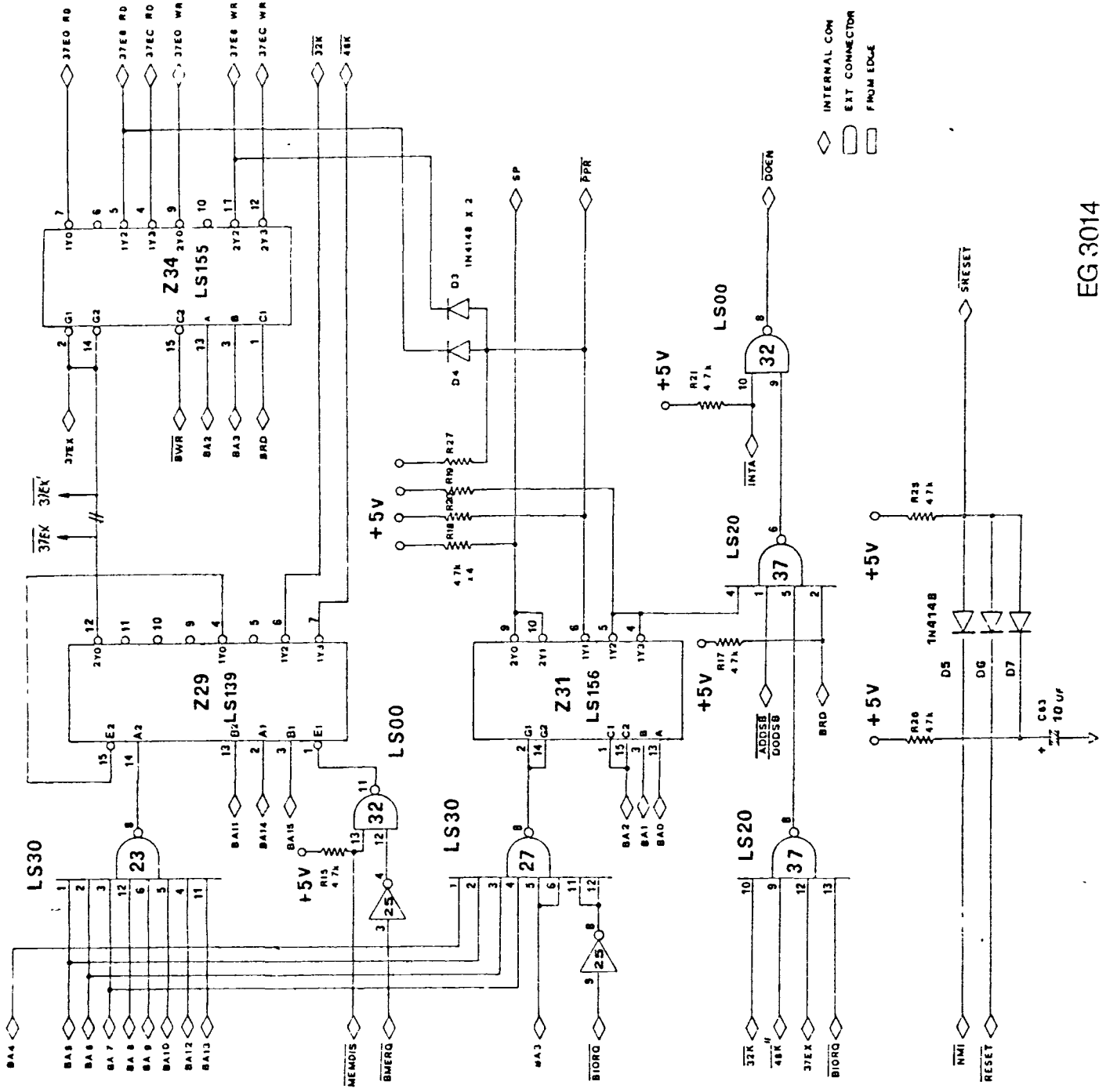
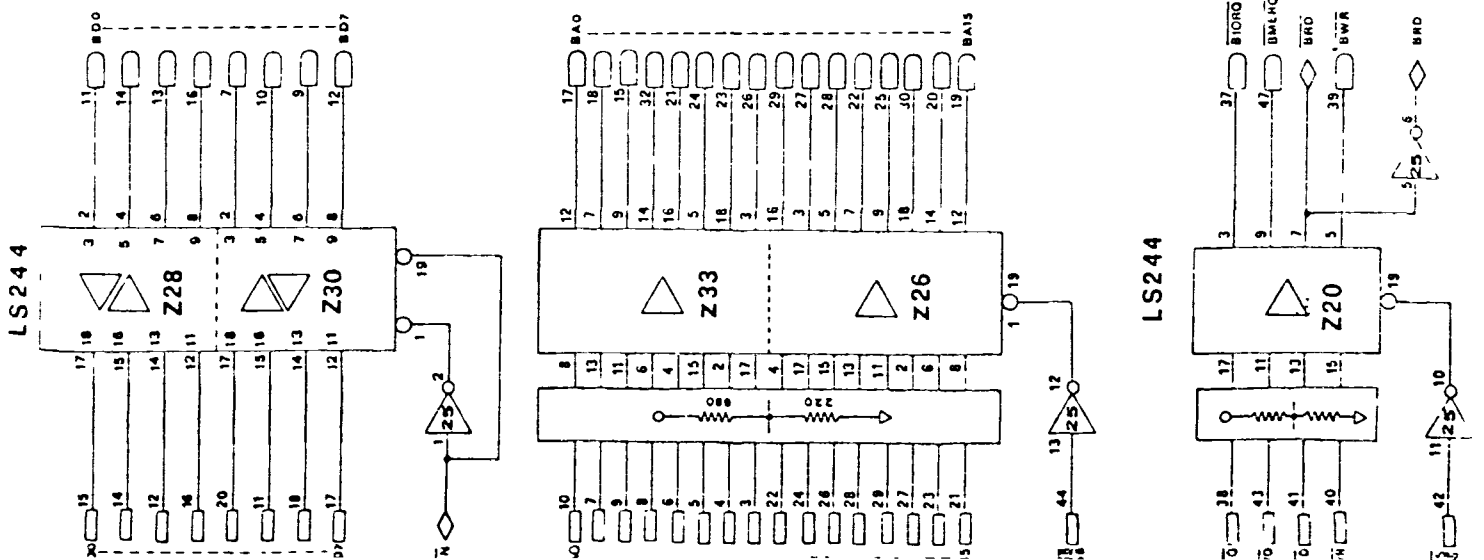
POWER SUPPLY



GENIE I & II  
INTERFACE CARD SHEET 1

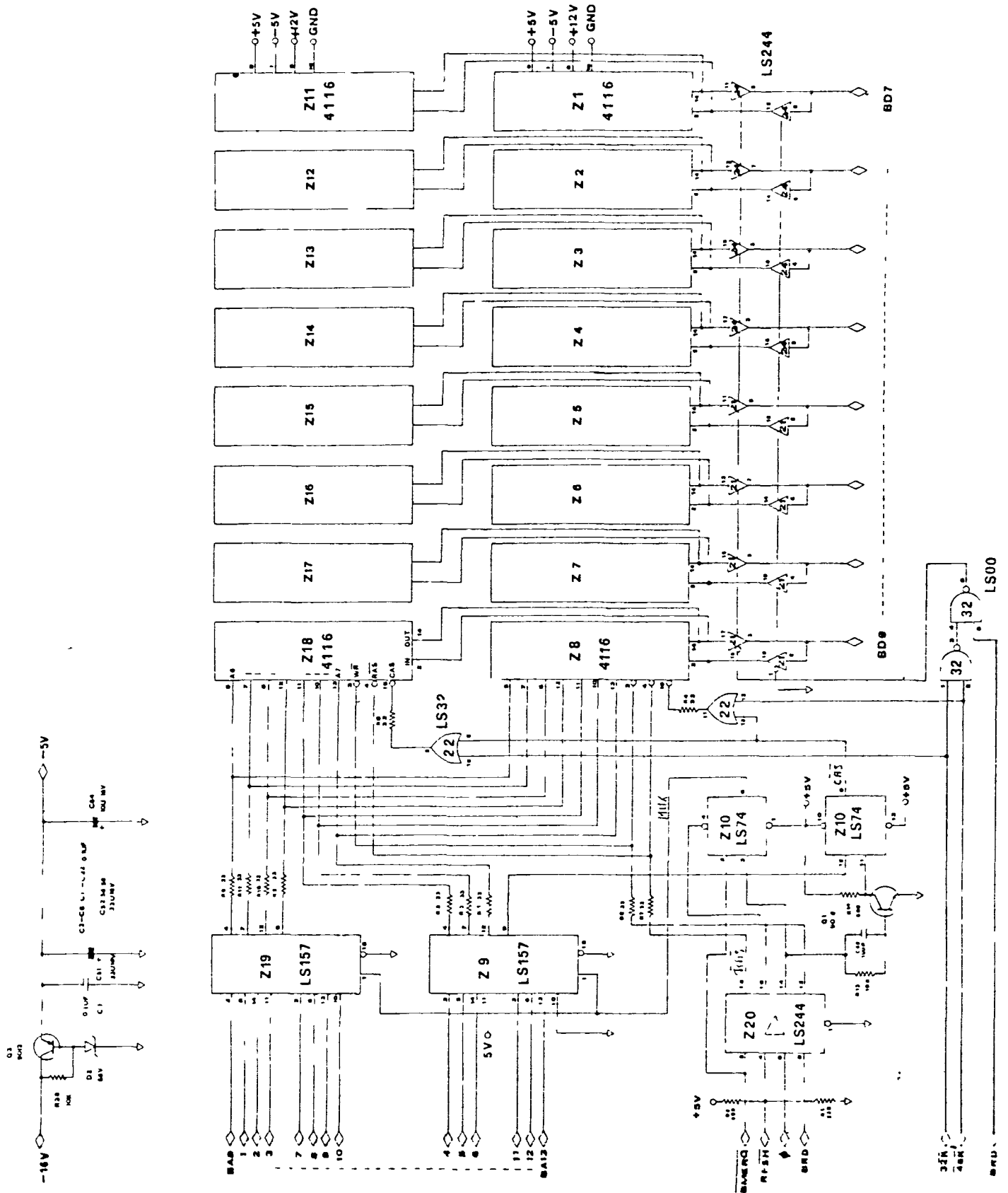


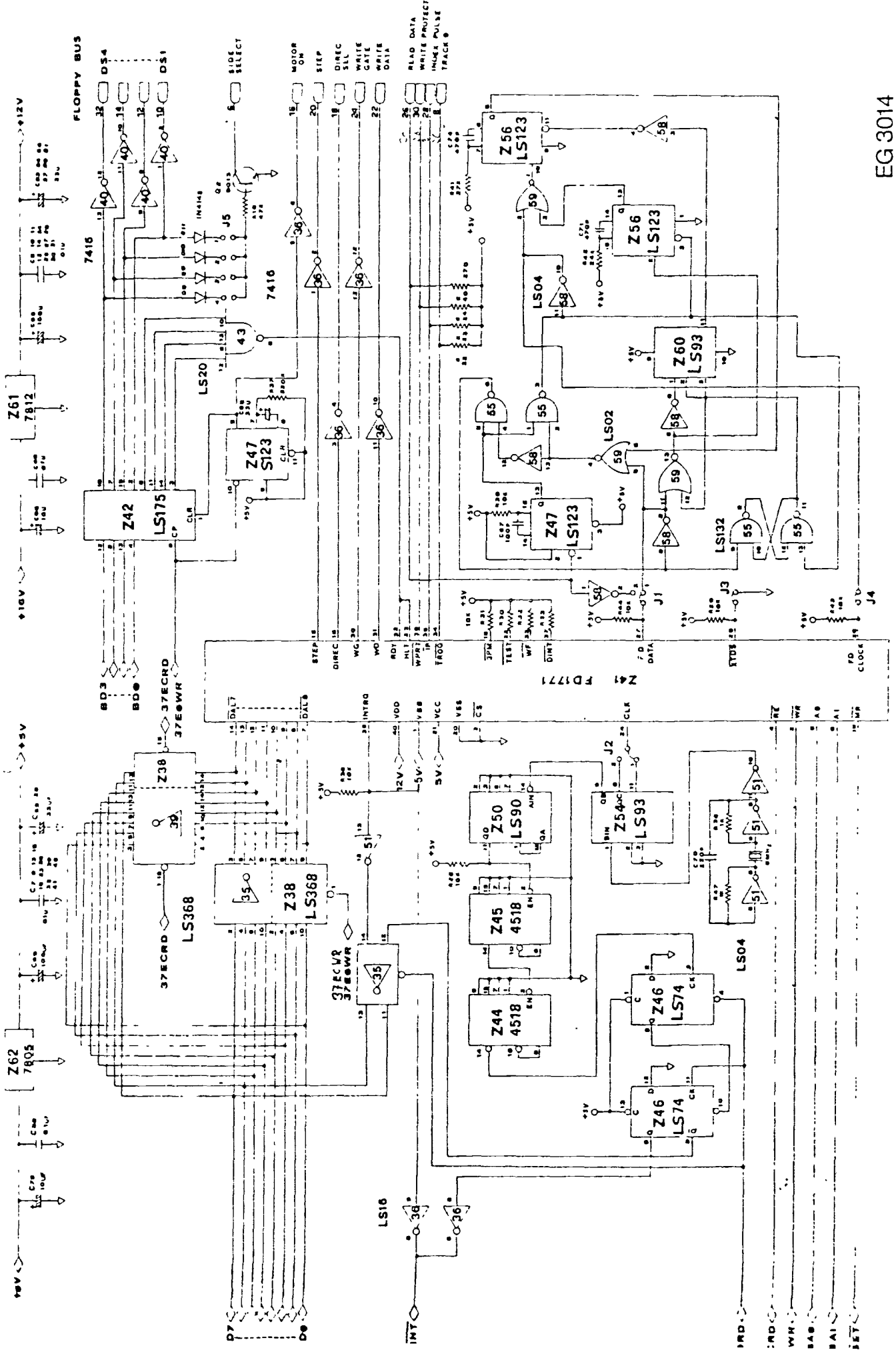
Schaltplan Expander-Platine (Blatt 1)

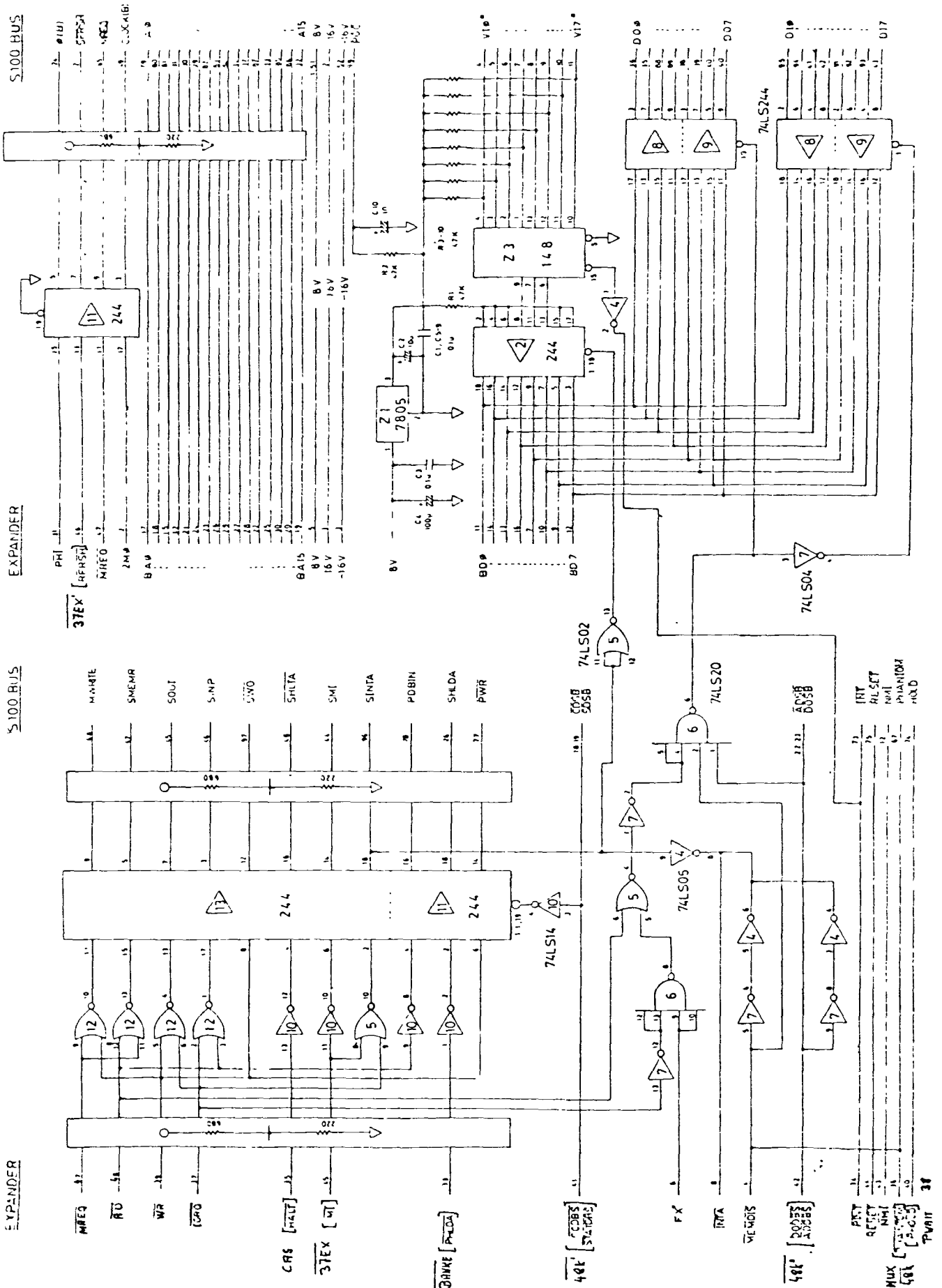


INTERNAL CON. □  
EXT CONNECTOR ○  
FROM EDGE ▭

EG 3014





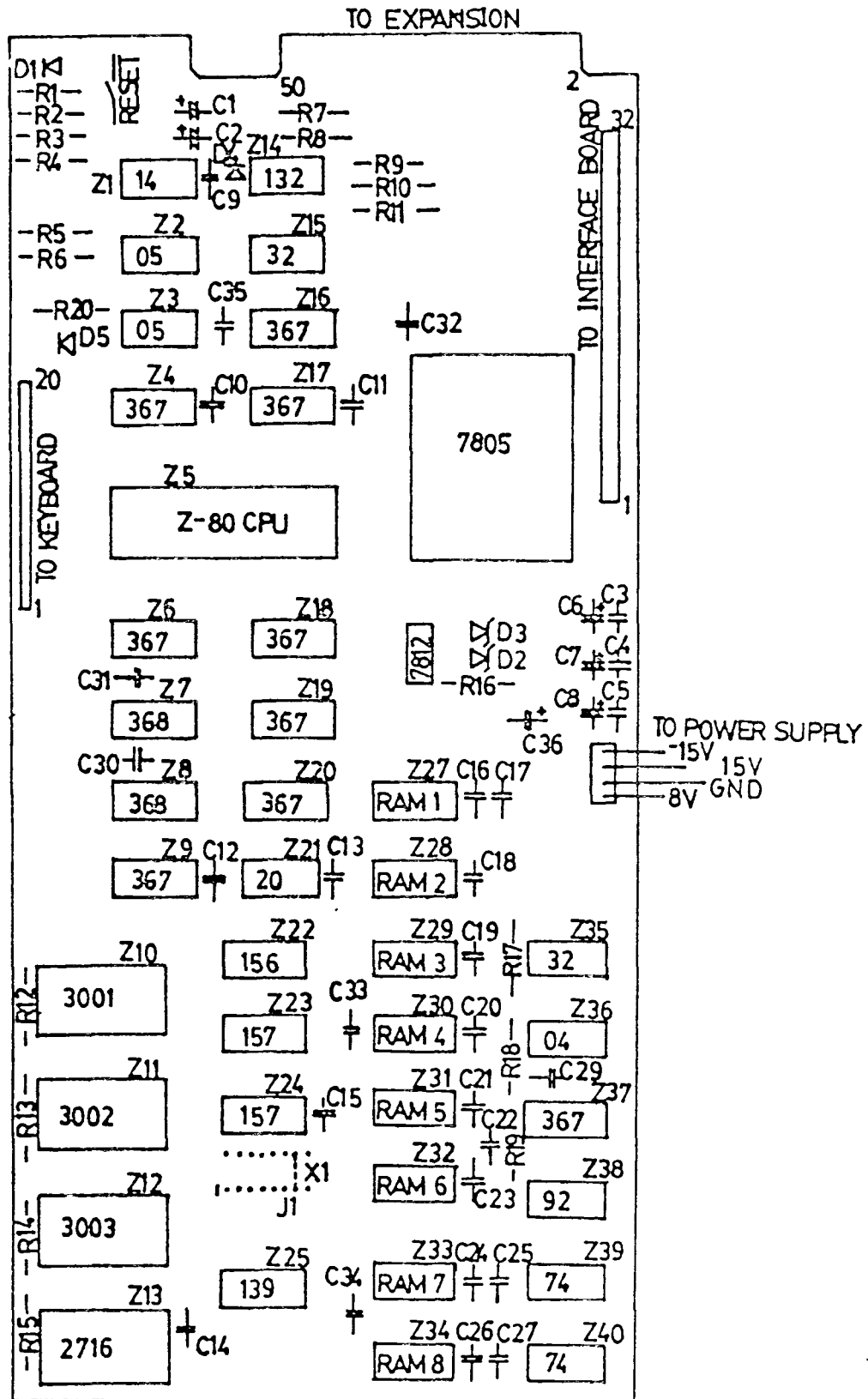


EXPANDER

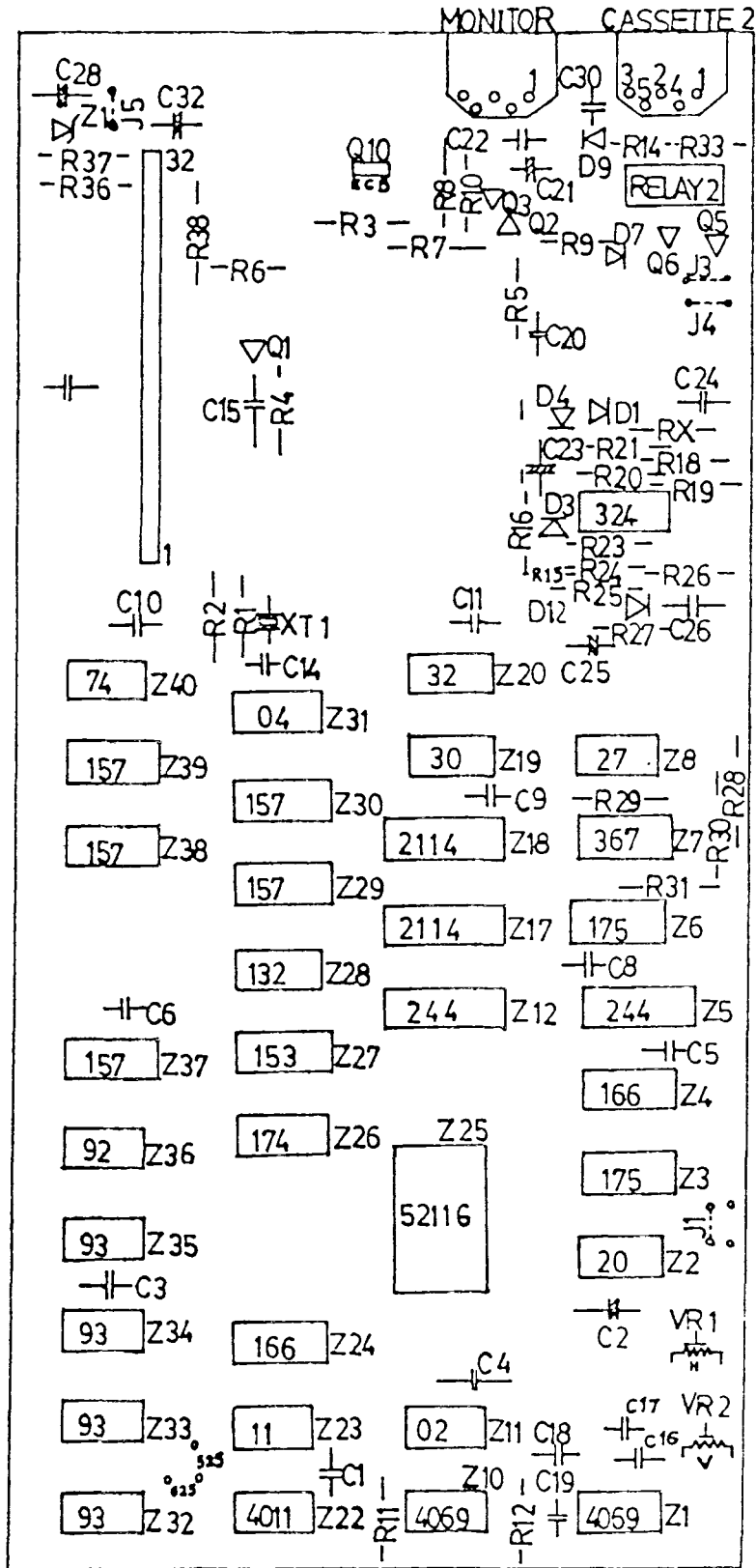
S100 BUS

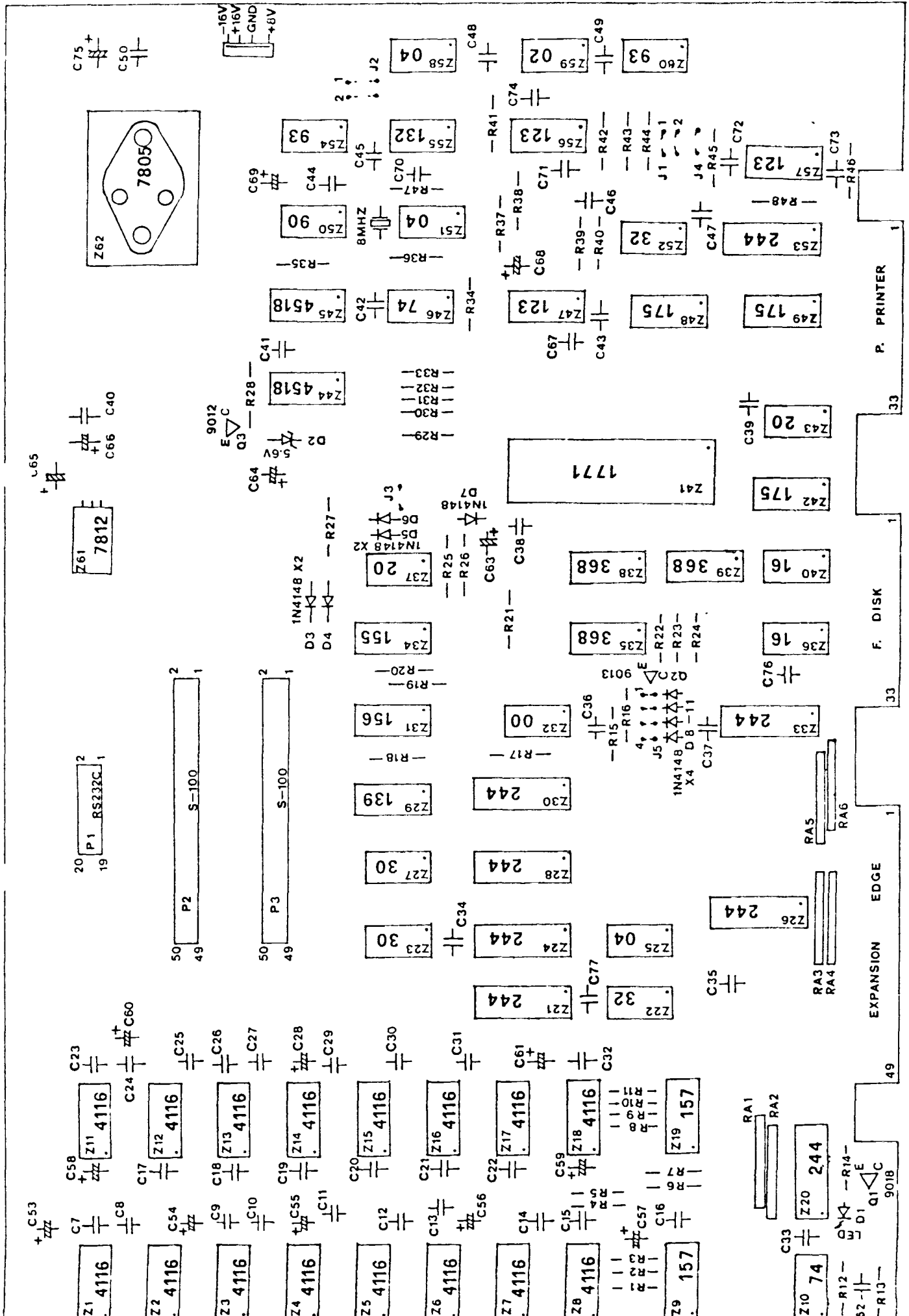
EXPANDER

S100 BUS









EG3014 EXPANDER

P. PRINTER

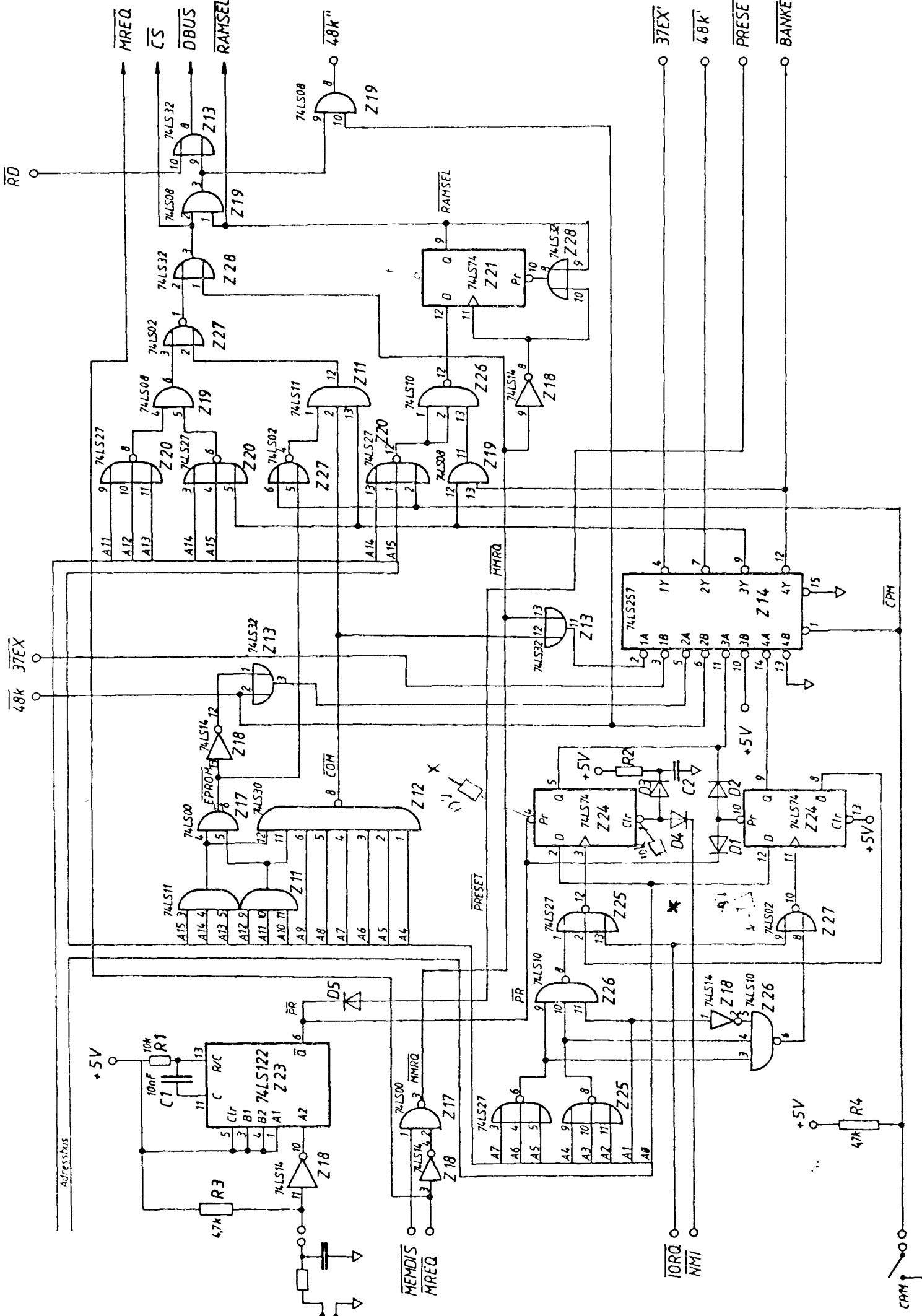
F. DISK

EXPANSION EDGE

EXPANSION EDGE

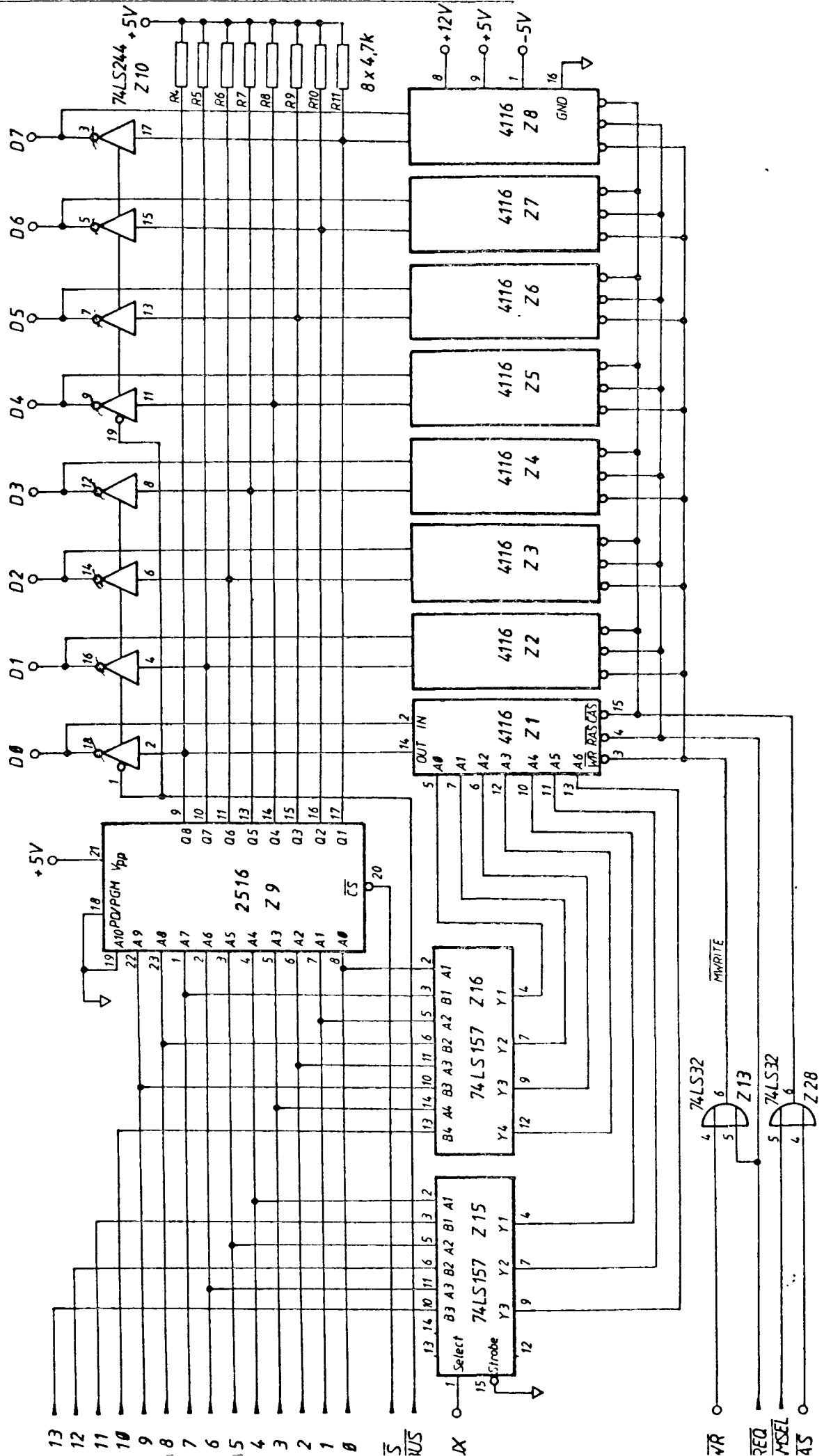
EXPANSION EDGE

EXPANSION EDGE



ADRESSBUS

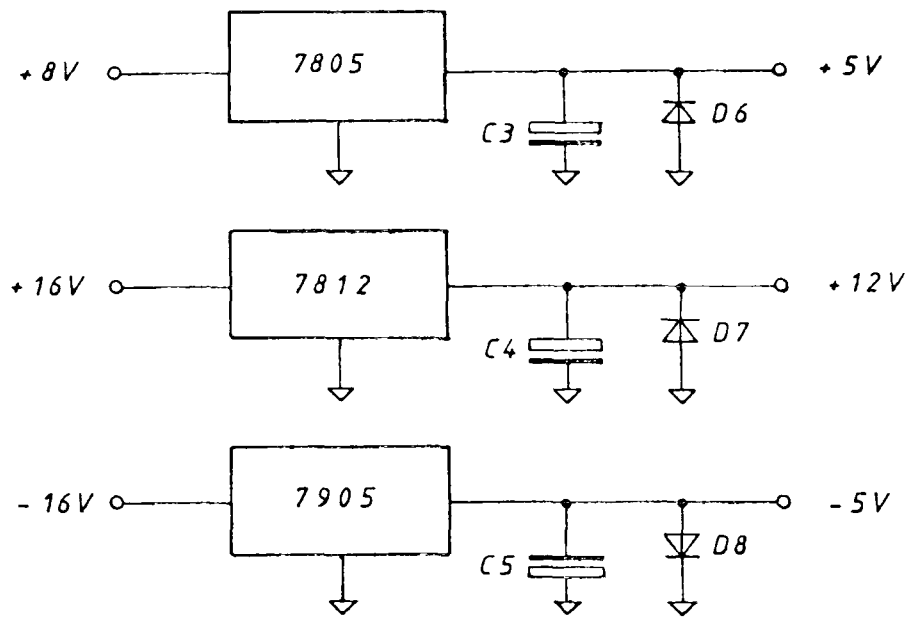
CPM

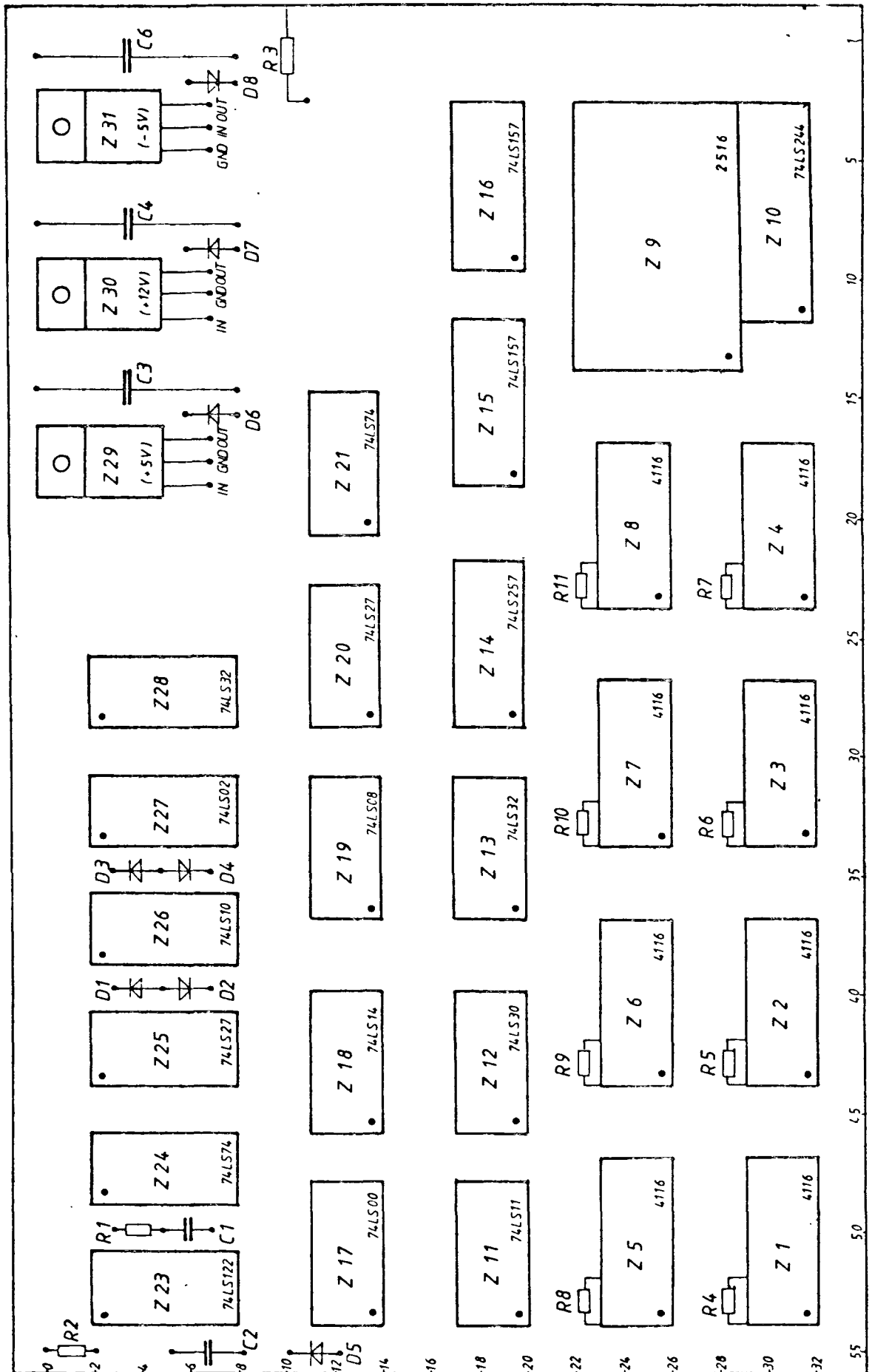


13  
12  
11  
10  
9  
8  
7  
6  
5  
4  
3  
2  
1  
0  
CS  
BUS

IX  
15  
1  
13  
14  
10  
11  
6  
5  
3  
2

NR  
REQ  
MSEL  
AS





Pinbelegung Steckerleiste (Speichererweiterung)

<u>Pin</u>	<u>S100-Bus</u>	<u>Bezeichnung</u>
1a	49,50	GND
1c	49,50	GND
2a	5	+8V
2c	5	+8V
3a	1	+16V
3c	1	+16V
4a	3	-16V
4c	3	-16V
5a		
5c		
6a		
6c		
7a		
7c		
8a		
8c		
9a		
9c		
10a		
10c		RESET
11a		CPM/RESET
11c		
12a		CPM
12c	33*	BANKE
13a		
13c	46*	37EX'
14a	45*	37EX
14c	42*	48k''
15a	41*	48k'
15c	40*	48k
16a	36*	MUX
16c	35*	CAS

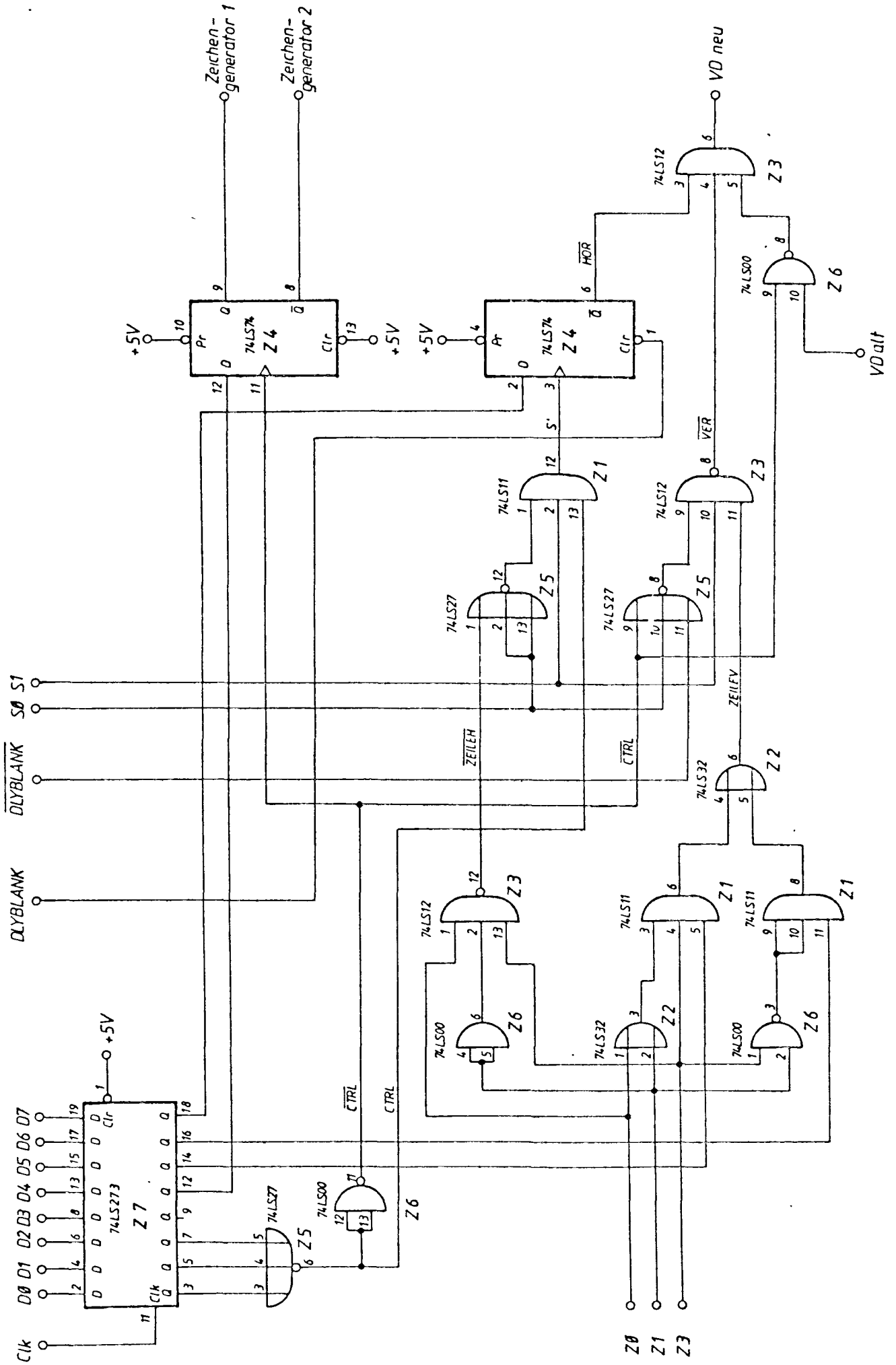
*Taster*

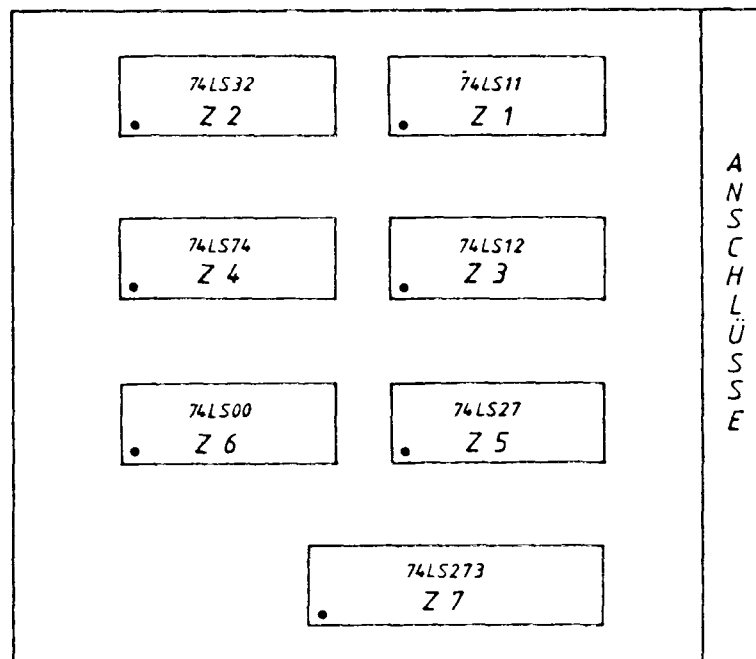
*Schalter*

<u>Pin</u>	<u>S100-Bus</u>	<u>Bezeichnung</u>
17a	44	$\overline{\text{PRESET}}$
17c	43	$\overline{\text{NMI}}$
18a	37	$\overline{\text{IORQ}}$
18c	48	$\overline{\text{RD}}$
19a	47	$\overline{\text{MREQ}}$
19c	4	$\overline{\text{MEMDIS}}$
20a	39	$\overline{\text{WR}}$
20c		
21a	12	D7
21c	9	D6
22a	10	D5
22c	7	D4
23a	16	D3
23c	13	D2
24a	14	D1
24c	11	D0
25a	17	A0
25c	18	A1
26a	15	A2
26c	32	A3
27a	21	A4
27c	24	A5
28a	23	A6
28c	26	A7
29a	29	A8
29c	27	A9
30a	28	A10
30c	22	A11
31a	25	A12
31c	30	A13
32a	20	A14
32c	19	A15

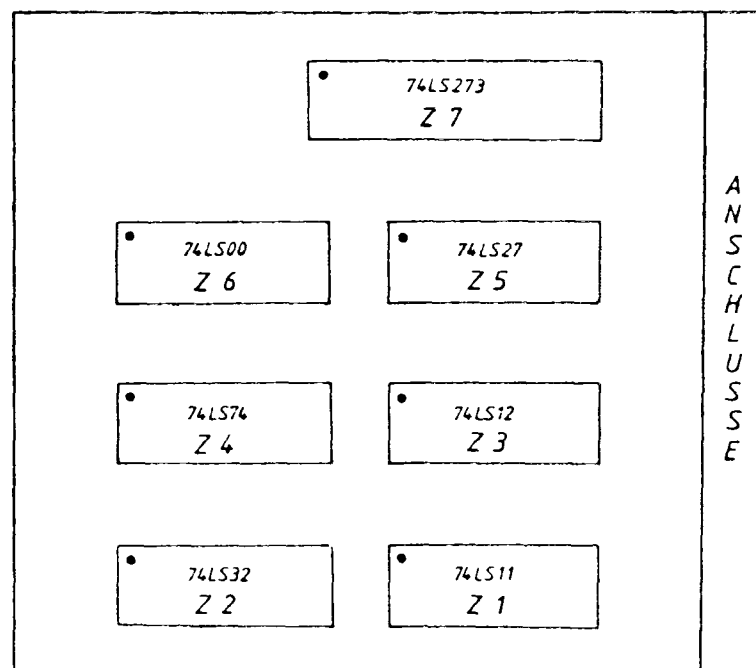
\* : im Expander geänderte Leitung







Vordersseite



Fadelseite