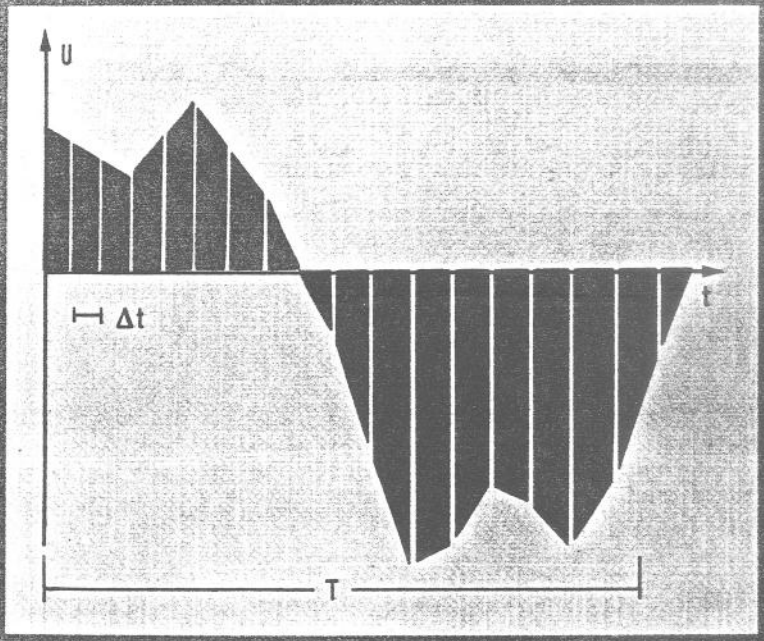
Achtung:
Textseiten mit 300 dpi gescannt, Platinenseiten und Layoutseiten mit 600 dpi gescannt !
Beim Ausdrucken der Platinenseiten aus Photoshop oder Adobe Reader heraus - unbedingt 100% wählen bzw. unter keinen Umständen "an Seite anpassen" benutzen, da sonst die Löcherabstände nicht mehr stimmen. Im Zweifelsfall Probeausdruck fertigen und IC-Fassung (möglichst lang) an Ausdruck halten und Abstände vergleichen!

Franzis
Computer-Praxis

Merker
Hardware-Erweiterungen für den ZX 81

Der nachträgliche Einbau von Schnittstellen zum Messen, Steuern und Regeln



chert ist. Nach Abarbeitung dieses Programms kehrt sie wieder zum ursprünglichen Programm zurück.

Die Interrupt-Bearbeitung kann softwaremäßig unterdrückt („maskiert") werden, so daß trotz Signal auf Leitung 16 keine Programmunterbrechung stattfindet.

Diese Leitung (Pin 17) hat dieselbe Funktion wie die $\overline{\text{INT}}$-Leitung, mit dem einzigen Unterschied, daß dieses Signal nicht blockiert werden kann.

l) $\overline{\text{WAIT}}$ (wait, Warten)
Es kann vorkommen, daß die CPU zu schnell für die angeschlossenen Speicher oder Peripheriebausteine ist. Dies können die Bausteine der CPU über ein L-Signal an Pin 24 signalisieren. Die CPU führt dann WAIT-Zyklen durch, bis die Bausteine soweit sind.

m) $\overline{\text{RESET}}$ (reset, Zurücksetzen)
Nach Einschalten der Betriebsspannung muß der Prozessor in einen definierten Ausgangszustand gebracht werden. Dies wird erreicht, indem der RESET-Eingang (Pin 26) kurzzeitig auf L-Pegel gelegt wird. Anschließend beginnt die CPU mit der Programmabarbeitung ab Adresse 0000.

n) $\overline{\text{BUSRQ}}$ (bus request, Busanforderung)
In gewissen Fällen kann es notwendig werden, daß die CPU die Kontrolle über die Adreß-, Daten- und Steuerbusleitungen aufgibt. Dies ist zum Beispiel der Fall, wenn Peripheriebausteine schnell große Datenmengen in den Speicher einlesen oder aus ihm heraus transferieren sollen (DMA: direct memory access, direkter Speicherzugriff).

Um dies zu erreichen, wird ein L-Signal an den $\overline{\text{BUSRQ}}$-Eingang (Pin 25) gegeben. Dadurch werden alle Tristate-Ausgänge der CPU hochohmig, so daß sie diese Leitungen nicht mehr beeinflußt.

o) $\overline{\text{BUSAK}}$ (bus acknowledgement, Bus-Anforderungsbestätigung)
Wenn eine Busanforderung von der CPU akzeptiert wird, signalisiert sie das durch L-Pegel auf Leitung 23. Jetzt kann die Peripherie auf die Leitungen zugreifen.

1.3.2 Programmabarbeitung durch die Z-80-CPU

Etwas vereinfacht betrachtet besteht die CPU aus einer Anzahl von Registern (d. h. Speichern) und einer Logik- und Verarbeitungseinheit. In *Abb.1.4* ist diese Registerstruktur der Z-80-CPU gezeigt. Um zu verstehen, wie ein Maschinenprogramm abläuft und wie Speicherbausteine bzw. Erweiterungsschaltungen angesprochen werden, ist es notwendig, sich damit vertraut zu machen.

Die Register A,B,C,D,E,H und L sind jeweils 8 Bit breite Speicher, die in einem Programm mit Daten geladen werden bzw. solche Daten auch ausgeben können. Das Register A (Akku) besitzt eine gewisse Sonderstellung, da es auch eine Reihe logischer Verknüpfungen durchzuführen gestattet.

Das F-Register (Flag, d. h. „Flagge" oder besser Anzeigeregister) gibt Auskunft über den Status des Prozessors und kann über verschiedene Befehle beeinflußt und auch auf seinen Inhalt hin abgefragt werden.

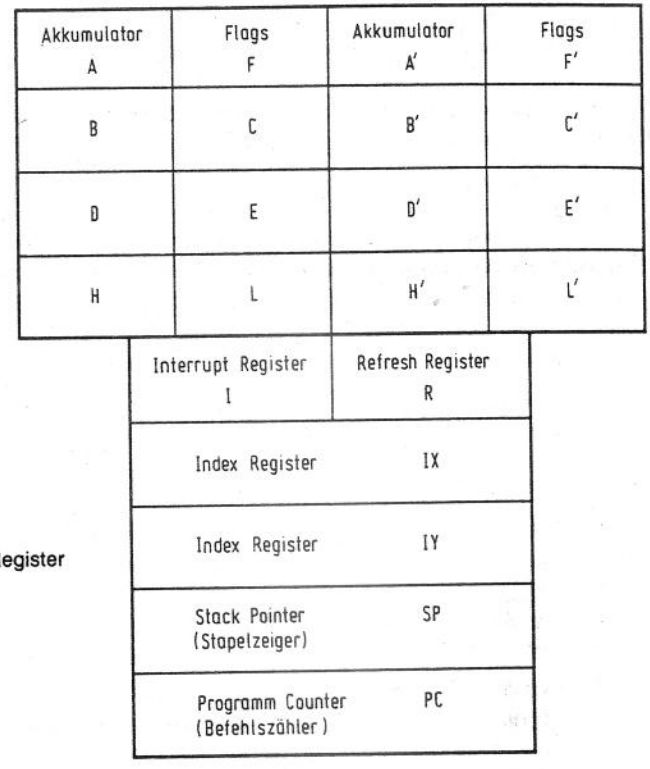


Abb. 1.4 Innerer Aufbau und Register der Z-80-CPU

Für viele Anwendungszwecke ist es von Bedeutung, daß die Registerpaare BC, DE und HL zu 16-Bit-Registern zusammengefaßt werden können und damit Zahlenwerte zwischen 0 und 65535 darstellbar sind. Auf diese Weise ist es sehr einfach möglich, einen Adressenwert zu speichern.

Der gesamte Registersatz ist in der CPU doppelt vorhanden. Durch Befehle ist ein Umsteigen auf den Zweitregistersatz möglich, wobei die bisherigen Registerinhalte nun „versteckt" im Hintergrundregistersatz erhalten bleiben.

Das Interrupt-Register I kann mit Daten geladen werden, die beim Aufruf eines Interrupts (zur Erinnerung: dabei wird die $\overline{\text{INT}}$-Leitung der CPU auf L-Pegel gezogen) wichtig sind. Der ZX 81 benützt dieses Register zum Auffinden seines Zeichensatzes bei der Bildschirmausgabe von Texten.

Das Refresh-Register R ist wichtig bei der Verwendung von dynamischen Speicher-ICs. Diese Bausteine verlieren ihre eingeschriebenen Daten, wenn nicht innerhalb bestimmter Zeitabstände eine „Auffrischung" stattfindet. Dabei merkt sich das R-Register, welche Speicherzellen als letztes aufgefrischt wurden. Beim ZX 81 wird dieses Register im Zusammenhang mit der Bildschirmausgabe „zweckentfremdet".

Die Index-Register IX und IY sind für Adressierungen relativ zu bestimmten Werten von Bedeutung.

Das PC-Register (program-counter - Befehlszähler) führt eine 16-Bit-Adresse, die auf die Speicherstelle zeigt, die den momentan ausgeführten Befehl enthält. Zu Beginn, d. h. nach einem Reset, beginnt der Befehlszähler bei Adresse 0. Mit jedem abgearbeiteten Befehl wird der Befehlszähler um eins bzw. mehr inkrementiert (erhöht), je nachdem wie lang der jeweilige Befehl war. Falls die CPU dabei auf einen Sprungbefehl oder Unterprogrammaufruf trifft, wird die Adresse, bei der das Programm fortgesetzt werden soll, in den Befehlszähler übernommen.

Das SP-Register (stack pointer - Stapelzeiger) zeigt auf die Speicheradresse, bei der der „Stapel" beginnt. Dieser Stapel beinhaltet (u.a.) Adressen, die beim Aufruf von Unterprogrammen dort abgelegt wurden. Es handelt sich dabei um die Adreßwerte, die der Befehlszähler beim Sprung in ein Unterprogramm besitzt. Auf diese Weise ist nach dem Abarbeiten des Unterprogramms ein Rücksprung in die richtige Adresse des Hauptprogramms möglich.

Anhand eines kleinen Beispielprogramms soll die Arbeitsweise verdeutlicht werden (alle Zahlenangaben sind dabei hexadezimal):

Speicheradresse	*Befehl*	*Mnemonik*
0000	31	LD SP, FFFF
0001	FF	
0002	FF	
0003	21	LD HL, FF00
0004	00	
0005	FF	
0006	01	LD BC, F000
0007	00	
0008	F0	
0009	CD	CALL 0011
000A	00	
000B	11	
000C	C3	JP 0003
000D	00	
000E	03	
000F	XX	
0010	XX	
0011	7E	LD A, (HL)
0012	02	LD (BC), A
0013	C9	RET
0014	XX	
0015	XX	
0016	XX	

Nach einem Reset besitzt der Befehlszähler den Wert 0000. Dadurch wird die Speicherstelle 0000 adressiert und der dort (z. B. in einem ROM) gespeicherte Wert 31h (=49d) von der CPU gelesen. Sie weiß darauf hin, daß ein Ladebefehl für das SP-Registerpaar vorliegt und lädt es mit dem Wert 65535d. Auf Adresse 3 findet die CPU einen Ladebefehl (21h = 33d)

für das HL-Registerpaar. Daraufhin erhält das H-Register (high: höherwertiges Byte) FFh (=255d), L (low: niederwertiges Byte) 00. Damit zeigt das HL-Register auf den Wert 65280d ($255 \times 256 + 0 \times 1$).

Nun besitzt das PC-Register den Wert 6 und liest in entsprechender Weise den Ladebefehl für das BC-Register. Dort wird dadurch der Wert 61440 eingeschrieben.

Mit dem PC-Wert 9 wird der Befehl zum Aufruf eines Unterprogramms, das auf der Adresse 17d beginnt, gefunden. Daraufhin erhält PC den Inhalt 17d, gleichzeitig wird in die Speicheradresse, auf die (SP-2) zeigt (nämlich 65533) 12d (=0Ch) geladen, die Speicherstelle 65534 (SP-1) erhält 00. Auf diese Weise wird für den Rücksprung nach dem Abarbeiten des Unterprogramms die Adresse des nächsten Befehls gespeichert. Der Stack Pointer besitzt nun den Wert 65533. Der Stapelbereich erstreckt sich also von höheren Adressen zu niedrigeren.

Das Unterprogramm lädt den Akku mit dem Inhalt der Adresse, auf die HL zeigt (65280d). Dabei kann es sich um Daten in einer Speicherstelle handeln oder auch um Informationen, die eine eventuell auf dieser Adresse angeschlossene Peripherie (z. B. ein Eingabeport) liefert. Diesen Zahlenwert schreibt der Befehl in die Speicherstelle ein, die das BC-Register adressiert (61440). Auch hier kann es sich sowohl um eine Speicherstelle (RAM) als auch um einen Peripherie-Baustein, z. B. ein Ausgabeport handeln.

Beim PC-Wert 19d (13h) wird der Befehl zum Rücksprung aus dem Unterprogramm gefunden. Daraufhin erhält PC den Inhalt 12d (0Ch) aus dem Stapel, der Stack Pointer zeigt wieder auf 65535. Es wird also wieder im normalen Programmablauf weitergearbeitet, wo ein Rücksprungbefehl zur Adresse 3 auftritt. Dadurch wird der Befehlszähler auf 3 gesetzt und der Programmdurchlauf beginnt wieder von vorne.

1.4 Der Mikrocomputer ZX 81

1.4.1 Struktur und Besonderheiten

In *Abb. 1.5* ist die Blockstruktur des ZX-81-Mikrocomputers gezeigt. Vergleicht man mit Abb. 1.1, so ist eine weitgehende Übereinstimmung zu erkennen. Im Detail weist die Schaltung allerdings einige Besonderheiten auf.

Die wichtigste Besonderheit stellt der Sinclair-Logic-Chip dar, ein kundenspezifischer Schaltkreis, der eine Reihe von Aufgaben ausführt, für die sonst ein mittleres „TTL-Grab" nötig wäre. Der Vorgänger des ZX 81, der ZX 80, war noch in dieser Technik aufgebaut und enthielt über 20 integrierte Schaltungen. Dieses Sonder-IC wickelt unter anderem die Save/Load-Prozeduren für den Kassettenrekorder, die Tastaturerkennung in Zusammenarbeit mit der CPU und die Signalausgabe an das Fernsehgerät ab.

Eine weitere spezielle Eigenheit des ZX 81 ist die Tatsache, daß sich die CPU neben der eigentlichen Programmbearbeitung um eine Menge zusätzlicher Dinge kümmert. So ist sie direkt an der Tastaturabfrage beteiligt, erstellt das Fernsehbild und steuert den Sinclair-Logic-Chip. Daraus resultiert allerdings eine gewisse Überbeschäftigung, wodurch die Zeit zum Abarbeiten eines Programms verlängert wird. Aus diesem Grund besitzt der ZX 81 einen FAST-Modus, während dem kein Fernsehbild ausgegeben wird, so daß die Rechengeschwindigkeit ansteigt.

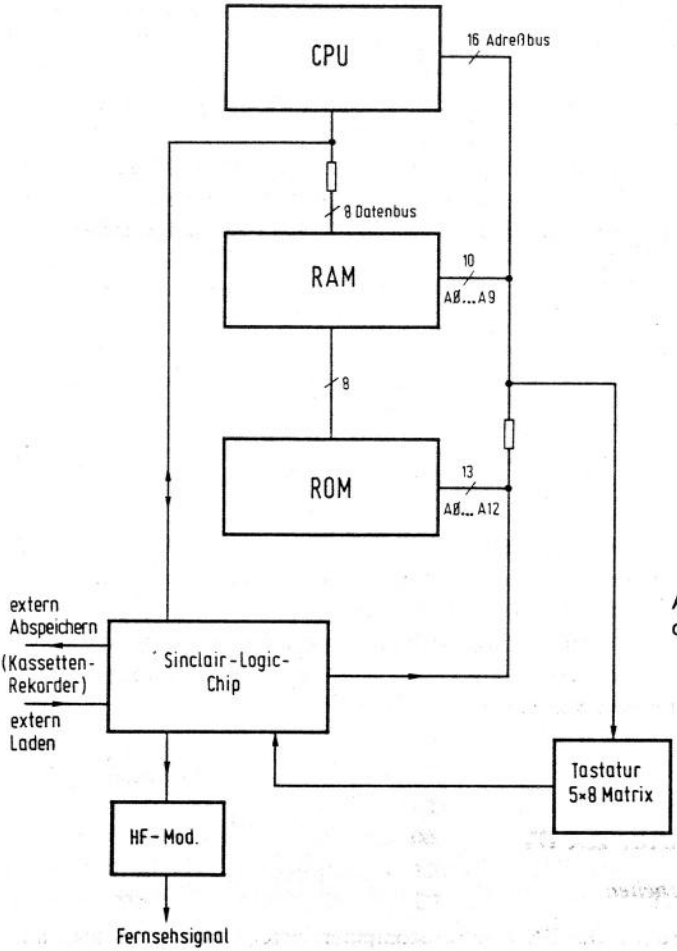


Abb. 1.5 Struktur des ZX-81-Mikrocomputers

Das 8-KByte-ROM enthält den BASIC-Interpreter und alle Steuerroutinen für Bildschirmausgabe, Tastaturabfrage und weitere Hilfsfunktionen. Das RAM umfaßt in der Grundversion 1-KByte, kann allerdings mit käuflichen Zusatzspeichern oder mit einer in diesem Buch vorgestellten Erweiterungsplatine vergrößert werden.

1.4.2 Tastaturbedienung

Der ZX 81 besitzt zur Programm- und Dateneingabe eine Folientastatur. Sie besteht aus zwei übereinandergeklebten Folien, die zueinander kreuzweise verlaufende Leiterbahnen enthält. Zwischen diesen beiden Folien befindet sich eine Isolierschicht, die an den Kreuzungspunkten Unterbrechungen aufweist. Die dort normalerweise isolierende Luftschicht wird bei Tastendruck überbrückt und schließt damit einen Schaltkontakt.

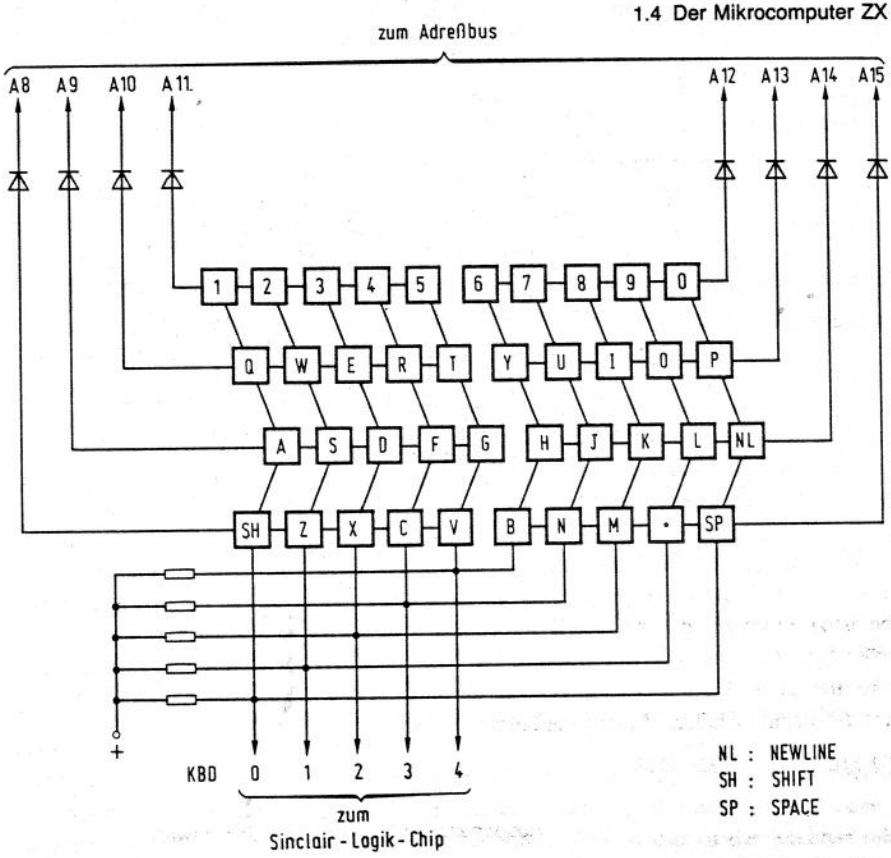


Abb. 1.6 Tastaturmatrix des ZX 81

Die Tastatur stellt also eine 8 × 5-Matrix dar *(Abb. 1.6)*. Durch die Angabe von Zeilen- und Spaltennummer ist damit jede einzelne Taste eindeutig bestimmt. Die acht Zeilenleitungen sind über Dioden mit den Adreßleitungen A8 ... A15 der CPU direkt verbunden. Die fünf Spaltenleitungen werden über Widerstände an +5 V gelegt. Sie erhalten dadurch den Signalpegel logisch 1. Diese Leitungen (KBD 0 ... 4) gelangen an die entsprechenden Eingänge des Sinclair-Logic-Chips (Pin 25, 27, 29, 31, 33). Betätigt man also beispielsweise die Taste „E", so wird die Adreßleitung A10 mit der Leitung KBD 2 verbunden. Dies bleibt ohne Auswirkung auf den Zustand von KBD 2, solange A10 H-Pegel führt. Lag im Augenblick der Tastenbetätigung an A10 jedoch L-Pegel, so wird KBD 2 ebenfalls auf L-Pegel gezogen.

Die jeweils gedrückte Taste ist also durch L-Pegel auf Adreß- und Keyboard-Leitung (KBD) bestimmt. Um die gesamte Tastatur abzufragen, werden demnach alle Adreßleitungen nacheinander auf L-Pegel gelegt. Zu jeder angesprochenen Adreßleitung müssen dann die Keyboard-Leitungen auf L-Pegel getestet werden.

Die Abfrage erfolgt durch die CPU. Dabei wird die Tatsache ausgenützt, daß etwa bei der Ausführung des Maschinenbefehls

IN A, (n)

die untere Adreßbushälfte (A0 ... A7) durch die CPU mit dem Operanden n belegt wird (der Operand n wählt die Adresse des gewünschten Eingabekanals aus), während gleichzeitig auf der oberen Adreßbushälfte (A8 ... A15) der Inhalt des Akkus erscheint.

Ein Beispiel soll das Ganze verdeutlichen:
Mit dem Maschinenbefehl

LD A, 127d

wird in den Akku das Bitmuster 0111 1111 geladen. Anschließend erfolgt der Befehl

IN A, (254d).

Dadurch erscheint auf der Adreßleitung A15 eine logische 0. Wird nun gleichzeitig eine der Tasten unten rechts gedrückt (B/N/M/./Space), so erhält auch die entsprechende Keyboard-Leitung einen L-Pegel. Das Bitmuster auf den fünf Leitungen KBD 0 ... KBD 4 stellt also eine Binärzahl dar, deren Zahlenwert der gedrückten Taste entspricht.

Der Sinclair-Logic-Chip registriert diesen Zahlenwert und führt ihn dem Datenbus zu, so daß er in den Akku geladen wird. In *Abb. 1.7* sind die über A8 ... A15 ausgegebenen Bitmuster (Zeilenauswahl) und die über den Datenbus eingelesenen Bitmuster (Spaltenauswahl) zusammengestellt. Damit kann jede einzelne Taste der Tastaturmatrix angesprochen werden. Dabei ist zu beachten, daß die ersten drei Bit der eingelesenen Bitmuster vom Sinclair-Logic-Chip bestimmt werden und unter Umständen nicht die Werte 0,1,1 haben. In Maschinensprache können diese Bits durch Undieren mit einem geeigneten Bitmuster maskiert werden. So liefert der nachgeschaltete Maschinenbefehl

AND 31d (31d = 0001 1111)

Ausgegebenes Bitmuster und Zahlenwert

Aktivierte Leitung	Bitmuster	Hex	Dez
A15	0111 1111	7F	127
A14	1011 1111	BF	191
A13	1101 1111	DF	223
A12	1110 1111	EF	239
A11	1111 0111	F7	247
A10	1111 1011	FB	251
A9	1111 1101	FD	253
A8	1111 1110	FE	254

Eingelesenes Bitmuster und Zahlenwert

Aktivierte Leitung	Bitmuster	Hex	Dez
keine	0111 1111	7F	127 (31)
KBD 0	0111 1110	7E	126 (30)
KBD 1	0111 1101	7D	125 (29)
KBD 2	0111 1011	7B	123 (27)
KBD 3	0111 0111	77	119 (23)
KBD 4	0110 1111	6F	111 (15)

Abb. 1.7 Bitmuster zur Tastaturabfrage

den reinen Tastencode. Die zugehörigen dezimalen Zahlenwerte sind in der Abb. 1.7 in Klammern gesetzt.

Möchte man in einem Maschinenprogramm z. B. testen, ob die BREAK(Space)-Taste gedrückt wurde, so fügt man an das oben begonnene Maschinenprogramm noch an:

```
AND 31d
CP30d
JP Z, nn.
```

Dabei stellt nn ein Sprungziel dar, wo die für den BREAK-Fall gewünschte Routine beginnt.

Soll die gesamte Tastaturmatrix abgefragt werden, so muß anschließend die Leitung A14 auf 0 gelegt werden, dann A13 usw. Das bedeutet, daß das im Akku enthaltene Bitmuster nach rechts rotieren muß (0111 1111, 1011 1111, 1101 1111, usw.). Hierfür kann der Maschinenbefehl RRCA (rotate right circular akku, 0Fh) verwendet werden. Damit kann diese Tastaturabfrage in einer Programmschleife integriert werden.

1.4.3 Der Bildschirm

Der ZX 81 kann 22 Zeilen mit je 32 Zeichen darstellen, dazu kommt noch eine Kommando- und Anzeigenzeile im unteren Teil des Bildschirms. Da im Grafikmodus der Raum eines Zeichens in eine obere oder untere bzw. linke und rechte Hälfte aufgeteilt wird, ergibt das 44 vertikale und 64 horizontale Druckpositionen.

Nachdem Bildaufbau und -ausgabe im wesentlichen von der CPU bewerkstelligt werden, muß hierfür Rechenzeit aufgebracht werden. Daher ist der ZX 81 zwischen den beiden Betriebsarten „SLOW" und „FAST" umschaltbar. Im SLOW-Modus ist der Rechner relativ langsam, da ein Fernsehbild ausgegeben wird. Für das Rechnen steht dadurch nur die Zeit zwischen zwei Fernsehbildern, also in der Austastlücke, zur Verfügung. Wird der FAST-Modus eingeschaltet, so erfolgt keine Bildausgabe, so daß die Rechengeschwindigkeit wesentlich ansteigt.

Das Bild, das auf dem Fernsehschirm erscheint, benötigt im RAM-Speicher $24 \times (32 + 1) + 1$ Plätze. Dies sind ein Byte für jedes Zeichen, dazu kommt noch je ein Byte für das Zeilenende (Wert 118: Symbol NEWLINE) sowie zu Beginn ein Symbol NEWLINE. Das ergibt 793 Byte Speicherbedarf. Damit wäre in der Grundversion des ZX 81 (Speicherinhalt 1 KByte) schon fast der gesamte RAM-Speicher belegt. Deshalb wird bei einem verfügbaren RAM-Speicher von weniger als $3^1/_4$ KByte nur ein Minimal-Bildschirm aufgebaut, der keine Leerzeichen enthält. Ein leerer Bildschirm besteht dann also aus 25 Speicherstellen, die alle den Wert 118 enthalten. Je mehr Bildinhalt dargestellt wird, desto größer wird der benötigte Speicherplatz, so daß man in der Grundversion sehr schnell an Speichergrenzen stößt. Daher wird in diesem Buch auch eine Speichererweiterung (zusätzliche 6 KByte RAM) beschrieben.

Die Größe des Bildschirmspeichers läßt sich durch Peeken der entsprechenden Systemvariablen berechnen: D-File (PRINT PEEK 16396 + 256 × PEEK 16397) zeigt auf den Beginn des Bildschirmspeichers, VARS (PRINT PEEK 16400 + 256 × PEEK 16401) auf den Beginn des Variablenbereichs, der sich unmittelbar an das Ende des Bildspeichers anschließt (vergleiche hierzu auch das ZX-81-Handbuch). Die Differenz dieser beiden Zahlenwerte ergibt die Größe des Bildspeichers.

Adresse	Bitmuster	Dezimal
7984	00000000	0
7985	00111100	60
7986	01000010	66
7987	01000010	66
7988	01111110	126
7989	01000010	66
7990	01000010	66
7991	00000000	0

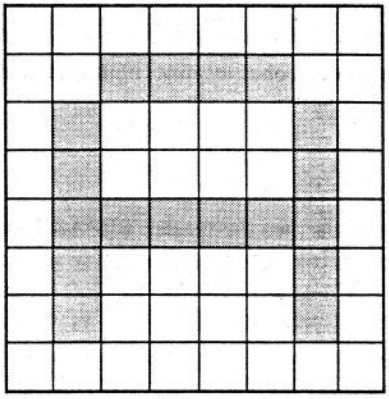


Abb. 1.8 Zeichendarstellung im ROM

So ist es auch möglich, direkt Zeichen auf dem Bildschirm auszugeben, was vor allem beim Programmieren in Maschinensprache von Bedeutung ist. Der Einfachheit halber kann man es auch in BASIC probieren:

POKE (PEEK 16396 + 256 × PEEK 16397 + 1), 38

setzt den Buchstaben „A" an die erste Printposition in Zeile 1. Entsprechend sind natürlich auch die anderen Stellen des Bildschirms erreichbar, auch die beiden Zeilen am unteren Rand des Bildschirms, die sonst für die normale Bildschirmausgabe nicht zur Verfügung stehen. Wird dabei eine Speicherstelle erwischt, die „118" (Zeilenende) enthält, gibt es Überraschungen. Aber keine Angst, kaputtgehen kann nichts, nach erneutem Einschalten funktioniert der Rechner wieder wie gewohnt.

Da an der Druckposition des Bildschirms lediglich der Zeichencode gespeichert ist, muß der Computer anhand des Codes noch das auszugebende Zeichen ermitteln. Hierzu ist im ROM ein Zeichengenerator enthalten. Der Zeichensatz ist gespeichert ab Adresse 7680d (1E00h) und umfaßt 512 Byte. Jedes einzelne Zeichen ist in einer 8 × 8-Matrix dargestellt.

Abb.1.8 zeigt dies am Beispiel des Buchstaben „A". Die Matrix-Positionen, die auf dem Bildschirm dunkel erscheinen, enthalten eine logische 1, die hellen Punkte eine logische 0. Daraus resultieren die dezimalen Zahlenwerte, die in Abb. 1.8 eingetragen sind.

Das nachfolgend aufgelistete Programm liest den Zeichengenerator ab Adresse 7680 aus und stellt die Zeichen in Großschrift auf dem Bildschirm dar.

```
  1 REM AUSLESEN DES ZEICHEN-
  2 REM GENERATORS - GROSSDARST
  3 REM
  4 REM
200 LET E=7680
300 LET Q=0
400 FOR J=E TO E+7
401 LET Z=PEEK J
410 POKE 16442,24-Q
420 LET I=PEEK 16442
430 IF 24-I<20 THEN GOTO 450
432 LET I=24
```

```
 434 LET Q=0
 436 CLS
 450 PRINT AT 24-I,0;J
 455 LET Q=Q+1
 460 FOR K=1 TO 8
 500 LET B=Z/2
 600 IF Z-2*INT B=1 THEN PRINT A
T (24-I),24-K;"*";
 601 IF Z-2*INT B=0 THEN PRINT A
T (24-I),24-K;" ";
 650 LET Z=INT B
 700 NEXT K
 800 NEXT J
 900 PAUSE 500
1000 CLS
1100 LET E=E+8
1101 POKE 16442,24
1102 LET Q=0
1200 GOTO 400
```

PROG 1

Interessant ist, wie der ZX 81 seinen Zeichengenerator findet. Dies geschieht über das Interrupt-Register I. Normalerweise enthält I den Wert 1Eh (30d). Dies sind die beiden führenden Ziffern einer 4stelligen Hex-Zahl. In dezimaler Schreibweise ist das also $1 \times 4096 + 14 \times 256 = 7680$, die Anfangsadresse des Zeichengenerators. Die beiden niederwertigen Stellen der Hex-Zahl werden durch den jeweiligen Zifferncode gebildet.

Beispiel: Der Buchstabe „A" hat den Code 38d (26h). Also enthält der Zeichengenerator den Buchstaben „A" ab der Adresse 7984 ($7680 + 38 \times 8$, da jedes Zeichen 8 Byte umfaßt).

Man könnte auf die Idee kommen, einen eigenen Zeichensatz zu definieren, indem man das Interrupt-Register beispielsweise mit 65d lädt und ab Adresse 16640 die Bitmuster dieses Zeichensatzes einschreibt. Leider funktioniert das aber nicht. Den Grund erkennt man aus *Abb.1.9*: Der Zeichengenerator wird über die Leitungen A9 ... A12 von der CPU her adres-

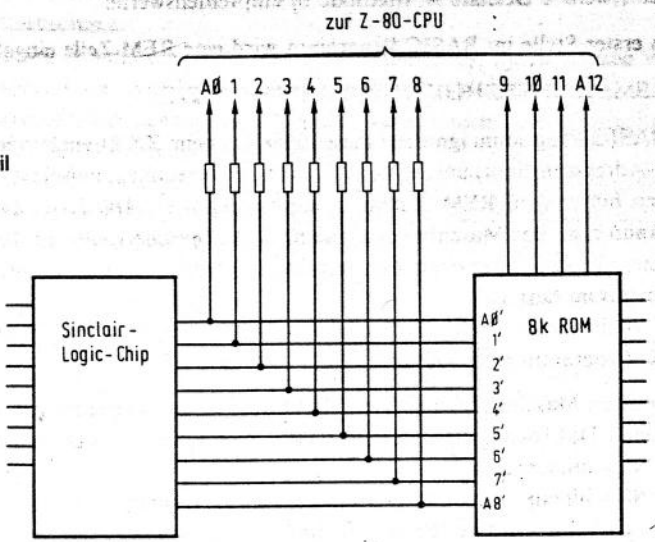


Abb. 1.9 Schaltungsdetail zum Zeichengenerator

siert. Das betreffende Zeichen wird ausgewählt, indem der Sinclair-Logic-Chip den Code-Wert des Zeichens auf A0 ... A8 ausgibt. Damit dies ungestört von den gerade ablaufenden sonstigen Aktivitäten der CPU geschehen kann, sind diese Adreßleitungen über Widerstände vom CPU-Bus entkoppelt. Es wird hier also eine spezielle Art von DMA (direct memory access - direkter Speicherzugriff) realisiert. Das bedeutet andererseits aber, daß der Zeichengenerator vom ZX 81 nur gefunden wird, wenn er sich am Steckplatz des Original-ROMs befindet.

1.4.4 Maschinenprogramme

Die in diesem Buch beschriebenen Hardware-Erweiterungen sind alle von BASIC aus bedienbar. Trotzdem ist in einigen Fällen eine sinnvolle Ausnützung aller Möglichkeiten erst durch ein Maschinenprogramm möglich.

Maschinenprogramme können beim ZX 81 mit

LET A = USR (NN)

aufgerufen werden. Dabei bedeutet NN die Speicherstelle, in der der erste Befehl des Maschinenprogramms steht. Allerdings müssen einige Vorkehrungen getroffen werden, damit das Maschinenprogramm nicht zerstört werden kann, z. B. durch das Eintippen weiterer BASIC-Programmzeilen. Hierzu gibt es im wesentlichen zwei Methoden:

a) RAMTOP wird herabgesetzt. Dazu müssen in die Speicherstellen 16388 und 16389 die entsprechenden Werte gepoket werden. Anschließend wird NEW eingegeben. Der Speicherplatz ab RAMTOP bis zur tatsächlichen Speichergrenze steht für Maschinenprogramme zur Verfügung.
Nachteilig ist, daß diese Maschinenprogramme bei SAVE nicht mit abgespeichert werden, da der Computer wegen des herabgesetzten RAMTOP-Wertes ja nichts von diesem Speicherbereich „weiß". Deshalb ist Methode b) empfehlenswerter.

b) An erster Stelle im BASIC-Programm wird eine REM-Zeile eingefügt, z. B.:

100 REM ABCDEFGHIJKLMNOPQRSTUVWXYZ.

Das BASIC-Programm ignoriert diese Zeile. Da beim ZX 81 ein Programm stets ab derselben RAM-Adresse beginnt, nämlich 16509, läßt sich berechnen, in welcher Speicherstelle das erste Zeichen hinter dem REM-Statement steht (vergleiche *Abb.1.10*). Diese Adresse ist 16514. Nun kann man das Maschinenprogramm in die Speicherstellen ab 16514 poken. Das REM-Statement muß also zuvor mindestens soviele (beliebige) Zeichen enthalten, wie das Maschinenprogramm lang ist.

NN ist in diesem Fall also 16514. Mit SAVE wird das Programm zusammen mit dem BASIC-Programm aufgezeichnet.

Bei längeren Maschinenprogrammen ist das Eintippen der entsprechenden REM-Zeile recht ermüdend. Das folgende Programm schafft hier Abhilfe. Es besteht aus einem BASIC- und einem Maschinenprogrammteil.

Das Maschinenprogramm wird zunächst in einer (kurzen) REM-Zeile ab Adresse 16514 untergebracht. Während des Programmablaufs wird interaktiv die endgültige Startadresse für

Programmzeile :
1 REM A B C D E
Speicherung :

16509	16510	16511	16512	16513	16514	16515	16516	16517	16518	16519
0	1	7	0	234	38	39	40	41	42	118
A1	A2	B1	B2							
Zeilennummer =256*A1 + A2		Länge des Textes einschl. NEWLINE =B1+256*B2		REM	A	B	C	D	E	NEWLINE

Abb. 1.10 Speicherung einer REM-Zeile

das Maschinenprogramm erfragt. Solange noch keine der später beschriebenen Speichererweiterungen vorhanden ist, empfielt es sich, dieses Programm, wie vorhin erläutert, in einen Bereich oberhalb RAMTOP zu kopieren.

Des weiteren kann die Anzahl der Zeichen im REM-Statement eingegeben werden. Diese Werte fügt das BASIC-Programm in das Maschinenprogramm ein, kopiert anschließend das Maschinenprogramm in den gewünschten Speicherbereich, ergänzt die gewünschte Anzahl von „Dummy-Zeichen" (Zeichen mit Code 8) und fügt als Zeilenende „118" (NEWLINE) an. Dann löscht es sich selbst. Nun kann mit

LET A = USR (NN)

die REM-Zeile erstellt werden. Hierbei wird die vom BASIC-Programm vorher oberhalb RAMTOP erstellte REM-Zeile als erste Programmzeile ab Adresse 16509 gespeichert. NN ist dabei die eingegebene Startadresse.

Das Maschinenprogramm benützt den Blocktransferbefehl LDIR. Hierzu müssen vorher die Register HL (Startadresse der umzukopierenden REM-Zeile), DE (Zieladresse) und BC (Anzahl der umzukopierenden Zeichen) geladen werden. Anschließend erfolgt ein Sprung nach 1027d (0403h) in die Initialisierungsroutine des ROMs.

Zu beachten ist, daß das Maschinenprogramm oberhalb RAMTOP um die Anzahl der „Dummyzeichen" (sowie eine Byte für Zeilenende) länger ist als das Programm in der ursprünglichen REM-Zeile (genügend Speicherplatz reservieren).

```
16514 ---> 33
16515 ---> 0
16516 ---> 0
16517 ---> 17
16518 ---> 125
16519 ---> 64
16520 ---> 1
16521 ---> 0
16522 ---> 0
16523 ---> 237
```

```
4082:21 00 00    LD    HL,0000
4085:11 7D 40    LD    DE,407D
4088:01 00 00    LD    BC,0000
408B:ED B0       LDIR
408D:EB          EX    DE,HL
408E:C3 03 04    JP    0403
```

```
16524 ---> 176
16525 ---> 235
16526 ---> 195
16527 ---> 3
16528 ---> 4
16529 ---> 0
16530 ---> 1
16531 ---> 0
16532 ---> 0
16533 ---> 234
```

```
 100 REM 5  )?RND■   GOSUB k FOR
 ?▀, ■   REM
 150 SLOW
 200 PRINT "STARTADRESSE FUER"
 300 PRINT "MASCHINENPROGRAMM --
-> ";
 400 INPUT S
 450 PRINT S
 500 PRINT "ANZAHL LEERZEICHEN -
-> ";
 600 INPUT A
 650 PRINT A
 660 FAST
 700 POKE 16515,S+15-256*INT ((S
+15)/256)
 800 POKE 16516,INT ((S+15)/256)
 900 POKE 16521,A+6-256*INT ((A+
6)/256)
1000 POKE 16522,INT ((A+6)/256)
1100 POKE 16531,A+2-256*INT ((A+
2)/256)
1200 POKE 16532,INT ((A+2)/256)
1300 LET X=16514-S
1400 FOR N=16514 TO 16533
1500 POKE (N-X),PEEK N
1600 NEXT N
1700 FOR N=1 TO A
1800 POKE (S+19+N),8
1900 NEXT N
2000 POKE (N+S+19),118
2100 NEW
```

PROG 2

Um Maschinenprogramme bequem eingeben zu können, empfiehlt sich ein Hilfsprogramm. Das nachfolgend aufgelistete Programm erfüllt drei Funktionen:

a) RUN 1900: zeigt Speicherinhalt zwischen S und E

b) RUN 2900: schreibt ein Maschinenprogramm in den Speicher zwischen S und E

c) RUN 4000: füllt den Speicher zwischen S und E mit dem Code des gewünschten Symbols SY.

```
 100 REM 00000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
00000000000000000000000000000000
0000000000000000000000000
 200 REM ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
00000000000000000000000000000000■
 300 REM
 400 REM
 500 REM LADEHILFE
 600 REM RUN 1900:SPEICHERINHALT
 700 REM RUN 2900:PROGRAMM POKEN
 800 REM RUN 4000:RAM FUELLEN
 900 REM
1000 REM
1600 REM ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
1700 REM ■ S - E:SPEICHERINHALT
1800 REM ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
1900 PRINT "START: ";
1910 INPUT S
1920 PRINT S;"  ENDE: ";
1930 INPUT E
1940 PRINT E
1950 PRINT "▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄
▄▄▄▄▄▄▄▄▄▄"
```

```
2000 FOR N=S TO E
2200 PRINT N;" - ";PEEK N
2300 NEXT N
2400 STOP
2450 REM ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
2460 REM █ S-E: PROGRAMM POKEN █
2470 REM ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
2900 PRINT "START: ";
2910 INPUT S
2920 PRINT S;"  ENDE: ";
2930 INPUT E
2940 PRINT E
2950 PRINT "████████████████████
████████"
3000 FOR N=S TO E
3100 PRINT N;
3200 INPUT K
3300 POKE N,K
3400 PRINT " - ";PEEK N
3410 IF PEEK 16442<=3 THEN GOSUB
 3620
3500 NEXT N
3600 STOP
3620 PRINT "  V O R S I C H T  "
3630 PAUSE 500
3650 CLS
3660 POKE 16442,24
3670 RETURN
3980 STOP
3990 REM ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
3991 REM ▌S-E:RAM MIT SY FUELLEN
3992 REM ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒
4000 PRINT "START = ";
4100 INPUT S
4200 PRINT S
4300 PRINT "ENDE = ";
4400 INPUT E
4500 PRINT E
4600 PRINT "CODE DES SYMBOLS = "
;
4700 INPUT SY
4800 PRINT SY;" --- ";CHR$ SY
4900 PAUSE 200
5000 FOR N=S TO E
5100 POKE N,SY
5200 NEXT N
5300 STOP
```

PROG 3

1.4.5 Steckleiste für Erweiterungsschaltungen

Der ZX 81 besitzt an seiner Rückseite eine Kontaktleiste, die normalerweise für die käuflichen Speichererweiterungen und den Druckeranschluß vorgesehen ist. Wie ein Blick auf die Anschlußbelegung der Kontaktleiste zeigt (*Abb. 1.11*), sind tatsächlich alle für Hardware-Erweiterungen wichtigen Signale herausgeführt. Neben den schon in Kap. 1.3 besprochenen BUS-Signalen sind dies die Betriebsspannungen +9 V und +5 V sowie die Leitungen $\overline{\text{ROM CS}}$/$\overline{\text{RAM CS}}$.

Während eines CPU-Zugriffs auf die internen Speicher gehen diese $\overline{\text{CS}}$-Signale (chip select — Bausteinauswahl) auf logisch 0. Da die beiden Leitungen durch Widerstände von den Ausgängen des Sinclair-Logic-Chips entkoppelt sind, kann durch ein von außen zugeführtes H-Signal der jeweilige Speicherbereich abgeschaltet werden, ohne daß dieses IC Schaden nimmt.

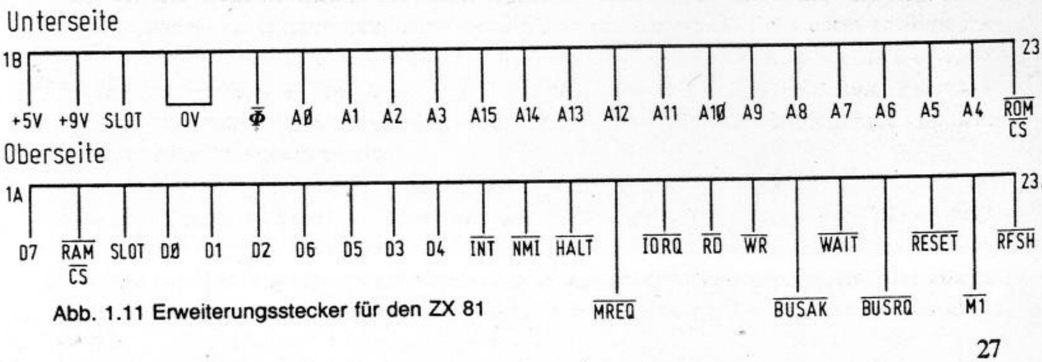


Abb. 1.11 Erweiterungsstecker für den ZX 81

Die 9-V-Leitung liefert als Ausgang die Gleichspannung des Rechnernetzteils. Falls die Belastung nicht zu groß wird, läßt sich diese Spannung auch extern verwenden. Bei umfangreicheren Erweiterungen allerdings sollte ein leistungsfähigeres Netzteil verwendet werden. Dabei kann über diese Leitung die Stromversorgung des Rechners erfolgen. Dabei wird der im Rechner eingebaute Spannungsregler mitverwendet. Der stabilisierte 5-V-Ausgang ist für externe Schaltungen nur bedingt geeignet, da bei zu hoher Belastung der Spannungsregler im Rechner zu heiß wird. Bei Verwendung der BUS-Signale ist zu beachten, daß einige Leitungen vom Betriebssystem des Rechners in einer Weise benützt werden, die ihre externe Verwendung problematisch, wenn nicht gar unmöglich werden läßt. Dies gilt für die Signale $\overline{\text{INT}}$, $\overline{\text{NMI}}$, $\overline{\text{HALT}}$, $\overline{\text{BUSRQ}}$ und $\overline{\text{BUSAK}}$.

2 Grundplatinen

2.1 Systemüberlegungen und Platinenherstellung

Damit Hardware-Erweiterungen an die jeweiligen Wünsche und Notwendigkeiten angepaßt werden können, werden zwei verschiedene Systeme vorgestellt. Beide besitzen eine Grundplatine, die für eine Bus-Entkopplung zwischen Rechner und Peripherie sorgt und einige Grundfunktionen, wie beispielsweise Adressendekodierung, bereitstellt. An diese Grundplatine können die gewünschten Funktionsplatinen, wie z. B. OUT-Ports, AD-Wandler und ähnliches angeschlossen werden.

Das „kleine" System enthält auf der Grundplatine neben den Bustreibern 6-KByte-RAM (CMOS), einen IN/OUT-Port, einen Software-Schalter sowie die Dekodierung für 13 weitere Adressen.

Beim „großen" System sind statt dessen zwei Centronics-Ausgänge und ein Centronics-Eingang vorhanden. Über eine Interface-Platine, die insgesamt 12-KByte-RAM enthält, geht es auf eine Bus-Platine, an der die gewünschten Zusatzkarten angeschlossen werden (19-Zoll-Einschubtechnik). Der RAM-Speicher auf der Interface-Karte ist allerdings nicht für BASIC-Programme gedacht, sondern dient besonderen Anwendungszwecken. 8-KByte dieses Speichers liegen als eine Art „language card" („Sprachenkarte") parallel zum ZX-81-ROM. Der Name „language card" rührt daher, daß eine Sprache (z. B. BASIC) ins RAM geladen werden kann. Da diese Programmiersprache nicht mehr unveränderlich im ROM vorliegt, kann sie nach Bedarf geändert, ergänzt oder durch eine andere (z. B. FORTH, PASCAL o. ä.) ersetzt werden. Für den ZX 81 wäre mit dem Ersatz durch eine andere Sprache allerdings ein unverhältnismäßig großer Aufwand verbunden, da Teile des Betriebssystems (Tastaturbedienung, Bildschirmerstellung usw.) ebenfalls im ROM untergebracht sind. Damit bleibt die Möglichkeit zur Veränderung des Interpreters sowie des Betriebssystems, was immer noch hochinteressant ist.

Auf diese Weise ergibt sich eine ungeahnte Zahl von Möglichkeiten. So kann der ZX 81 beispielsweise einen normalen Centronics-Drucker bedienen, es lassen sich die Keyboard-Tasten umbelegen, eigene Funktionen können definiert werden usw. Der Centronics-Eingang kann benützt werden, um Druckerdaten eines anderen Computers einzulesen. Durch entsprechende Programmierung läßt sich der ZX 81 für diesen Computer als Druckerpuffer verwenden. Als zusätzlicher BASIC-Speicher kann das 6-KByte-RAM der Grundplatine aus dem kleinen System verwendet werden.

Für I/O-Zwecke besitzt die Z-80-CPU spezielle Befehle. Dabei wird hardwaremäßig die $\overline{\text{IORQ}}$-Leitung der CPU verwendet (vergl. Kap. 1.3.1). Allerdings stehen diese Befehle nicht in BASIC zur Verfügung. Bei Anwendung dieser I/O-Technik können die jeweiligen Peripherieschaltungen also nur durch Maschinenprogramme angesprochen werden. Darüber hinaus verwendet der ZX 81 diese IN/OUT-Befehle für die Bedienung von Bildschirm, Tastatur,

Kassettenrecorder usw. Dadurch ergeben sich bei Verwendung der $\overline{\text{IORQ}}$-Leitung unter Umständen größere Probleme.

Diese Gründe waren maßgeblich dafür, daß hier zum Ansprechen der Peripherie-Schaltungen auf die speziellen I/O-Möglichkeiten der Z-80-CPU verzichtet wurde und statt dessen die I/O-Leitungen „memory mapped" betrieben werden. Das bedeutet, daß sie wie Speicherstellen angesprochen werden. Die Datenausgabe erfolgt also mit

POKE Adresse, Zahl

während eine Dateneingabe z. B. mit

A = PEEK Adresse

durchgeführt wird. Selbstverständlich können auch Maschinenprogramme unter Verwendung der entsprechenden Ladebefehle eingesetzt werden.

Einziger Nachteil dieser Methode ist die Tatsache, daß die von den I/O-Bausteinen belegten Adressen für Speicherzwecke nicht zur Verfügung stehen. Es muß also ein geeigneter Adreßbereich für die Bausteine reserviert werden.

Aufgebaut werden die Schaltungen auf Platinen im Europaformat. Bei der Vielzahl an Leitungen, die bei Computerschaltungen nötig sind, ist eine zweiseitig kupferkaschierte Platine im Regelfall nicht zu umgehen. Dabei ist vor allen Dingen auf eine exakte Passung zwischen Vorder- und Rückseite der Platine zu achten, da sonst Kurzschlüsse entstehen können. Um diese Passung zu gewährleisten, besitzen die Platinenvorlagen stets an den vier Ecken Lötaugen als Markierungen, die jeweils 5 mm vom Rand entfernt sind. Diese Markierungen werden durchstochen. Die Europakarte erhält vor der Belichtung ebenfalls vier Bohrungen (1mm Durchmesser), jeweils 5 mm von den Kanten entfernt. Auf einem Holzbrett werden vier Stifte in einem Rechteck (9 cm × 15 cm) angeordnet. Auf das Holzbrett steckt man nun die Platine und die Vorlage. Nach dem Belichten der Vorderseite wird die Rückseite in derselben Weise hergestellt. Dabei ist nur darauf zu achten, daß die Vorlage richtig herum aufgelegt wird. Bei einigermaßen sorgfältigem Vorgehen können auch mit Amateurmitteln sehr brauchbare Resultate erzielt werden.

Nach dem Ätzen ist die fertige Platine sehr sorgfältig auf eventuelle Leiterbahnunterbrechungen und Kurzschlüsse zu untersuchen. Bohrungen werden im Regelfall mit 0,8 mm Durchmesser ausgeführt. Lediglich bei Bauteilen mit dickeren Anschlußdrähten wie Steckerleisten, Siebkonsensatoren u. ä. müssen 1,0 oder 1,2 mm verwendet werden.

Für das Durchkontaktieren sind die Platinen so ausgelegt, daß im Regelfall alle von der Bestückungsseite aus sichtbaren Bohrungen durchkontaktiert werden. Hierzu wird ein dünner Draht (z. B. abgeschnittener Anschlußdraht der einzulötenden Bauelemente) durch das Loch hindurchgesteckt und auf beiden Seiten verlötet. Es wird also nicht, wie bei kommerziellen Platinen üblich, eine Durchkontaktierstelle auch zum Verlöten von Bauelementen benützt. Auf diese Weise läßt sich zwar keine ganz so große Packungsdichte erreichen, dafür ist aber das Verlöten wesentlich erleichtert.

Ein direktes Einlöten der ICs ist auf gar keinen Fall zu empfehlen, da sich eine eventuelle Fehlersuche und Beseitigung sonst äußerst schwierig gestaltet. Statt dessen sollten (gute) Stecksockel verwendet werden, die ein dichtes Anreihen der ICs gestatten.

2.2 Grundplatine I (kleines System)

2.2.1 Speichererweiterung

In der Grundversion besitzt der ZX 81 nur eine geringe Speicherkapazität von 1-KByte-RAM. Dies ist eigentlich von vornherein viel zu wenig, da bereits ein vollständig aufgebauter Bildschirm diesen Speicher schon nahezu belegt. Mit zusätzlichen 6-KByte-RAM ist man für viele Anwendungsfälle gerüstet, soweit nicht größere Programme eingetippt werden sollen (wozu die Tastatur des ZX 81 auch nicht unbedingt animiert) oder umfangreiche Datenmengen bearbeitet werden müssen. Durch Verwendung von Speicher-ICs des Typs HM 6116 (2K × 8) kann dieser Speicher auch in Einzelschritten von jeweils 2 KByte ausgebaut werden. Außerdem besteht die Möglichkeit, pinkompatible Eproms (2516 bzw. 2716, je nach Hersteller) zu verwenden, und dadurch häufig benötigte Maschinenprogramme permanent im Speicher zu haben.

Damit der Computer die Speichererweiterung für BASIC-Programme auch nützen kann, muß sich der externe RAM-Bereich nahtlos an den internen Bereich anschließen. Dazu ist es nötig, sich die Speicherorganisation des ZX 81 klarzumachen (*Abb.2.1*). Eine Eigentümlichkeit dieses Computers ist die dezimale Angabe von Adressen und Speicherinhalt. Daher sind in der Abbildung die Adressen sowohl dezimal als auch hexadezimal aufgeführt. Die vierte Spalte zeigt die dazugehörigen logischen Pegel auf den Adreßleitungen.

HEX	DEZ	Y	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	
5FFF	24575	Y7	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	Adreß-bereich-für I/O
5C00	23552		0	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	
5BFF	23551	Y6	0	1	0	1	1	0	1	1	1	1	1	1	1	1	1	1	6116 III
5800	22528		0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	
57FF	22527	Y5	0	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	
5400	21504		0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	
53FF	21503	Y4	0	1	0	1	0	0	1	1	1	1	1	1	1	1	1	1	6116 II
5000	20480		0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
4FFF	20479	Y3	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	
4C00	19456		0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	
4BFF	19455	Y2	0	1	0	0	1	0	1	1	1	1	1	1	1	1	1	1	6116
4800	18432		0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	
47FF	18431	Y1	0	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	
4400	17408		0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
43FF	17407	Y0	0	1	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1K internes RAM
4000	16384		0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
3FFF	16383		0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	ROM Doppel
2000	8192		0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
1FFF	8191		0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	ROM
0000	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Abb. 2.1 Speicherorganisation des ZX 81

Da die Z-80-CPU bei Reset den ersten (Maschinen-)Programmbefehl auf Adresse 0 erwartet, liegt das ROM für Betriebssystem und BASIC-Interpreter (8 KByte) zwischen 0 und 8191. Überraschenderweise erscheint es nochmals zwischen 8192 und 16383. Der Grund dafür ist in einer unvollständigen Adreßdekodierung zu suchen, d. h. für die Zuordnung der Speicheradressen im ROM zu den von der CPU ausgegebenen Adreßsignalen werden nur die Leitungen A0 ... A12 verwendet. Dadurch „sieht" die CPU 0 und 8192 als dieselbe Adresse.

Man kann sich diesen Effekt an den dezimalen Zahlen 1293 und 7293 klarmachen. Betrachtet man nur die letzten drei Ziffern, so erhält man beide Male dieselbe Zahl.

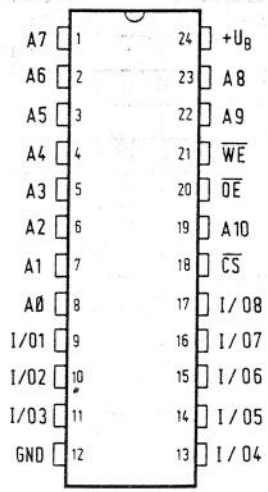
Von 16384 bis 17407 schließt sich das interne RAM an. Die RAM-Obergrenze wird beim Einschalten vom Computer mit einem Testprogramm ermittelt und als Systemvariable RAMTOP gespeichert. Dabei enthält die Speicherstelle 16388 die unteren 8 Bit, die Speicherstelle 16389 die oberen 8 Bit der entsprechenden 16-Bit-Zahl. Durch die Befehlseingabe

PRINT PEEK 16388 + 256 × PEEK 16389

erhält man diesen Zahlenwert in dezimaler Schreibweise. Er stellt die Adresse des ersten nicht existierenden Bytes dar, ohne Speichererweiterung ergibt sich also die Zahl 17408.

Käufliche Speichererweiterungen benützen in der Regel das interne RAM nicht und schalten es über die Leitung $\overline{\text{RAM-CS}}$ (H-Pegel) ab. Dadurch erhält man eine bequemere Organisation des Zusatzspeichers (z. B. 8 × 2 KByte Speicher-ICs ergeben 16 KByte) und kann die Adreßdekodierung einfacher vornehmen. Für den Selbstbau besitzt diese Methode aber ein ernstzunehmendes Problem: funktioniert die Speichererweiterung nach dem Zusammenbau aus irgendeinem Grund nicht (und nach dem Gesetz von Murphy ist sogar sicher damit zu rechnen), stürzt der Computer unweigerlich ab, da er zu seinem Betrieb auf jeden Fall RAM-Bereich benötigt. Dann aber ist eine Fehlersuche praktisch unmöglich, da ja keinerlei Testprogramme zum Einkreisen der Fehlerstelle eingegeben werden können. Das heißt, entweder findet man den Fehler durch Zufall, oder eben nie.

In *Abb. 2.2* ist das Anschlußschema und die Funktionstabelle des Speicherbausteins HM 6116 gezeigt. Wie man erkennt, besitzt dieser Speicher 11 Adreßleitungen (A0 ... A10).



$\overline{CS}$	$\overline{OE}$	$\overline{WE}$	Betriebsart
1	X	X	nicht angesprochen
0	0	1	Lesen
0	1	0	Schreiben
0	0	0	Schreiben

Abb. 2.2 Anschlußbelegung und Funktionstabelle des HM 6116

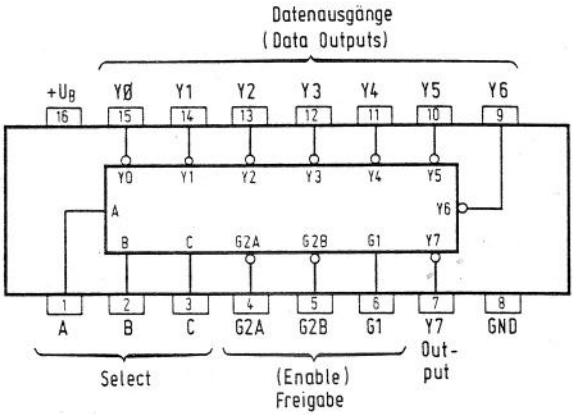


Abb. 2.3 Anschlußbelegung und Funktionstabelle des 74LS138

Eingänge/Inputs					Ausgänge							
Freigabe		Select										
G1	G2*	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	1	X	X	X	1	1	1	1	1	1	1	1
0	X	X	X	X	1	1	1	1	1	1	1	1
1	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	1	1	0	1	1	1	1	1	1
1	0	0	1	0	1	1	0	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1	1	1
1	0	1	0	0	1	1	1	1	0	1	1	1
1	0	1	0	1	1	1	1	1	1	0	1	1
1	0	1	1	0	1	1	1	1	1	1	0	1
1	0	1	1	1	1	1	1	1	1	1	1	0

Würde man ihn direkt mit den entsprechenden Leistungen des Adreßbusses verbinden, so würde er gemeinsam mit dem internen ROM zwischen Adresse 0 und 2047 angesprochen. Erwünscht ist er allerdings zwischen 17408 und 19455. Erreichen läßt sich das, indem man die Adreßleitungen A11 ... A14 zur Unterscheidung heranzieht. Im Bereich von 16384 bis 24575 ist stets A14 = 1 und A13 = 0. Betrachtet man A12, A11 und A10 als dreistellige Binärzahl, so entspricht jeder solchen Zahl ein 1 KByte großer RAM-Bereich.

Ein 3-zu-8-Dekoder 74LS138 kann in Abhängigkeit von dieser Zahl (CBA in *Abb.2.3*) jeweils einen der Ausgänge Y0 ... Y7 auf logisch 0 schalten (Spalte 3 in Abb.2.3). Führt man seinem Enable-Eingang G1 (H-aktiv) das Signal A14, den Eingängen G2A und G2B (L-aktiv) die Signale MREQ und A13 zu, so ist gewährleistet, daß der Dekoder nur während eines Speicherzugriffs auf die Adressen 16384 bis 24575 angesprochen wird.

A15 wird nicht benützt. Das hat zur Folge, daß sich die gesamte Speicheraufteilung zwischen 32768 und 65535 wiederholt (unvollständige Adressendekodierung).

Da der 3-zu-8-Dekoder jeweils 1-KByte-Blöcke auswählt, die Speicher-ICs aber 2 KByte umfassen, müssen jeweils zwei Ausgangsleitungen über eine ODER-Verknüpfung zusammengefaßt werden, um den betreffenden Speicher anzusprechen. Das bedeutet, Y1 oder Y2 aktiviert den $\overline{CS}$-Eingang (vergl. Abb. 2.3) des ersten Speicher-ICs. Y3 oder Y4 bzw. Y5 oder Y6 bewirken dasselbe für die anderen Speicher.

Wenn Y0 = 0 ist, ist das interne RAM an der Reihe. Das bedeutet umgekehrt, daß während Y0 = 1 externe Bausteine angesprochen werden. Daher ist dieses Signal zur Abschaltung des internen RAMs geeignet. Diese Abschaltung ist nötig, da infolge unvollständiger Adreßdekodierung dieses RAM mehrfach erscheint und dadurch mit externen Speichern kollidieren würde.

2.2.2 I/O-Ports

Mit Y7 = 0 wird der Bereich von 23552 bis 24575 erfaßt. Dieser Bereich wird nun nicht für einen weiteren 2-KByte-Speicher genützt (obwohl das natürlich genauso möglich wäre), sondern für Ein-/Ausgabeleitungen (I/O). Streng genommen besitzt der ZX 81 bereits solche I/O-Ports: Die Tastatur zur Daten- und Befehlseingabe (I-Port) und den Bildschirm zur Informationsausgabe (O-Port). Darüber hinaus steht mit dem Kassettenrekorder-Anschluß ein bidirektionaler I/O-Port für das Laden und Speichern von Programmen zur Verfügung. Hierbei handelt es sich um einen seriellen Port mit einer Datenbreite von 1 Bit. Wer sich einen Drukker zulegt, besitzt eine weitere Ausgabemöglichkeit.

Nachteil all dieser Schnittstellen ist aber, daß kein direkter Kontakt zur Umwelt besteht, stets muß der Mensch als Vermittler zusätzlich in Aktion treten. Wünschenswert ist also eine direkte Verbindung, um etwa ein Gerät direkt durch den Computer ein- oder ausschalten zu können. Umgekehrt kann auf diesem Weg eine unmittelbare Datenerfassung, z. B. der Zimmertemperatur, Raumhelligkeit o. ä. erfolgen.

Für diesen Zweck stehen eine Reihe spezieller ICs zur Verfügung, z. B. die Z-80-PIO (*P*arallel *I*n/*O*ut). Vor allem für den Anfänger von Nachteil ist die Notwendigkeit einer Programmierung dieses Bausteins für seine verschiedenen Betriebsarten, was zu einer gewissen Unübersichtlichkeit führt. Zudem kann es durch falsche Programmierung u. U. zu Hardware-Schäden kommen. Dies geschieht beispielsweise dann, wenn eine fälschlicherweise als Ausgang programmierte Leitung gegen eine andere Ausgangsleitung treibt. Führt in einem solchen Fall der eine Ausgang H-Pegel, der andere L-Pegel, fließt ein hoher Strom, der zur Zerstörung der Ausgangstransistoren führen kann. Daneben liefern die Ausgänge der PIO relativ geringe Ströme, so daß z. B. Leuchtdioden zur Pegelanzeige nicht direkt angeschlossen werden können, sondern einen zusätzlichen Leistungstreiber benötigen. Daher wurde im folgenden ein anderes Konzept verwendet.

Speicherstellen können in BASIC mit dem Befehl

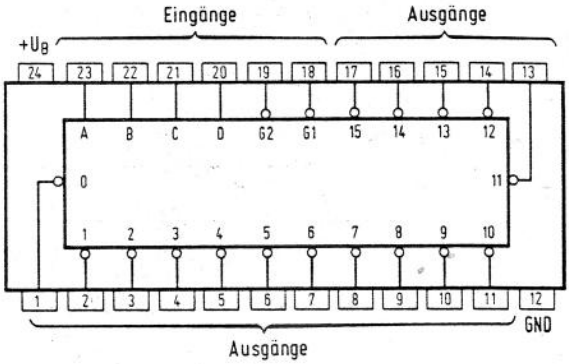
PRINT PEEK Adresse

gelesen werden, mit

POKE Adresse, Zahl

geladen werden. Ersetzt man das Speicher-IC durch ein geeignetes Port-IC, kann auf gleiche Weise eine Ein- oder Ausgabe realisiert werden. Daher schließt sich der I/O-Bereich an den Adressenbereich der Speichererweiterung an.

Zu diesem Zweck steht vom Adreßdekoder 74LS138 (IC4 der Speichererweiterung) das Signal Y7 zur Verfügung, das im Adreßbereich 23552 ... 24575 logisch 0 ist. Mit diesem Signal läßt sich ein 4-zu16-Dekoder 74LS154 über die Strobe-Eingänge G1/G2 aktivieren (*Abb.2.4*). Werden an seine Eingänge A, B, C und D die Adreßleitungen A0, A1, A2 und A3 angelegt, so

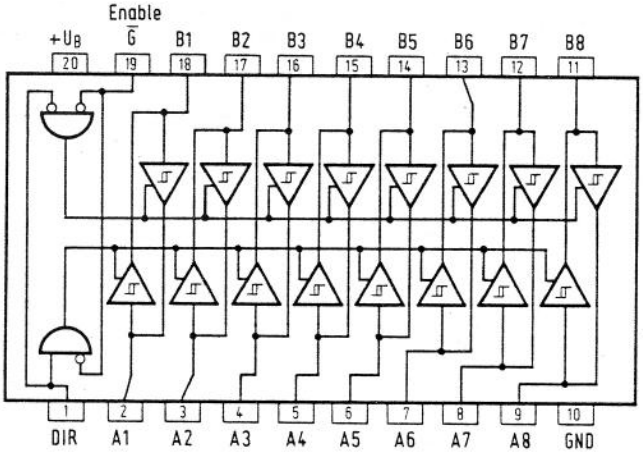


Eingänge						Ausgänge														
G1	G2	D	C	B	A	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	1	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1	0	0	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1
0	0	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
0	0	1	0	0	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
0	0	1	0	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
0	0	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	X	X	X	X	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	X	X	X	X	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	X	X	X	X	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Abb. 2.4 4-zu-16 Dekoder 74LS154: Anschlußbild und Funktionstabelle

wählt er in Abhängigkeit von der entsprechenden Adressenkombination eine der 16 Ausgangsleitungen an (L-aktiv). Auf diese Weise können die Adressen 23552 ... 23567 einzeln angesprochen werden. Wegen unvollständiger Adreßdekodierung wiederholen sich diese Adressen 64mal bis zu Adresse 24575, was aber nicht weiter stört.

Im Falle eines Eingabe-Ports müssen beim Ansprechen der gewünschten Adresse die von außen kommenden Daten auf den Datenbus gelegt werden. Hierfür läßt sich sehr gut der Bustreiber 74LS245 „zweckentfremden" (*Abb.2.5*). Legt man seinen Anschluß 1 an Masse, so ist eine Datenübertragung von B (Außenwelt) nach A (Datenbus) möglich. Dies geschieht allerdings nur, wenn an Pin 19 (Enable) ein L-Signal anliegt. Ist das Signal an Pin 19 logisch 1, so sind die IC-Ausgänge im hochohmigen Zustand (Tristate) und belasten den Datenbus

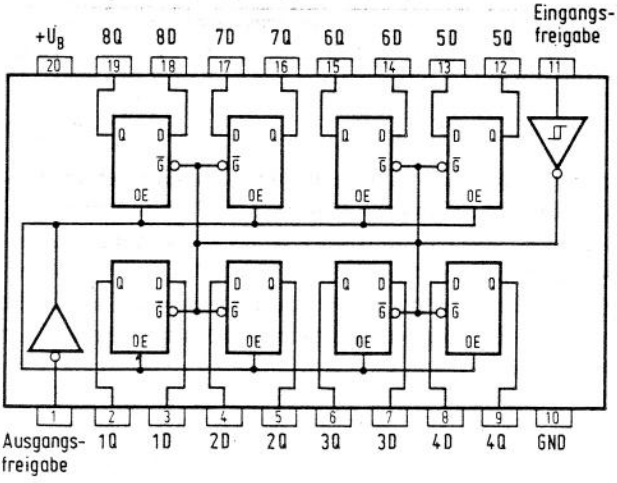


Pin 19	Pin 1	
Freigabe Enable $\overline{G}$	Richtungs-steuerung DIR	Datenfluß
0	0	von B nach A
0	1	von A nach B
1	X	hochohmig

Abb. 2.5 Bustreiber 74LS245: Anschlußbild und Funktionstabelle

nicht. Die Eingabedaten müssen vorhanden sein, wenn die Port-Adresse angesprochen wird, eine Zwischenspeicherung erfolgt also nicht. Dies ist in den meisten Anwendungsfällen allerdings kein Nachteil.

Da die Ausgabedaten von der CPU nur sehr kurze Zeit auf den Datenbus gebracht werden (einige μs lang), ist für die Ausgabe-Ports eine Zwischenspeicherung erforderlich.



Pin 1	Pin 2		
Ausgangs-freigabe	Eingangs-freigabe $\overline{G}$	D	Ausgang
0	1	1	1
0	1	0	0
0	0	X	Q_0
1	X	X	Z (hochohmig)

Abb. 2.6 8-fach-Latch 74LS373: Anschlußbild und Funktionstabelle

Diese Speicherung kann ein 8fach-Latch 74LS373 übernehmen *(Abb. 2.6)*. Seine Ausgänge werden durch ein L-Signal an Pin 1 (output control) aktiviert, ein H-Signal schaltet die Ausgänge in den hochohmigen Zustand. Ein H-Signal an Pin 11 (enable) veranlaßt das IC, die an den Eingängen anliegenden Daten zu übernehmen und zu speichern. Dieses Signal muß also vom Computer geliefert werden, sobald die Ausgabe-Daten auf dem Datenbus vorhanden sind.

2.2.3 Gesamtschaltung der Grundplatine I

Abb.2.7 zeigt die Gesamtschaltung der Grundplatine I. Die Stromversorgung erfolgt über den 9-V-Ausgang des ZX 81 und einen Spannungsregler 7805. Zum Abblocken von Impulsstörun-

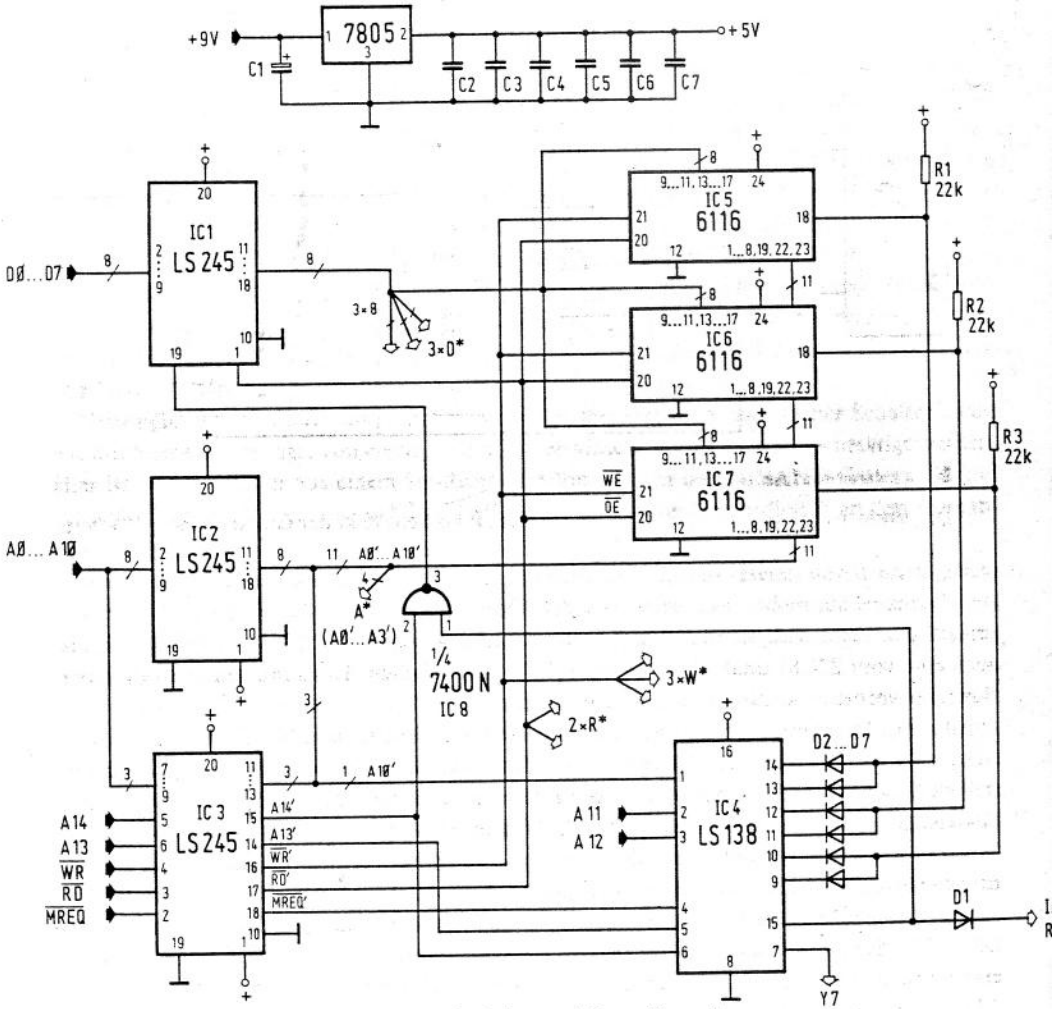


Abb. 2.7a Schaltbild der Grundplatine I (Speicher und Buspufferung)

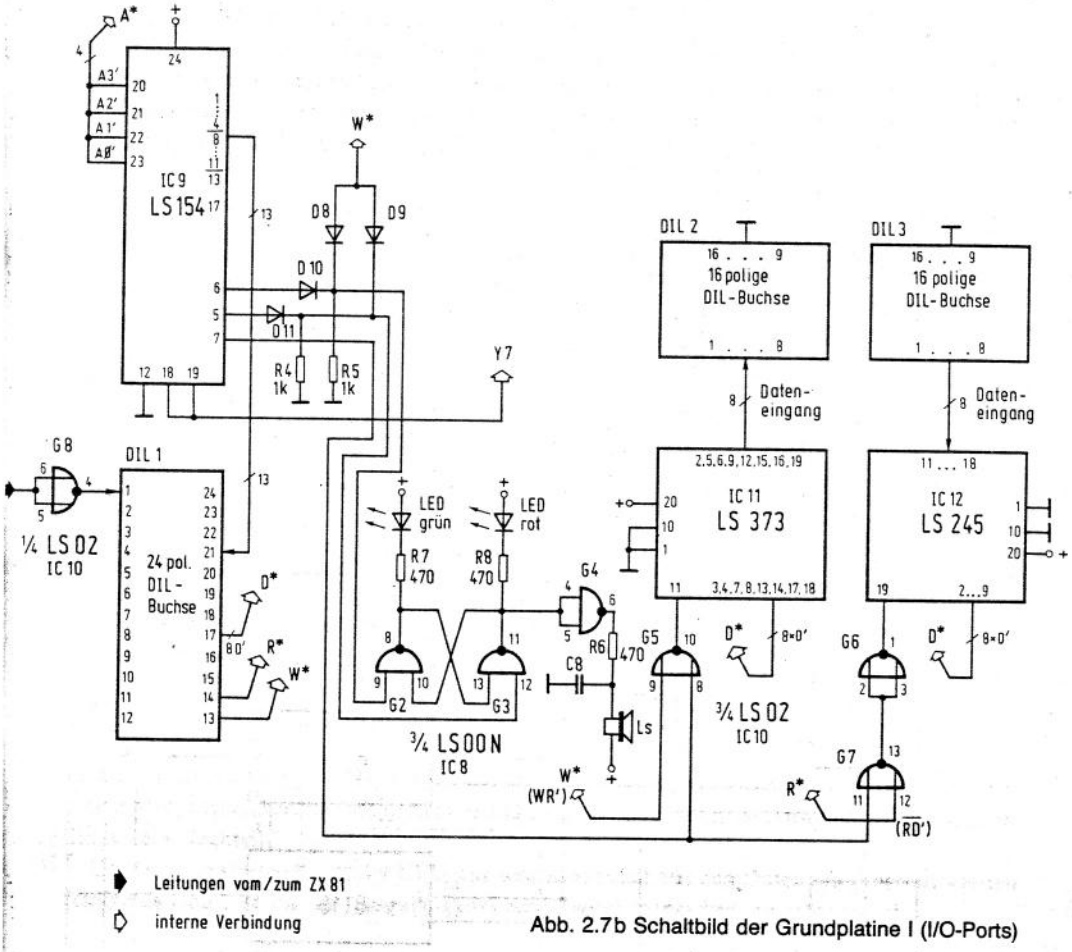


Abb. 2.7b Schaltbild der Grundplatine I (I/O-Ports)

gen auf den Versorgungsleitungen dienen die Kondensatoren C1 ... C7. Selbstverständlich ist auch eine vom ZX 81 unabhängige Stromversorgung möglich. Dazu müssen dann die überflüssig gewordenen Leitungen aufgetrennt werden.

Alle vom Computer kommenden Signale werden mit Bustreibern 74LS245 (IC1 ... 3) gepuffert (Ausnahme: die Adreßleitungen A11 und A12, die beide nur eine LS-TTL-Last zu treiben haben). A0 ...A10 gehen nach der Pufferung direkt an die Anschlüsse der Speicher-ICs (IC 5 ... 7). Die Leitungen A14', A13' und MREQ' aktivieren den Adreßdekoder 74LS138 wie oben erläutert. In Abhängigkeit von dem aus A10', A11 und A12 gebildeten Eingangswort wird eine der acht Ausgangsleistungen auf logisch 0 geschaltet. Die ODER-Verknüpfung erfolgt für jeweils zwei dieser Ausgangsleitungen über die Germanium-Dioden D2 ... D7. Silizium-Dioden sind wegen ihrer Flußspannung von 0,7 V ungeeignet, da dann eine logische 0 bis zu 1,1 V betragen kann (maximale Ausgangsspannung für logisch 0 bei TTL-Gattern 0,4 V + 0,7 V), das Speicher-IC am CS-Eingang hierfür aber maximal 0,8 V

akzeptiert. Das $\overline{RD'}$-Signal wird dem Anschluß $\overline{OE}$ (output enable), das $\overline{WR'}$-Signal dem Anschluß $\overline{WE}$ (write enable) von IC 5 ... 7 zugeführt.

Um zu verhindern, daß das interne RAM gleichzeitig mit dem externen Speicher auf den Datenbus zugreift, wird es während Y0 = 1 über die Diode D1 abgeschaltet. Solange Y0 = 0 oder A14 = 0 ist (vergl. Abb.2.1) wird der Datenbus vom internen RAM bzw. ROM belegt. Dann wird der Datenbustreiber IC1 abgeschaltet (der Tristate-Ausgang wird hochohmig), um einen Doppelzugriff zu vermeiden. Die hierfür nötige ODER-Verknüpfung erfolgt über das NAND-Gatter G1, das in negativer Logik (Y0 und A14 sind hier L-aktiv) eine ODER-Funktion realisiert.

Da der Datenbustreiber in Abhängigkeit von einem Schreib- bzw. Lesezugriff die Daten aus dem Computer hinaus- oder in den Computer hineinleiten soll, muß seine Signalrichtung über die $\overline{RD'}$-Leitung umgeschaltet werden (Pin 1 von IC1). $\overline{RD'} = 0$ bedeutet Lese-Operation („hinein"), $\overline{RD'} = 1$ bedeutet Nichtlese-Operation („hinaus").

Die Ports werden vom Adreßdekoder IC9 angesprochen. Dieses IC wird im Adreßbereich 23552 ... 24575 über die Leitung Y7 von IC4 aktiviert. Dabei wählen A0 ... A3 wie erläutert eine der 16 Ausgangsleitungen an.

Wenn die Adresse 23558 angesprochen wird, führt Pin 7 von IC9 ein L-Signal. Falls gleichzeitig $\overline{WR'}$ logisch 0 ist (d. h. wenn z. B. ein POKE-Befehl gegeben wurde), wird über das NOR-Gatter G5 (da negative Logik: UND-Verknüpfung) das 8fach-Latch IC 11 angesprochen (H-Pegel an Pin 11) und speichert die auf dem Datenbus liegenden Informationen. Sie stehen dann an der Buchse DIL 2 extern zur Verfügung.

Falls $\overline{RD'}$ und Pin 7 von IC9 ein L-Signal führen (also beispielsweise bei einem PEEK-Aufruf) wird auf analoge Weise über G7 IC12 adressiert. G6 dient lediglich als Inverter, da IC12 an Pin 19 in diesem Fall L-Pegel benötigt. Dadurch werden die an DIL3 von außen kommenden Daten in den Computer übernommen.

Neben den Ports ist auch noch ein Software-Schalter enthalten. Ein solcher Schalter kann, wie der Name schon sagt, von einem BASIC- oder Maschinenprogramm aus betätigt werden. Hier ist dieser Schalter aus einem Set-Reset-Flipflop mit den beiden NAND-Gattern G2 und G3 aufgebaut. Der Schaltzustand des Flipflops ist durch zwei Leuchtdioden an den Ausgängen erkennbar.

Durch einen Schreibbefehl auf Adresse 23556 wird das Flipflop gesetzt, durch einen gleichartigen Befehl auf Adresse 23557 rückgesetzt. Realisiert wird dies, indem die entsprechende Ausgangsleitung von IC9 über die Dioden D8 ... D11 mit dem $\overline{WR'}$-Signal UND-verknüpft wird (negative Logik, da L-aktiv).

Zusätzlich treibt einer der beiden Schalterausgänge über G4 einen Lautsprecher. Jedesmal, wenn umgeschaltet wird, ist im Lautsprecher ein Klick-Geräusch zu hören. Erfolgt die Umschaltung schnell genug, so entsteht ein Ton. Seine Frequenz ist durch die Umschaltfrequenz gegeben, auf diese Weise ist also eine softwaregesteuerte Tonerzeugung möglich. Der Wert von R6 bestimmt die Lautstärke des Tons. Der Lautsprecher ist nicht gegen Masse, sondern gegen +U geschaltet, da TTL-Gatter gegen Masse einen höheren Strom vertragen können.

Die verbleibenden 13 dekodierten Adressen sind zusammen mit dem Datenbus und den $\overline{RD'}$- sowie $\overline{WR'}$-Signalen an die Buchse DIL1 geführt. Auch der CPU-Takt steht an dieser Buchse zur Verfügung. Hier lassen sich die weiteren Zusatzkarten anschließen.

Die Portadressen sowie die Stiftbelegung der DIL-Buchsen sind ind *Abb.2.8* zusammengefaßt.

23556 Einschalten LED grün
23557 Einschalten LED rot
23558 I/O-Port (jeweils unidirektional)

DIL 1				DIL 2	(Ausgabe)	DIL 3	(Eingabe)
1	Takt (∅)	13	23562	1	D 0	1	D 0
2	$\overline{RD}$'	14	23561	2	D 1	2	D 1
3	$\overline{WR}$'	15	23560	3	D 2	3	D 2
4	23552	16	23559	4	D 3	4	D 3
5	23553	17	D 0'	5	D 4	5	D 4
6	23554	18	D 1'	6	D 5	6	D 5
7	23555	19	D 2'	7	D 6	7	D 6
8	23567	20	D 3'	8	D 7	8	D 7
9	23566	21	D 4'				
10	23565	22	D 5'	9...16	Masse	9...16	Masse
11	23564	23	D 6'				
12	23563	24	D 7'				

Abb. 2.8 Portadressen und Steckerbelegung

2.2.4 Aufbauhinweise, Inbetriebnahme und Test

Das Platinenlayout ist in *Abb.2.9* gezeigt. Nach Überprüfung der Platine müssen zuerst alle Durchkontaktierungen erfolgen. Ausnahmsweise dürfen bei dieser Platine nicht alle von der Bestückungsseite sichtbaren Bohrungen kontaktiert werden, die Bohrungen der Steckerleiste müssen frei bleiben. Es sollte möglichst nichts vergessen werden, da einige Kontaktierungen unter IC-Sockeln liegen und später nicht mehr zugänglich sind. Anschließend werden die IC-Sockel, dann Kondensatoren und sonstige Bauteile eingesetzt und verlötet.

Beim Einlöten der Steckerleiste ist genügend Abstand zur Platine einzuhalten, damit das Anstecken an den ZX 81 ohne Schwierigkeiten möglich ist. Die A-Kontaktreihe wird von der Bestückungsseite her verlötet. Für den Spannungsregler sollte ein geeignetes Kühlblech vorgesehen werden (Befestigungsbohrungen sind auf der Platine vorhanden).

Abb.2.10 zeigt den Bestückungsplan (Blick auf Bauteileseite) sowie die Stückliste zur Grundplatine I.

Bei der Inbetriebnahme empfiehlt es sich, die Platine zunächst ohne eingesteckte ICs in Betrieb zu nehmen. Wenn keine Kurzschlüsse vorhanden sind, muß nun auf dem Bildschirm der Cursor erscheinen. Dann werden die ICs eingesetzt (Betriebsspannung ausschalten!). Es dauert jetzt etwas länger, bis der Cursor erscheint, da infolge der Speichererweiterung nun ein größerer Speicherbereich geprüft wird.

RAMTOP (PRINT PEEK 16388 + 256 × PEEK 16389) liefert bei richtiger Funktion 23552. Falls gewünscht, lassen sich auch nur ein oder zwei Speicher-ICs einsetzen. Dabei ist auf die richtige Reihenfolge zu achten, da der Computer einen durchgehenden Speicherbereich benötigt. RAMTOP ist in diesem Fall 19456 bzw. 21504.

POKE 23556, Zahl

schaltet LED grün,

POKE 23557, Zahl

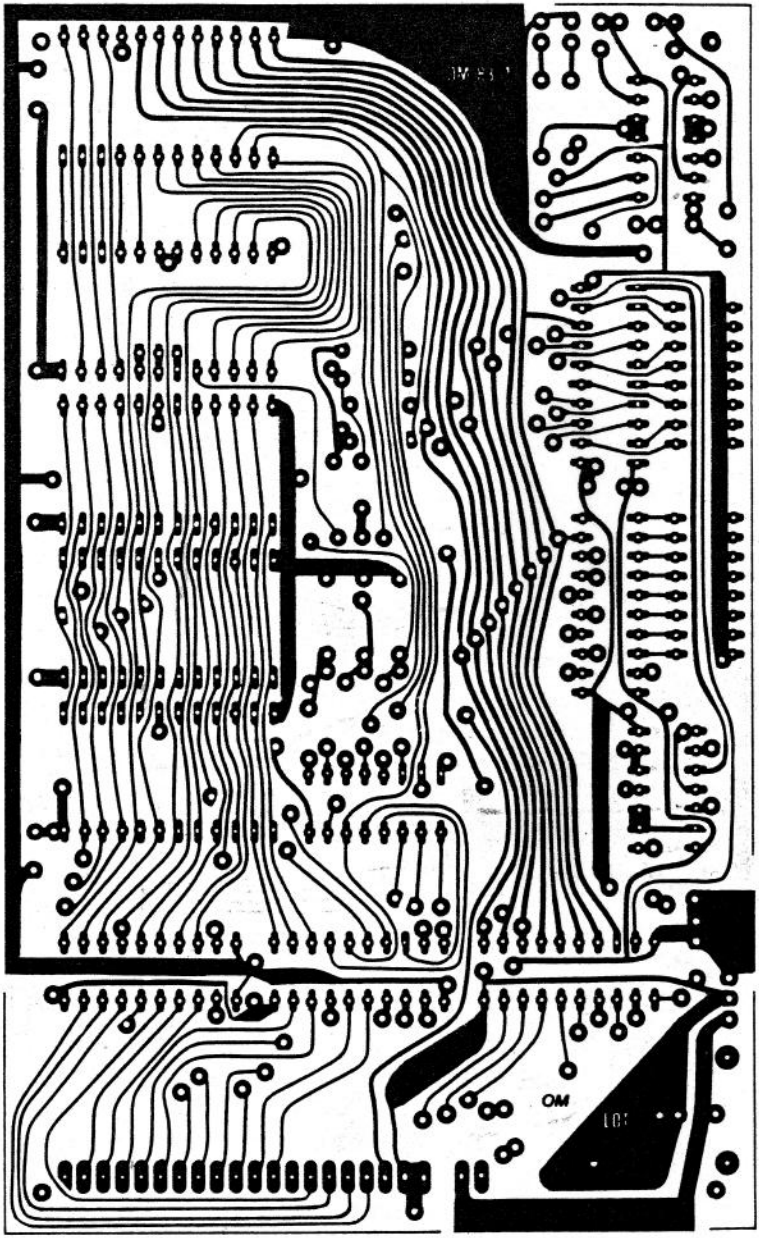


Abb. 2.9a Platinenlayout, Lötseite

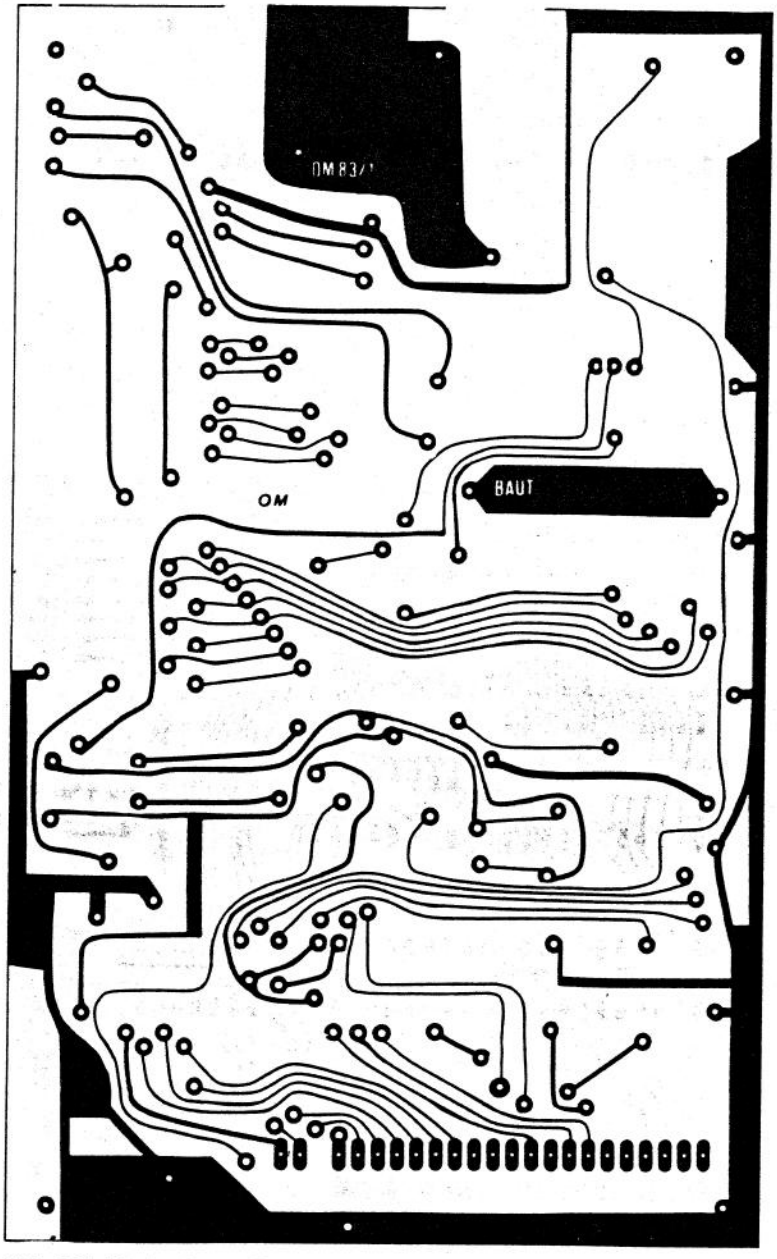


Abb. 2.9b Platinenlayout, Bestückungsseite

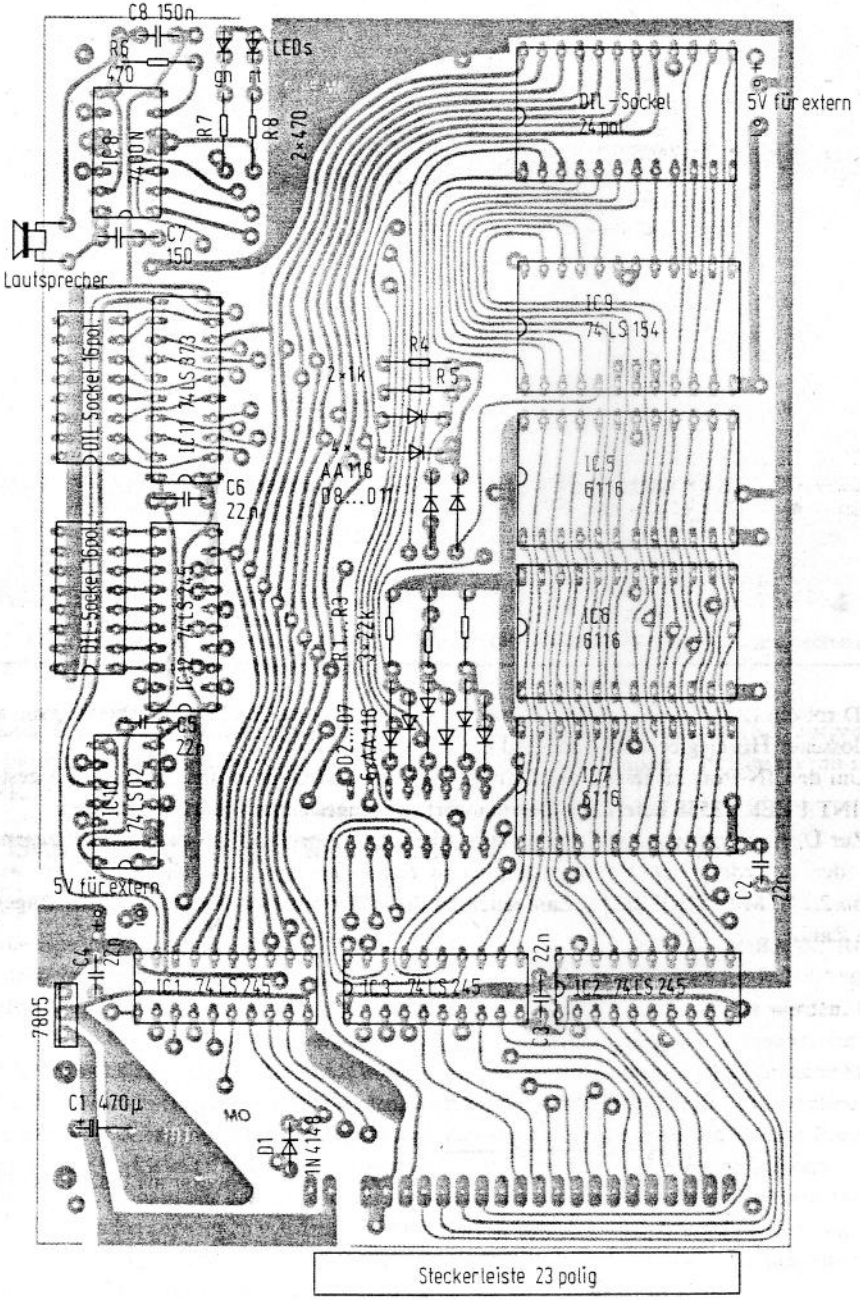


Abb. 2.10 Bestückungsplan (Blick auf die Bauteileseite)

Stückliste der Grundplatine I

IC1, 2, 3, 12	4 × 74LS245
IC5, 6, 7	3 × HM 6116LP-3
IC10	1 × 74LS02
IC9	1 × 74LS154
IC4	1 × 74LS138
IC11	1 × 74LS373
IC8	1 × 7400N (keine LS-Version, da höherer Ausgangsstrom für den Lautsprecher)
	1 × 7805 (Spannungsregler 5V)
	1 × LED rot 3 mm
	8 × Anreih-LED rot (für OUT-Kontrolle)
D1	1 × 4148 (o. ä. Si-Diode)
D2 ... D11	10 × AA 116 (o. ä. Ge-Diode)
R1 ... R3	3 × 22 kΩ
R4 ... R5	2 × 1 kΩ
R6 ... R8	3 × 470 Ω
C1	1 × 470 µF
C2 ... C6	5 × 22 nF
C7 ... C8	2 × 150 nF
Sp	1 × Lautsprecher 50 Ω (geeignet auch Postkapsel 32 Ω)
	1 × Steckerleiste 2 × 23polig, Raster 2,54 mm
	1 × DIL-Schalter 8polig
	1 × DIL-Stecker 16polig
	5 × Stecksockel 24polig
	5 × Stecksockel 20polig
	3 × Stecksockel 16polig
	2 × Stecksockel 14polig
	Kleinmaterial (Steckstifte)

LED rot ein (Zahl kann zwischen 0 und 255 liegen). Ein an den Lautsprecherausgang angeschlossener Hörer gibt dabei jedesmal ein Klickgeräusch ab.

Um den IN-Port zu testen, wird ein 8poliger DIL-Schalter in den Sockel DIL3 gesteckt. PRINT PEEK 23558 liefert den Dezimalwert der eingestellten Binärkombination.

Zur Überprüfung des OUT-Ports (DIL2) sind acht Anreih-LEDS geeignet, die zusammen mit den erforderlichen Vorwiderständen in einen 16poligen DIL-Stecker gelötet werden (*Abb.2.11*). Mit POKE 23558, Zahl leuchten die LEDs gemäß dem Binärwert der eingegebenen Zahl.

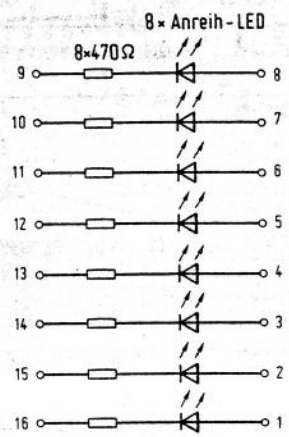


Abb. 2.11 DIL-Stecker mit LEDs für OUT-Port

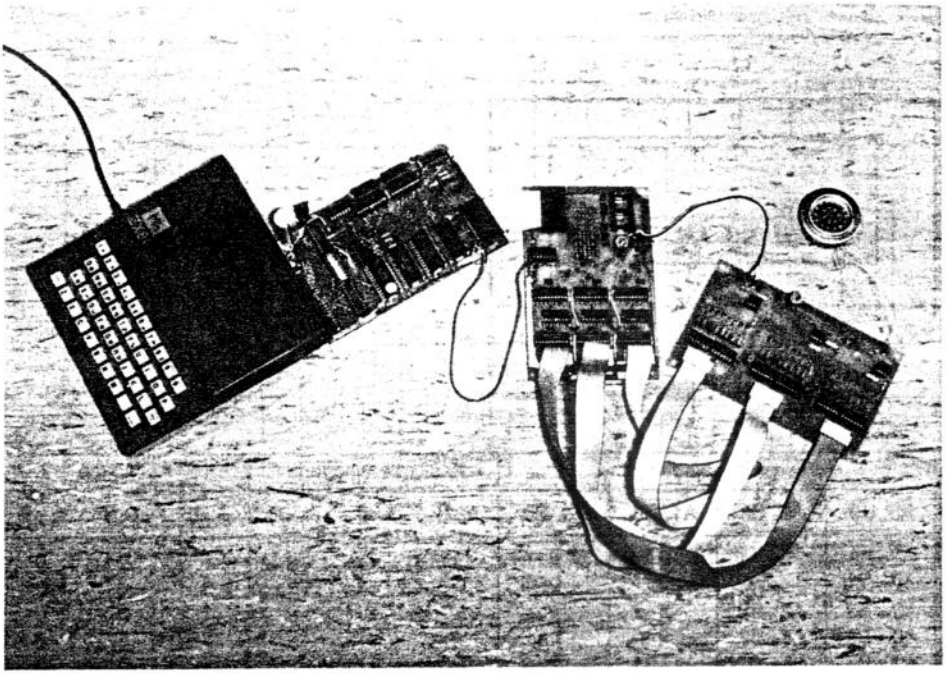


Abb. 2.12 Grundplatine I zusammen mit dem Sechsfach-OUT-Port am ZX 81. Daran angeschlossen ist die Tongenerator-Platine (C-Umschaltung).

Damit ist die Grundplatine I einsatzbereit. Anwendungen finden sich thematisch aufgegliedert an anderer Stelle in diesem Buch. *Abb.2.12* zeigt die Grundplatine I im Einsatz mit dem ZX 81.

2.3 Grundplatine II (großes System)

2.3.1 Konzeption und Adreßdekodierung

Aus der Speicherorganisation des ZX 81 (Abb.2.1) war zu erkennen, daß das BASIC-ROM im Adreßbereich 8192 ... 16383 doppelt erscheint. Natürlich ist dieses „Geister-ROM" völlig überflüssig und kann durch ein H-Signal auf der Leitung $\overline{\text{ROMCS}}$ ausgeblendet werden. Dadurch steht ein 8 KByte großer Adreßbereich zusätzlich zur Verfügung. Es liegt nahe, ihn für Erweiterungsschaltungen zu nützen. Außerdem kann in diesem Bereich auch zusätzliche Speicherkapazität zur Verfügung gestellt werden, die allerdings nicht für BASIC-Programme geeignet ist. Maschinenprogramme sind hier dagegen sehr gut aufgehoben, da sie den BASIC-Speicher nicht verkleinern und vom System auch nicht überschrieben werden können.

Um keine Probleme mit unvollständiger Adreßdekodierung zu bekommen, müssen alle 16 Adreßleitungen herangezogen werden. In Sonderfällen läßt sich eine Adreßdekodierung einfach durch die Verwendung von NAND-Gattern erreichen. Allerdings können dann beliebige Adressen nicht ohne größeren Aufwand benützt werden.

Hierfür besser geeignet ist der 4-Bit-Comparator 74LS85 (*Abb.2.13*). Ein an den Eingängen A0 ... A3 angelegtes 4-Bit-Wort (A) wird mit einem zweiten an B0 ... B3 (B) verglichen.

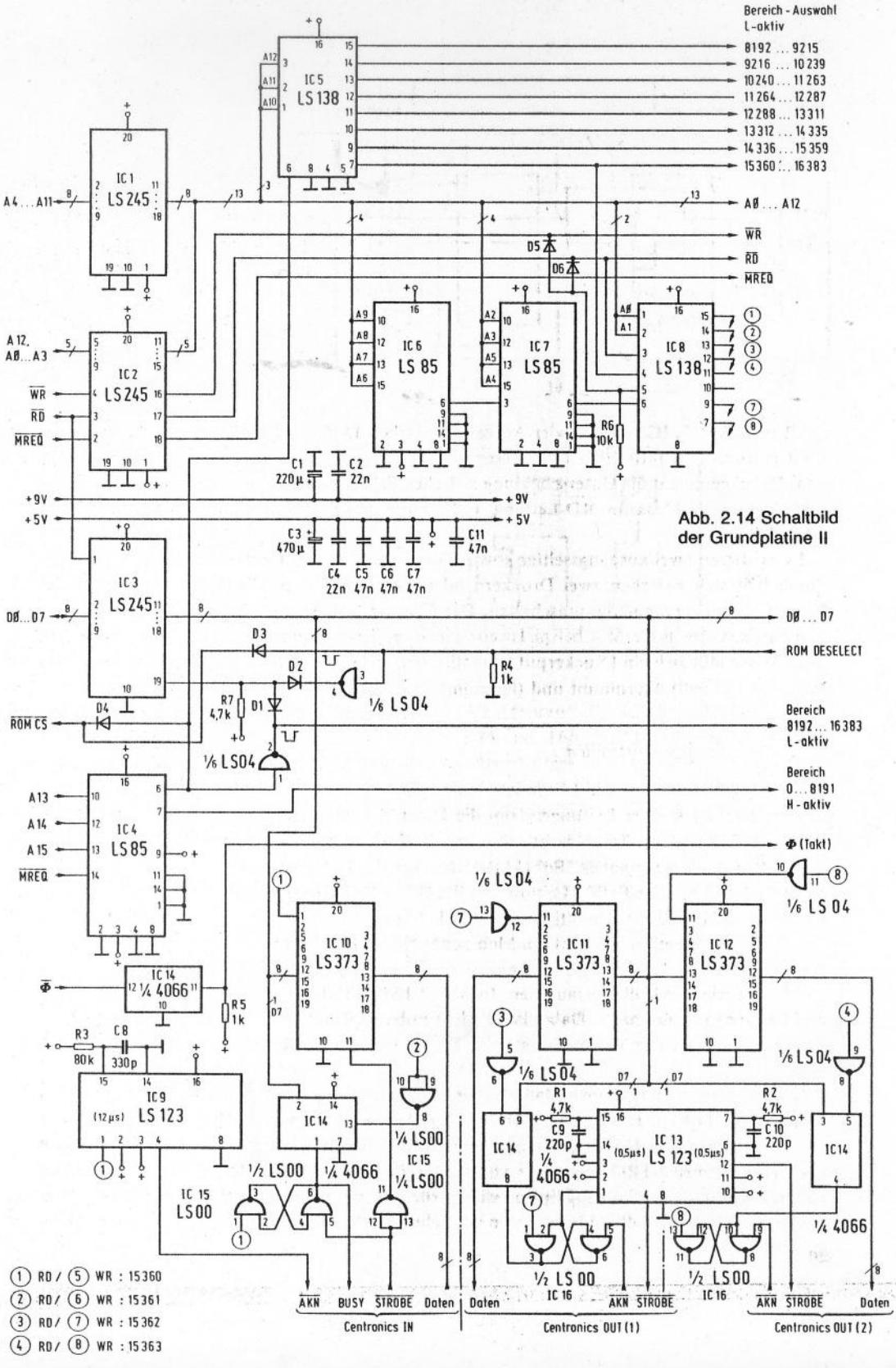


Abb. 2.14 Schaltbild der Grundplatine II

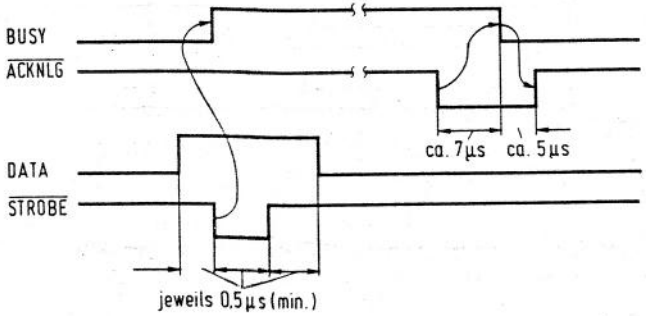


Abb. 2.15 Signale bei einer Centronics-Parallelschnittstelle

Über IC6, IC7, IC8 werden vier Adressen zwischen 15360 und 15363 festgelegt, auf denen die Centronics-Schnittstellen bzw. deren Status-Informationen angesprochen werden können. Dabei geschieht die Unterscheidung zwischen Schreib- und Leseadresse über die an Pin 3 von IC8 angeschlossene $\overline{RD}$-Leitung (vgl. auch das Logikdiagramm des 74LS138 in Abb. 2.3).

Es existieren zwei ausgangsseitige sowie eine eingangsseitige Centronics-Schnittstelle. Dadurch läßt sich zwischen zwei Druckern oder ähnlichen Peripherie-Geräten mit paralleler Schnittstelle softwaremäßig umschalten. Der Centronics-Eingang erlaubt es, parallele Daten von einem anderen Gerät - beispielsweise einem anderen Computer - zu übernehmen. Auf diese Weise läßt sich ein Druckerpuffer realisieren, indem der ZX 81 die Druckerdaten dieses Rechners (schnell) übernimmt und (langsam) an einen Drucker weiterreicht.

2.3.2 Centronics-Schnittstellen

Centronics-Schnittstellen sind Parallelschnittstellen mit einer Datenbreite von 8 Bit. Zusätzlich besitzen sie weitere Leitungen, um die Datenübertragung zu kontrollieren (*Abb.2.15*). Wenn die Daten an der Schnittstelle zur Verfügung stehen (Datenausgabe), wird der $\overline{STROBE}$-Impuls ausgegeben. Sobald der Drucker die Daten übernimmt, signalisiert er das durch H-Pegel auf der BUSY-Leitung. Ist die Datenübernahme beendet, gibt die $\overline{ACKNLG}$-Leitung (acknowledge - Bestätigung) einen L-Impuls ab. Nun können erneut Daten in die Schnittstelle eingeschrieben werden. Man nennt ein solches Verfahren „handshaking" (Händeschütteln). Für den Centronics-Eingang sind die Verhältnisse sinngemäß, nur daß ZX 81 und Peripherie ihre Rolle vertauschen. In Abb.2.15 ist zusätzlich das Zeitverhalten der einzelnen Leitungen eingetragen. Dabei ist es nicht unbedingt nötig, daß BUSY- und $\overline{ACKNLG}$-Signal gemeinsam zur Verfügung stehen. So verwenden die Ausgänge nur das $\overline{ACKNLG}$-Signal.

In Abb. 2.14 ist zu erkennen, daß die Ausgangsdaten in ein 8-bit-Latch geschrieben werden (74LS373, IC11 bzw. IC12, angesprochen über (7) bzw. (8) von IC8). Dabei wird gleichzeitig über ein SR-Flipflop (IC16) der monostabile Multivibrator (1/2 IC13, 74LS123) getriggert und erzeugt einen $\overline{STROBE}$-Impuls von 0,5 µs Dauer. Wenn der Drucker die Datenübernahme bestätigt, wird das SR-Flipflop wieder rückgesetzt. Sein Zustand wird über einen Analogschalter (IC14, 1/4 4066) beim Lesen der Schnittstellenadresse auf die Datenleitung D7 ge-

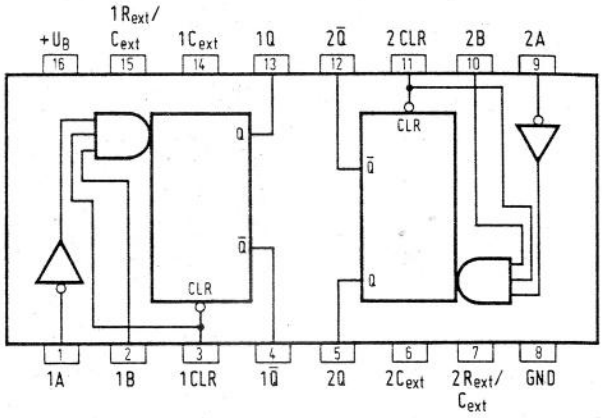


Abb. 2.16 2-fach-Monovibrator 74LS123

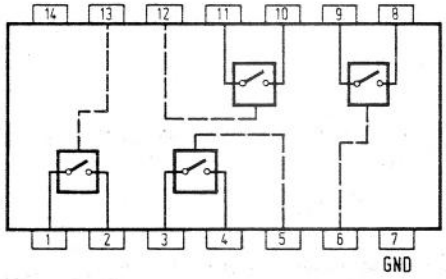


Abb. 2.17 4-fach-Analogschalter 4066

Schaltzustand	Schaltspannung
Ein	H (+5V)
Aus	L (0V)

legt. Durch Abfragen dieses Status-Bits kann ein „Überfahren" der Schnittstelle infolge einer zu hohen Datenrate vermieden werden. In *Abb. 2.16* und *Abb. 2.17* sind die Anschlußbelegungen des Monovibrators 74LS123 bzw. des Analogschalters 4066 aufgeführt.

Das Zeitverhalten der hier auftretenden Signale entspricht nicht ganz exakt den vorher beschriebenen Konventionen. Im praktischen Betrieb ergaben sich aber keinerlei Probleme.

Bei der Eingangsschnittstelle liegen die Verhältnisse entsprechend umgekehrt. Die von außen kommenden Daten liegen am Eingang von IC10 (74LS373) an, und werden mit dem $\overline{\text{STROBE}}$-Signal abgespeichert. Dabei wird gleichzeitig ein SR-Flipflop (IC15) gesetzt. Der Zustand dieses Flipflops wird beim Lesen der entsprechenden Statusadresse über einen Analogschalter (IC14, 1/4 4066) ebenfalls auf die Datenleitung D7 gelegt. Werden die Daten aus dem Latch ausgelesen, erfolgt gleichzeitig ein Rücksetzen des SR-Flipflops und das Auslösen des $\overline{\text{ACKNLG}}$-Signals (IC9, 1/2 74LS123).

In *Abb. 2.18* sind die Daten für die Bedienung der Schnittstellen noch einmal zusammengefaßt. Diverse Signale für „language card" und Erweiterungsbus sind auf die drei Steckplätze DIL1 ... DIL3 geführt. Das Platinenlayout ist in *Abb. 2.19* zu sehen. *Abb. 2.20* bringt Stückliste und Bestückungsplan der Grundplatine II. Dabei ist zu beachten, daß der Anschlußstecker für den ZX 81 von der Lötseite her montiert wird. Die Platine steht dabei senkrecht zum ZX 81. Auf der Bestückungsseite wird mit den rückwärtigen Anschlüssen des Steckers eine Kontaktleiste verlötet (senkrecht zur Grundplatine), die das Durchschleifen der Systemsignale gestattet. In *Abb. 2.21* ist die Stiftbelegung der auf der Grundplatine II verwendeten Stecker aufgelistet.

Bedienung der Centronics-Schnittstellen

CENTRONICS IN

```
Lese 15361: Status CENTRONICS IN
            D7 = 1: gültige Daten liegen an
            D7 = 0: keine gültigen Daten liegen an
Lese 15360:         Daten einlesen
```

CENTRONICS OUT 1

```
Lese 15362: Status CENTRONICS OUT 1
            D7 = 1: Daten vom Drucker noch nicht abgerufen
            D7 = 0: AKN vom Drucker eingetroffen/ Daten
                    abgerufen
Schreibe 15362:     Daten ausgeben
```

CENTRONICS OUT 2

```
Lese 15363: Status CENTRONICS OUT 2
            D7 = 1: Daten vom Drucker noch nicht abgerufen
            D7 = 0: AKN vom Drucker eingetroffen/ Daten
                    abgerufen
Schreibe 15363:     Daten ausgeben
```

Abb. 2.18 Bedienung der Centronics-Schnittstellen

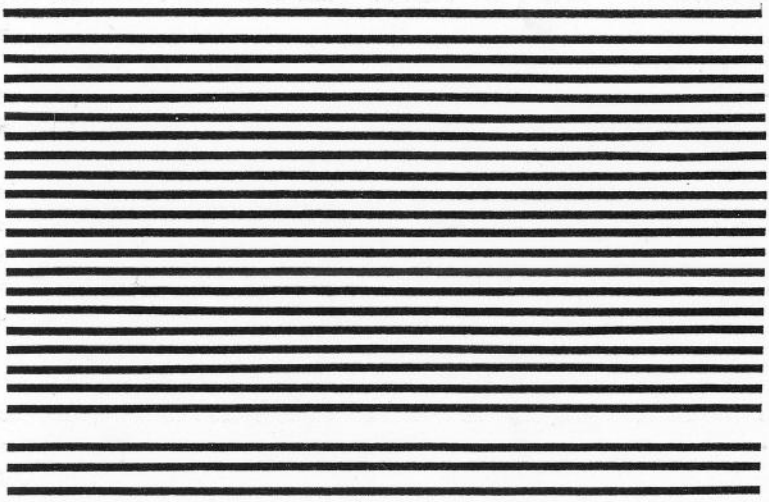


Abb. 2.19a Platinenlayout für die Kontaktleiste zur Grundplatine II zum Durchschleifen der Signalleitungen des ZX 81. Das Layout der Kontaktleiste ist für Vorder- und Rückseite gleich (Leiterbahnen verlaufen deckungsgleich), daher ist es nur einmal abgebildet. Diese Kontaktleiste wird auf der Bestückungsseite rechtwinklig zur Grundplatine angebracht und mit den Anschlüssen der Steckleiste verlötet.

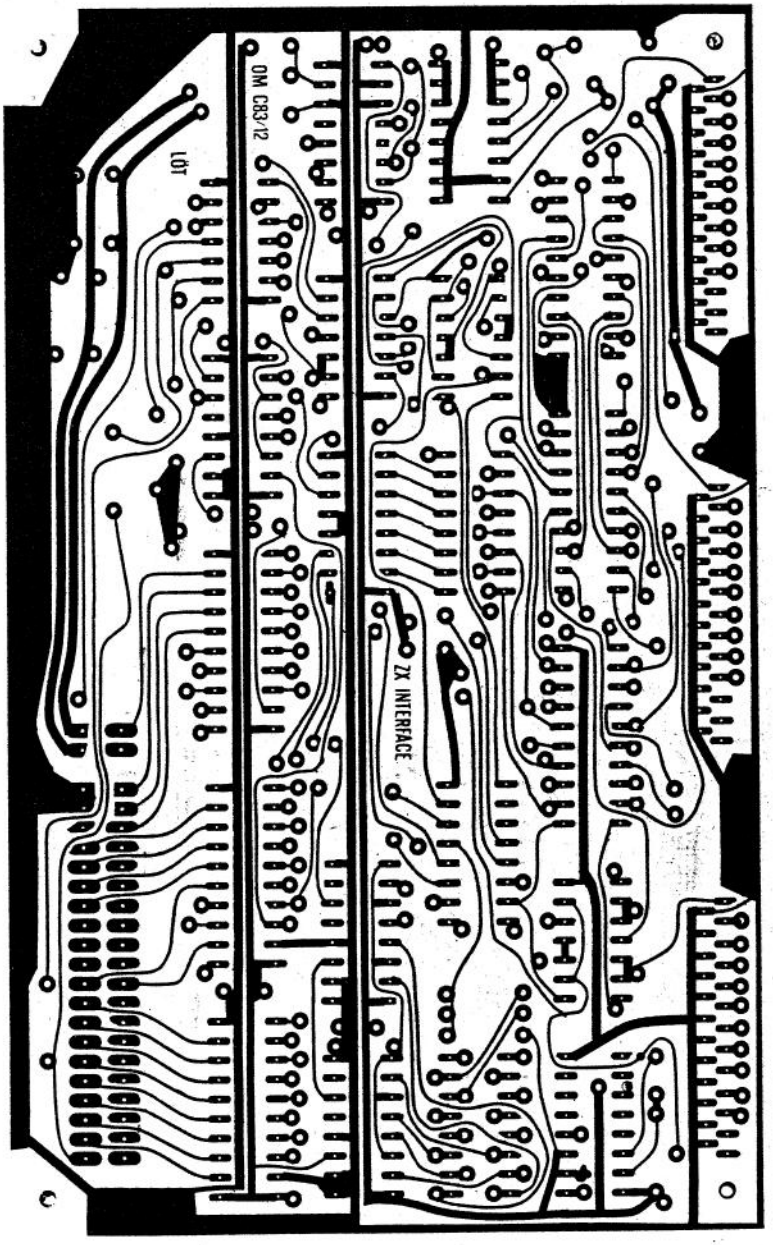


Abb. 2.19b Platinenlayout der Grundplatine II, Lötseite.

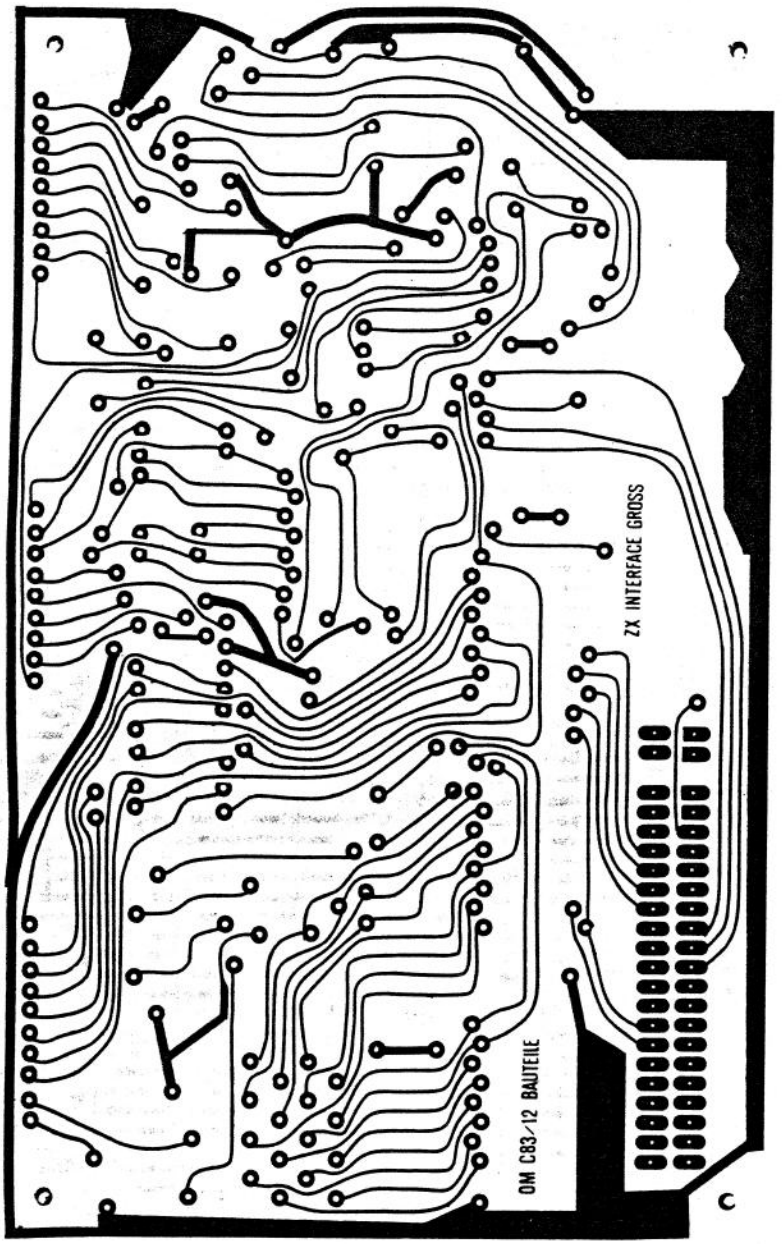


Abb. 2.19c Platinenlayout der Grundplatine II, Bestückungsseite.

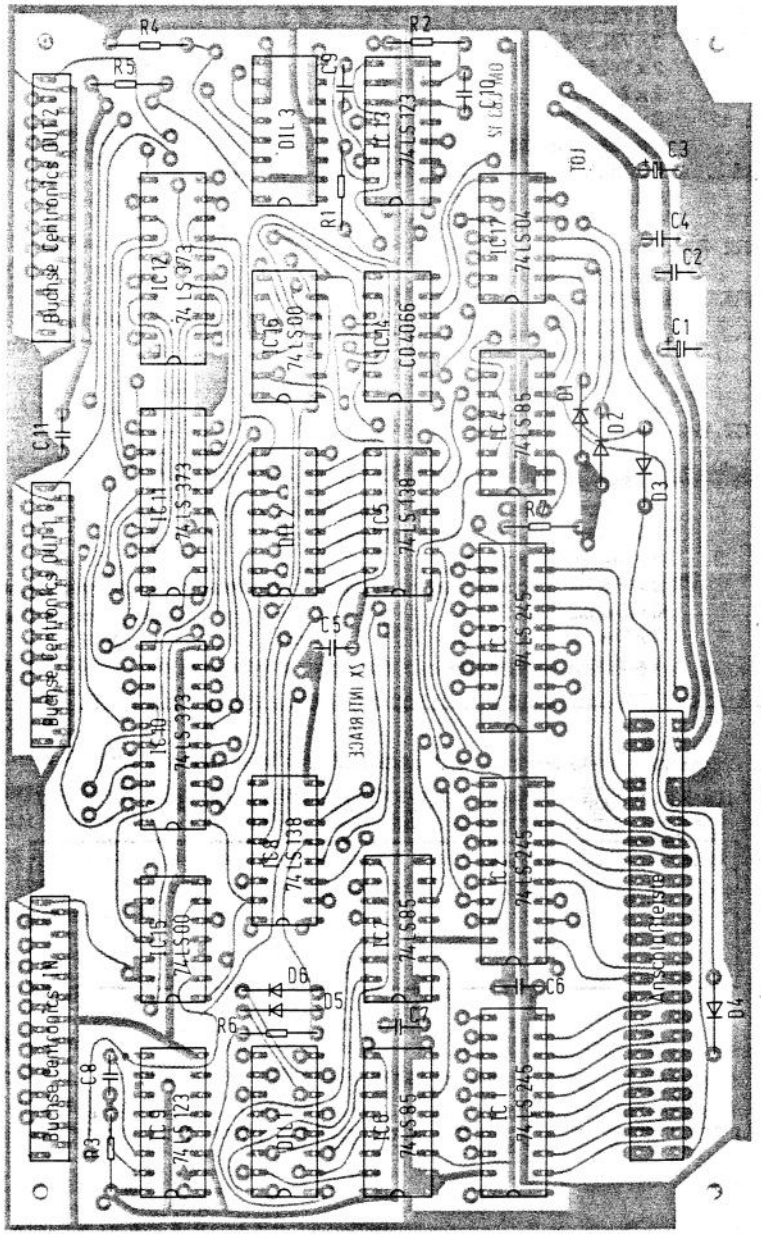
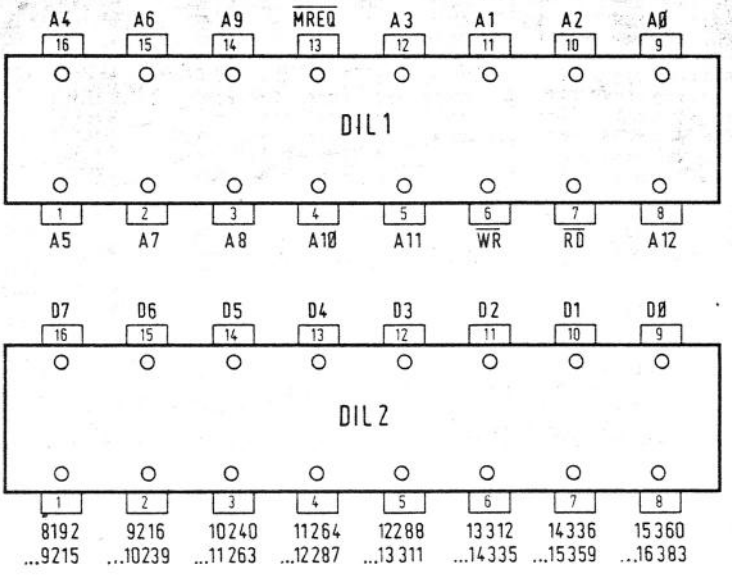


Abb. 2.20 Bestückungsplan der Grundplatine II

Stückliste der Grundplatine II

IC1 ... 3	3 × 74LS245
IC4, 6, 7	3 × 74LS85
IC5, 8	2 × 74LS138
IC9, 13	2 × 74LS123
IC10 ... 12	3 × 74LS373
IC14	1 × 4066 (CMOS)
IC15, 16	2 × 74LS00
IC17	1 × 74LS04
D1 ... D6	6 × AA 116 o. ä. Ge-Diode
R1 ... R2, R7	3 × 4,7 kΩ
R3	1 × 80 kΩ
R4 ... R5	2 × 1 kΩ
R6	1 × 10 kΩ
C1	1 × 220 µF
C2, C4	2 × 22 nF
C3	1 × 470 µF
C5 ... C7, C11	4 × 47 nF
C8	1 × 330 pF
C9, C10	2 × 220 pF
	1 × Steckerleiste 2 × 23polig, 2,54 mm für ZX 81
	3 × Normbuchse für Druckeranschluß
DIL 1 ... 3	3 × Stecksockel 18polig
	3 × Flachleitung mit Stecker 18polig für Verbindung zur „language-card"
	6 × Stecksockel 20polig
	7 × Stecksockel 16polig
	4 × Stecksockel 14polig



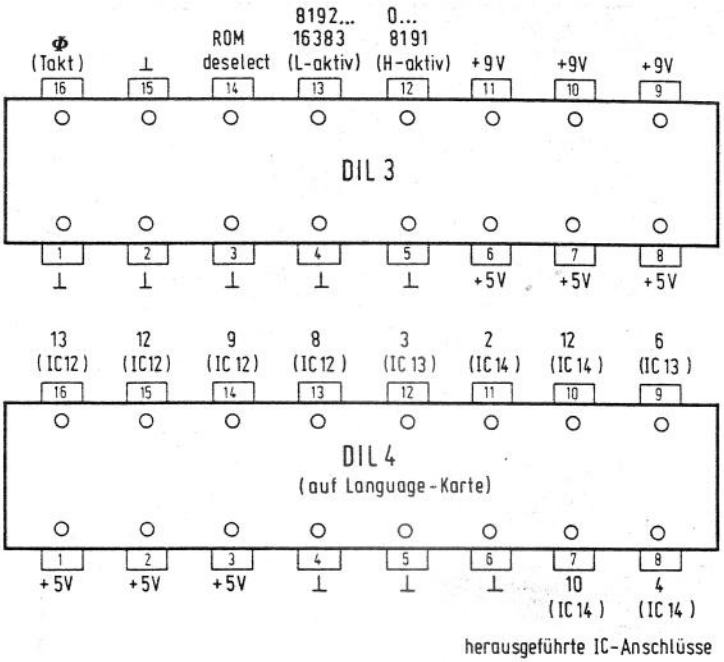


Abb. 2.21 Steckerbelegung der Grundplatine II. DIL 4 befindet sich auf der „language card"

2.3.3 Aufbau und Test

Für den Aufbau gelten sinngemäß dieselben Gesichtspunkte wie bei der Grundplatine I. *Abb.2.22* zeigt die Grundplatine II zusammen mit der „language card" (Prototypen) im Einsatz.

Nach dem Zusammenlöten wird die Platine auch hier zuerst ohne eingesteckte ICs an den ZX 81 angeschlossen. Ist kein Kurzschluß vorhanden, muß der ZX 81 arbeiten wie normal. Dann setzt man nach und nach die ICs ein, dabei selbstverständlich jedesmal die Betriebsspannung ausschalten. Sind alle ICs eingesetzt und ergaben sich keine Störungen, so prüft man am besten die Generierung der Signale für die Adreßauswahl. Hierzu schreibt man ein kleines Programm:

```
100 POKE nn, Zahl
200 GOTO 100
```

Dabei ist nn eine der interessierenden Adressen und „Zahl" ein Wert zwischen Null und 255.

Wenn dieses Programm im FAST-Modus abläuft, werden die entsprechenden Anschlüsse von IC5 und IC8 untersucht. Es sollten dabei jeweils L-Impulse zu registrieren sein. Bei IC8 muß für den Lesezugriff die Programmzeile 100 geändert werden in:

```
100 LET A = PEEK nn
```

Danach wird nn in die Adresse der jeweiligen Centronics-Schnittstelle abgeändert, um die Signale $\overline{\text{STROBE}}$ bzw. $\overline{\text{ACKNLG}}$ zu testen. Wenn alles zur Zufriedenheit funktioniert, läßt sich ein Drucker anschließen. Mit

```
POKE nn, Zahl
```

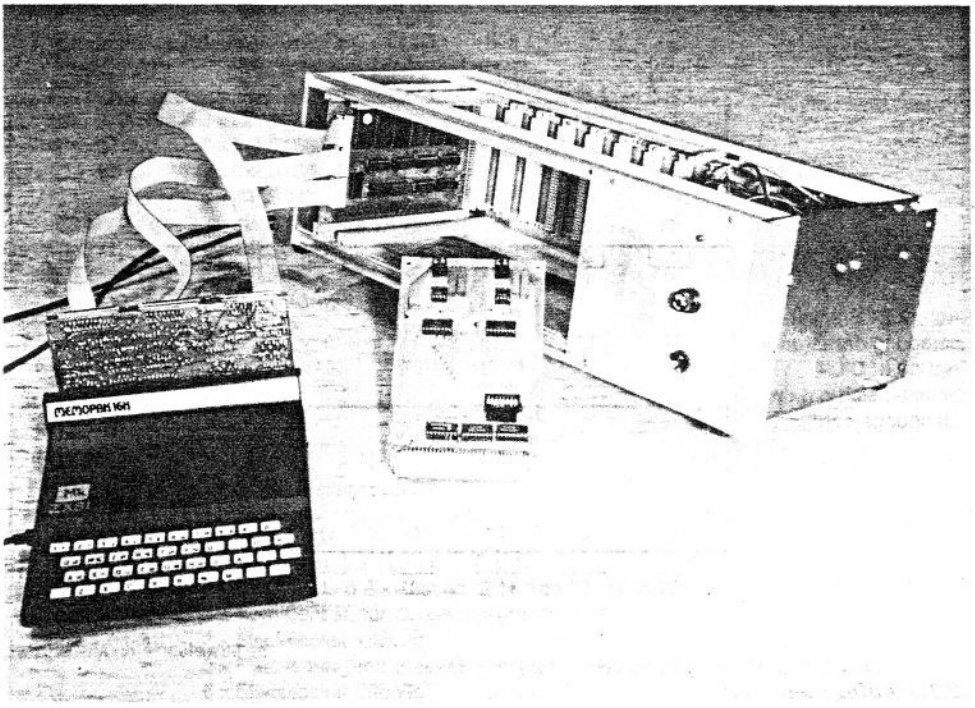


Abb. 2.22 Grundplatine II mit „language card" am ZX 81. Im Vordergrund ist die D/A-Wandlerkarte für System II zu sehen.

wird der zu „Zahl" gehörende Code ausgedruckt (Achtung: manche Drucker geben keine Einzelzeichen, sondern immer erst eine vollständige Zeile, also z. B. 80 Zeichen, aus).

Über die Einbindung der Druckerschnittstellen in die entsprechenden Routinen des ZX 81 (LLIST, LPRINT, COPY) wird bei den Anwendungen der „language card" ausführlich berichtet.

Falls nicht alle Möglichkeiten der Grundplatine II genützt werden sollen, können natürlich die entsprechenden Schaltungsteile weggelassen werden, ohne daß die Funktion des Restes leidet. So kann beispielsweise auf den Centronics-Eingang verzichtet werden, indem IC9, IC10, IC15 sowie die dazugehörigen Bauelemente nicht eingesetzt werden. Allerdings sind einige ICs für die Funktion unerläßlich, z. B. die Bustreiber. Diese Bauteile müssen also auf jeden Fall bestückt werden. Anhand des Schaltbildes kann leicht geklärt werden, wo Abmagerungsmöglichkeiten gegeben sind.

2.4 Language Card

2.4.1 Schaltungsbeschreibung

Abb.2.23 zeigt das Schaltbild der „language card". Sie erhält über drei Flachkabel verschiedene Signale von der Grundplatine II. Soweit die „language card" diese Signale nicht selbst benötigt, werden sie zum Erweiterungsbus durchgeschleift.

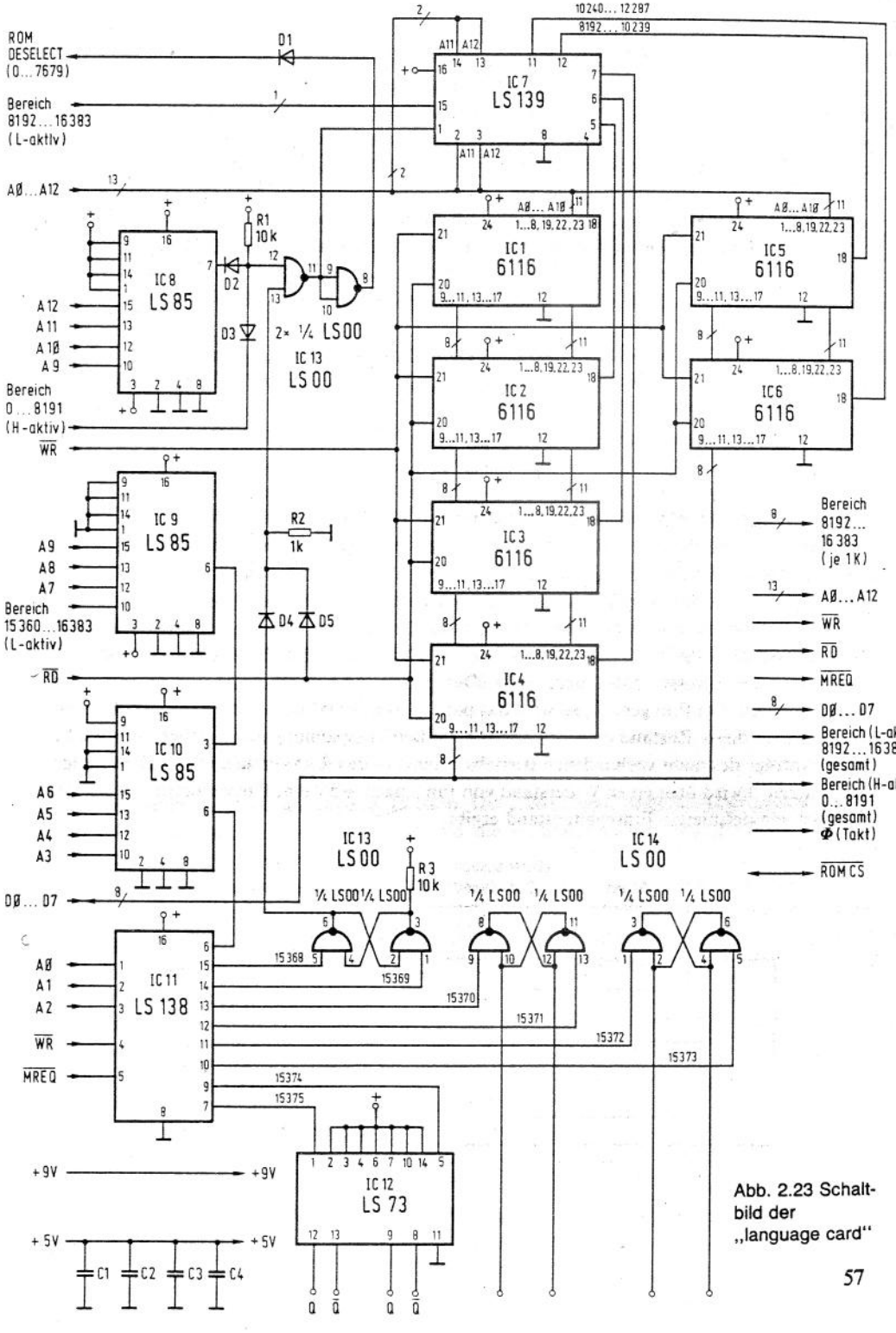


Abb. 2.23 Schaltbild der „language card"

IC8 (74LS85) dient der Adreßauswahl für die RAM-Speicher, die parallel zum ZX-81-ROM liegen (IC1 ... IC4). Eigentlich würde man erwarten, daß hierzu das im Bereich 0 ... 8191 aktive Signal allein ausreicht. Beim ZX 81 kommt aber eine kleine Besonderheit dadurch dazu, daß im Bereich 7680 ... 8191 der Zeichengenerator untergebracht ist. Das wäre auch nicht weiter schlimm, wenn sich nicht aus Gründen, die durch die Hardware gegeben sind, der Zeichengenerator im ROM-Sockel des ZX 81 befinden müßte. Näheres ist hierzu im Kapitel 1.4.3 nachzulesen. Daher ist es nötig, den Zeichengenerator aus dem ROM zu verwenden und den RAM-Speicher im Adreßbereich 7680 ... 8191 auszublenden.

Diese 512 Byte sind angesprochen, wenn A9, A10, A11 und A12 H-Pegel führen. Daher zieht IC8 diese Leitungen zur Bereichsauswahl heran. Der RAM-Speicher soll eingeschaltet werden bei Adressen die kleiner sind als die des Zeichengenerators. Daher wird Pin 7 des 74LS85 über Dioden mit der Bereichsauswahl 0 ... 8191 (jeweils H-aktiv) UND-verknüpft. Das folgende NAND-Gatter sorgt dafür, daß das RAM nur aktiviert wird, wenn ein Software-Schalter entsprechend gesetzt ist. In diesem Fall steht an Pin 13 von IC13 ebenfalls H-Pegel und der Ausgang (Pin 11) geht auf L.

Mit diesem Signal wird eine Hälfte des Adreßdekoders 74LS139 (IC7) aktiviert. Die Adressen A11 und A12 wählen dann jeweils einen 2-KByte-Bereich aus und steuern dadurch den entsprechenden Speicherbaustein (IC1 ...IC4) an. Die Adreßleitungen A0 ... A10 werden den Speicher-ICs direkt zugeführt. Um das interne ROM während des RAM-Zugriffs abzuschalten, wird das Select-Signal für IC7 nochmals invertiert und über eine Entkopplungsdiode der Leitung $\overline{\text{ROMCS}}$ zugeführt.

Abb. 2.24 zeigt das Anschlußbild und die Funktionstabelle des 74LS139.

IC9, IC10 und IC11 dienen zur vollständigen Dekodierung der Adressen für verschiedene Software-Schalter. So wird über die Adresse 15368 das aus 1/2 IC13 gebildete Flipflop gesetzt, über die Adresse 15369 rückgesetzt. Dabei müssen jeweils Schreibbefehle verwendet werden. Ist das Flipflop gesetzt, so wird das parallel zum ROM liegende RAM angesprochen (s.o.). Damit dieser Zustand beim Einschalten der Betriebsspannung nicht vorliegt (der ZX 81 würde infolge des nicht vorhandenen Betriebssystems — das RAM ist ja leer — hoffnungslos „abstürzen") wird über einen Widerstand von Pin 3 nach + 5V eine Unsymmetrie erzeugt, so daß sich ein definierter Einschaltzustand ergibt.

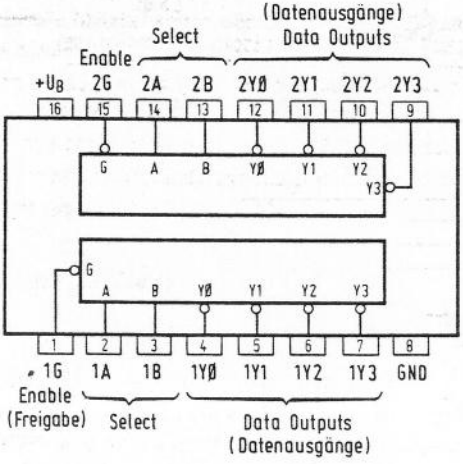


Abb. 2.24 Adreßdekoder 74LS139: Anschlußbild und Funktionstabelle

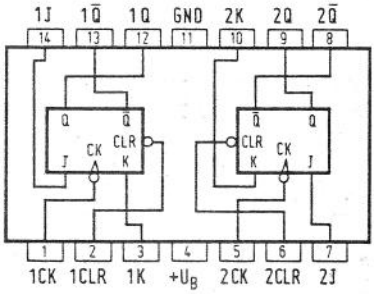


Abb. 2.25 2-fach-JK-Flipflop 74LS73

Auf der Platine sind noch weitere Software-Schalter vorhanden, die frei verwendet werden können. Ihre Ausgänge sind an der Buchse DIL4 vorhanden. Es handelt sich dabei um zwei SR-Flipflops (jeweils 1/2 IC14) und zwei „toggle-switches". Die beiden SR-Flipflops verwenden die bereits oben erläuterte Technik. Ein „toggle-switch" ändert seinen Schaltzustand bei jedem auf der betreffenden Adresse eingehenden Impuls. Ein solches Schaltverhalten läßt sich mit einem JK-Flipflop 74LS73 (IC12) realisieren. Das IC enthält zwei derartige Flipflops, sein Anschlußbild ist in *Abb.2.25* dargestellt. Auch diese Schalter werden durch Schreibbefehle betätigt.

Die zweite Hälfte von IC7 dient zur Ansteuerung von zwei weiteren Speicherbereichen (IC5 und IC6), die zwischen 8192 und 10239 bzw. 10240 und 12287 liegen. Hier können statt der CMOS-RAMs auch EPROMs 2716 bzw. 2516 verwendet werden (Vorsicht: bei den EPROMs Anschlußbilder überprüfen, hier gibt es je nach Hersteller unterschiedliche Belegungen, auch bei gleicher Typenbezeichnung!). Auch das in Kapitel 7 beschriebene Pseudo-ROM kann Verwendung finden. Die Auswahl von IC5 und IC6 erfolgt über die Adressen A11 und A12.

2.4.2 Aufbau und Inbetriebnahme

Das Platinenlayout der „language card" ist in *Abb.2.26* gezeigt, Bestückungsplan und Stückliste findet man in *Abb.2.27*. In *Abb.2.28* sind alle wichtigen Adressen für die Bedienung der „language card" zusammengestellt.

Für den Zusammenbau gelten die üblichen Gesichtspunkte. Zur Inbetriebnahme wird die „language card" mit der Grundplatine II über drei jeweils 16polige Flachbandkabel, die in die entsprechenden DIL-Stecker eingesetzt werden, verbunden. Dabei ist sorgfältig darauf zu achten, daß auf beiden Platinen jeweils die entsprechenden Pins miteinander in Kontakt sind und die Stecker nicht um 180° verdreht eingeführt werden.

Sind diese Voraussetzungen gegeben, so kann über Schreib-/Lesebefehle (POKE/PEEK) getestet werden, ob die Speicher IC5 und IC6 ordnungsgemäß arbeiten. Es sei nochmals darauf hingewiesen, daß diese Speicher nicht für BASIC-Programme geeignet sind. Sie sind zum Abspeichern von Maschinenprogrammen gedacht.

Die Software-Schalter testet man, indem die Ausgangsleitungen an DIL4 mit einem Voltmeter überwacht werden, während die entsprechenden Schreibbefehle gegeben werden. Dabei darf allerdings nicht der für die ROM/RAM-Umschaltung zuständige Schalter betätigt werden, sonst stürzt der Rechner sofort ab. In einem solchen Fall muß aus- und erneut eingeschaltet werden.

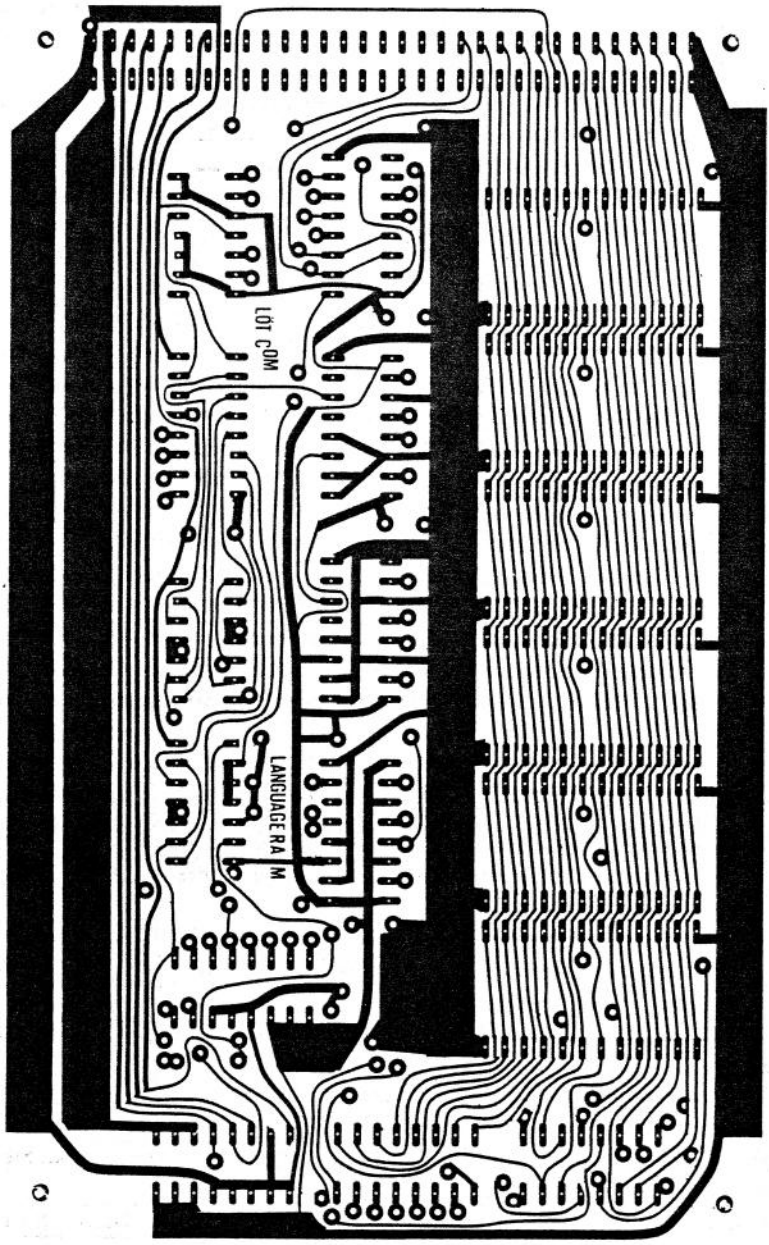


Abb. 2.26a Platinenlayout der „language card", Lötseite

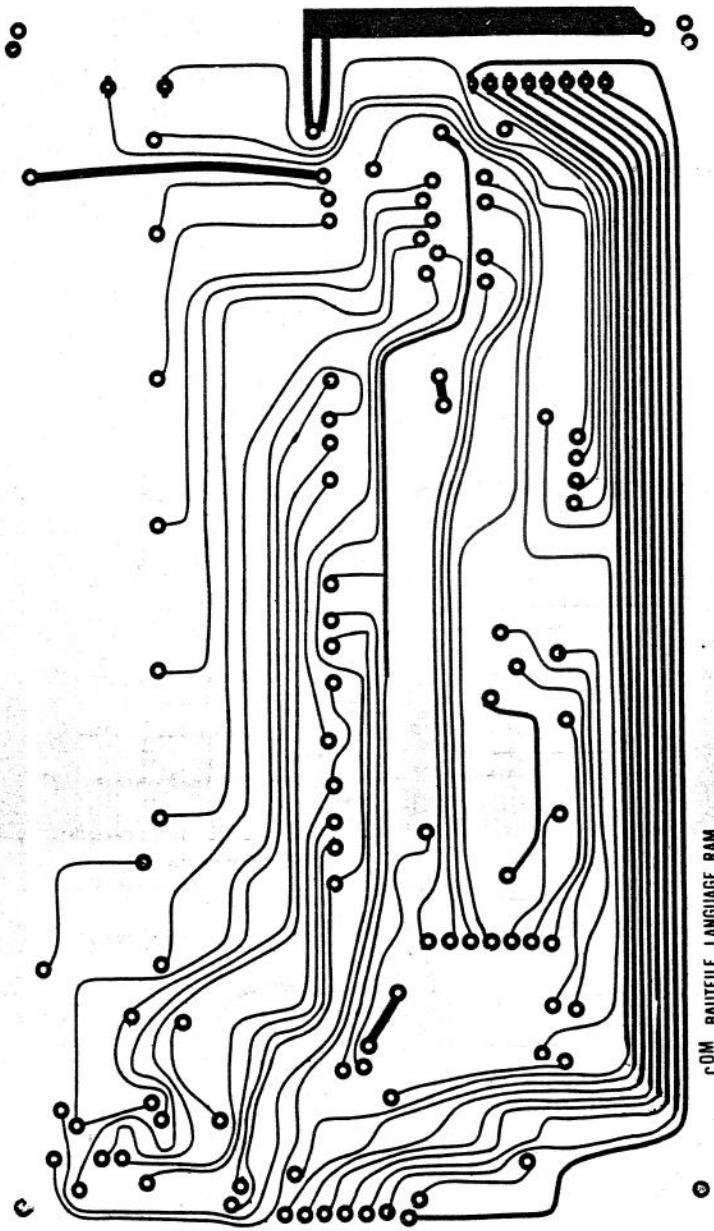


Abb. 2.26b Platinenlayout der „language card", Bestückungsseite

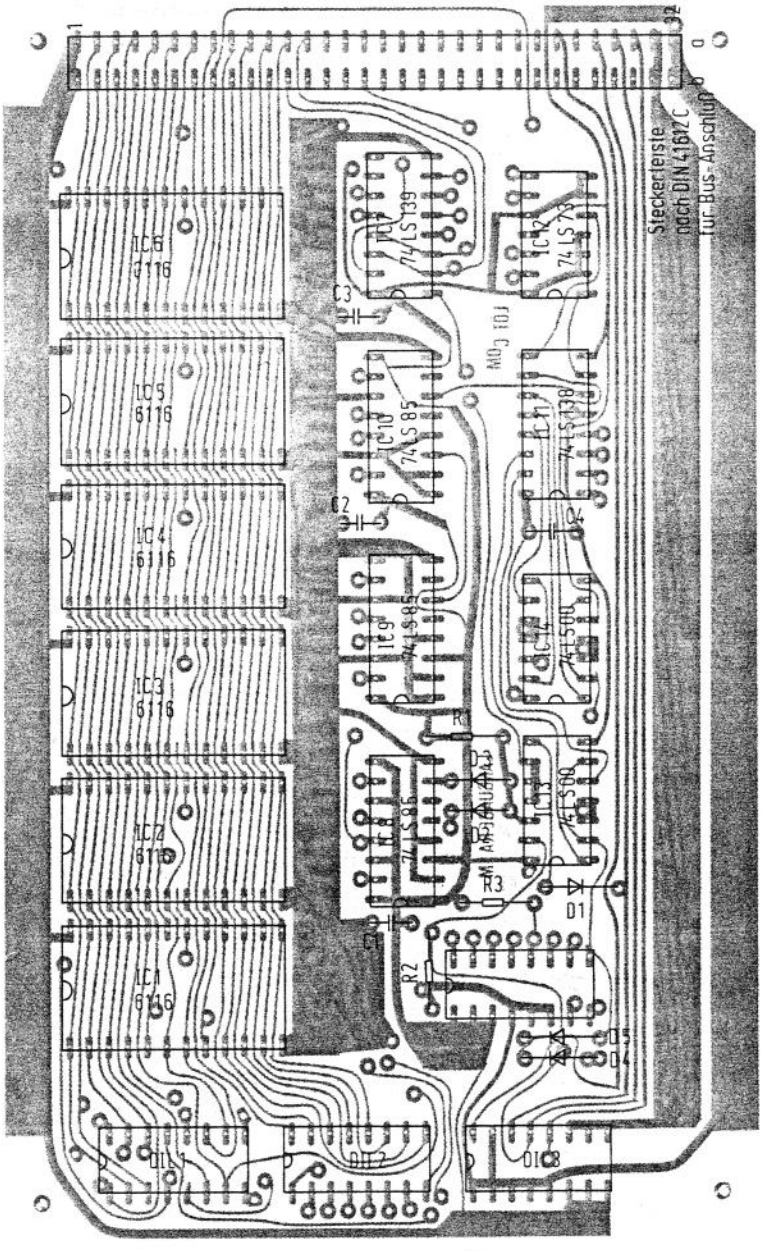


Abb. 2.27 Bestückungsplan zur „language card"

Stückliste „language card"

IC1 ... IC6	6 x HM6116
IC7	1 x 74LS139
IC8 ... IC10	3 x 74LS85
IC11	1 x 74LS138
IC12	1 x 74LS73
IC13, IC14	2 x 74LS00
D1 ... D5	5 x AA 116 o. ä. Ge-Diode
C1 ... C4	4 x 47 nF
R1, R3	2 x 10 kΩ
R2	1 x 1 kΩ
	6 x Steckfassung 24polig
	9 x Steckfassung 16polig
	3 x Steckfassung 14polig
	1 x Steckleiste 64polig
	nach DIN 41612 Bauform C

Systemadressen der "language card"

15368 ... Einschalten des RAMs (vorher mit Betriebssystem laden, sonst stürzt Computer ab)

15369 ... Einschalten des ROMs

15370 ... Setzen von Flipflop I

15371 ... Rücksetzen von Flipflop I

15372 ... Setzen von Flipflop II

15373 ... Rücksetzen von Flipflop II

15374 ... Schaltimpuls "toggle switch I"

15375 ... Schaltimpuls "toggle switch II"

Wichtig: Alle Systemadressen müssen mit Schreibbefehlen bedient werden.

Abb. 2.28 Adressen für die Bedienung der „language card"

Bevor man umschalten kann, muß das ROM in das parallel liegende RAM umkopiert werden. Dies läßt sich durch das folgende kleine BASIC-Programm erreichen:

```
100 FOR N = 0 TO 8191
200 POKE N, PEEK N
300 NEXT N
```

Jetzt kann man mit

```
POKE 15368,0
```

auf den RAM-Speicher zugreifen. Es sollte dabei nicht verwundern, daß die ersten fünf Byte nun einen anderen Inhalt besitzen, sie werden vom ZX 81 überschrieben. Da dieser Bereich nur nach dem Einschalten (bzw. bei RESET) durchlaufen wird, ist das während des Betriebs unwesentlich.

2.5 Systemzusammenstellung

Die Grundplatinen (I und II) sind Basis für die Zusammenstellung eines Systems, mit dem sich verschiedene spezifische Problemstellungen lösen lassen. Dazu gehören weitere Funktionsplatinen, die in den Kapiteln 3 ... 6 beschrieben sind. *Abb.2.29* zeigt die Gesamtsysteme im Überblick.

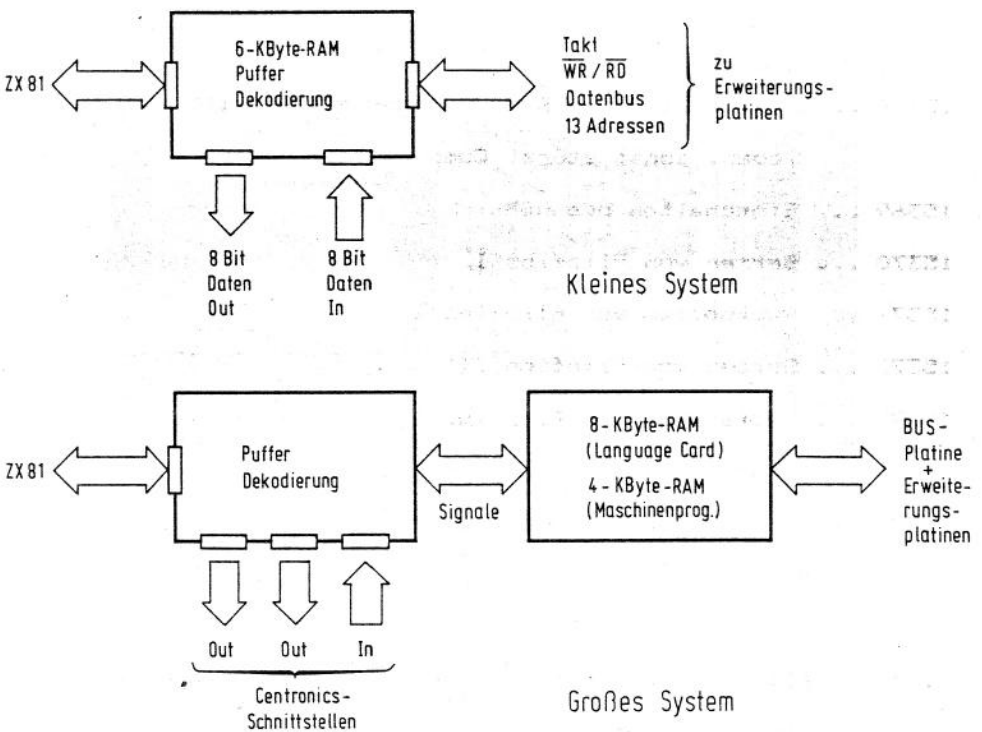


Abb. 2.29 Systemüberblick

Beim „kleinen" System (System I) können an der 24poligen DIL-Buchse, neben weiteren Signalen, 13 dekodierte Adressen abgegriffen werden, die nach eigenen Wünschen mit den Funktionsplatinen belegt werden. Der Signalbus ist auf den Funktionsplatinen durchgeschleift, so daß verschiedene dieser Platinen kombinierbar sind.

Für das „große" System (System II) erfolgt auf den Grundplatinen keine Dekodierung von Einzeladressen, lediglich eine Dekodierung von 1-KByte-Blöcken. Alle wichtigen Signale sind auf eine 64polige Normleiste geführt (*Abb.2.30*). Diese Normleiste (Stecker) kann in eine Buchse einer BUS-Platine eingesteckt werden, die die anderen Funktionsplatinen trägt (vergl. auch Abb.2.22).

Die Funktionsplatinen für System I und System II stimmen weitgehend überein. Der wesentliche Unterschied besteht in der Adressenauswahl. Beim System I wird die gewünschte Adresse über ein Lötfeld aus den 13 dekodierten Adressen ausgewählt. In Kapitel 3.1.2 ist dies näher erläutert.

	a		c	
+5V	1	1	1	+5V
A7	2	2	2	A8
A6	3	3	3	A9
A5	4	4	4	$\overline{WR}$
A4	5	5	5	$\overline{RD}$
A3	6	6	6	A10
A2	7	7	7	A1
D7	8	8	8	A0
D6	9	9	9	D0
D5	10	10	10	D1
D4	11	11	11	D2
D3	12	12	12	A12
A11	13	13	13	NC
NC	14	14	14	NC
NC	15	15	15	NC
NC	16	16	16	15360...16383
NC	17	17	17	NC
NC	18	18	18	−5V
NC	19	19	19	NC
NC	20	20	20	+12V / 15V
NC	21	21	21	−12V / 15V
NC	22	22	22	NC
NC	23	23	23	$\overline{MREQ}$
NC	24	24	24	8192...9215
NC	25	25	25	9216...10239
Φ(Takt)	26	26	26	10240...11263
NC	27	27	27	11264...12287
ROM Deselect	28	28	28	12288...13311
8192...16383 (L-aktiv)	29	29	29	13312...14335
0...8191 (H-aktiv)	30	30	30	14336...15359
+9V	31	31	31	+9V
GND	32	32	32	GND

Abb. 2.30 BUS-Belegung für System II

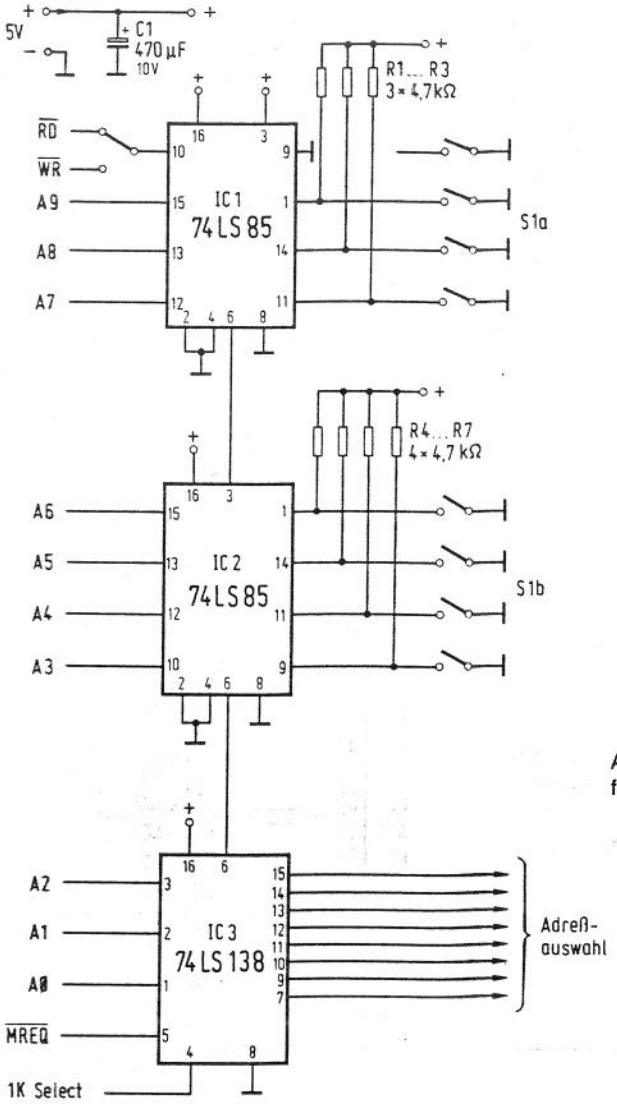
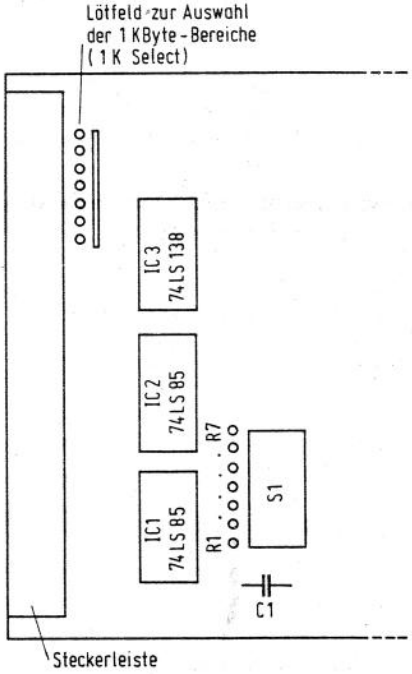


Abb. 2.31 Adreßdekodierung für System II

Da bei System II die Grundplatine keine Adreßdekodierung übernimmt, muß dies auf den Funktionsplatinen durchgeführt werden. Daher ist der eigentlichen Schaltung noch die Dekoderschaltung nach *Abb.2.31* vorangestellt. Sie besteht aus zwei Dekodern 74LS85, die die durch Minischalter (auch direktes Verlöten ist möglich) vorgegebene Binärzahl mit dem auf dem Adreßbus vorliegenden Bitmuster vergleicht. Bei Übereinstimmung wird der 1-zu-8-Dekoder 74LS138 aktiviert. Über den Eingang „1k select" kann die gewählte Adresse in den jeweils gewünschten Bereich zwischen 8192 und 16383 gelegt werden. Die Auswahl erfolgt über eine Lötbrücke auf der Platine. Dabei muß aber unbedingt beachtet werden, daß einige dieser



Stückliste der Adreßdekodierung für Funktionsplatinen (System II)

IC1, IC2	2 × 74LS85
IC3	1 × 74LS138
C1	1 × 470 µF/10V
R1 ... R7	7 × 4,7 kΩ (senkrecht verlötet) oder Widerstands-Netzwerk (SIL)
S1	1 × DIL-Switch (8polig)
	4 × Steckfassung 16polig
	1 × Steckerleiste 64polig nach DIN 41612 Bauform C

Abb. 2.32 Bestückungsplan des Dekodierteils der Funktionsplatinen

Adressen u. U. belegt sind (Maschinenprogrammspeicher 8192 ... 10239, sowie Centronics-Schnittstellen im Bereich 15360 ... 16383). Eine Doppelbelegung kann zu Hardwareschäden führen.

Falls eine Funktionsplatine nur mit Schreib- bzw. Lesebefehlen angesprochen wird, ist darüber hinaus die $\overline{WR}$- bzw. $\overline{RD}$-Leitung in die Dekodierung mit einbezogen. Dadurch entfallen u. U. Bauteile im Vergleich zu System I.

Bei den Funktionsplatinen ist der Schaltplan sowie der Bestückungsplan jeweils nur für System I dargestellt. Daher zeigt *Abb.2.32* Bestückungsplan und Stückliste für die Adreßdekodierung bei den Funktionsplatinen im System II. Die 64polige Normleiste muß, soweit die entsprechenden Anschlüsse auf der Platine verwendet werden, auch von der Bestückungsseite her verlötet werden.

Über Kombinationsmöglichkeiten der in diesem Buch vorgestellten Hardwarezusätze mit im Handel erhältlichem Zubehör, insbesondere Speichererweiterungen lassen sich keine allgemein gültigen Aussagen machen, hier ist man auf Probieren angewiesen. Dabei ist allerdings Vorsicht geboten, da im Falle von Doppelbelegungen der Adressen Hardwareschäden nicht auszuschließen sind. Eine Garantie kann nicht gegeben werden!

Bei der Grundplatine I wird eine Kombination mit Zusatzspeichern in der Regel nicht zur Diskussion stehen. Anders sieht es dagegen bei Grundplatine II aus. Sie kann mit der Speichererweiterung von Grundplatine I zusammen betrieben werden. Getestet wurde sie ebenfalls *hinter* einer 16-KByte-RAM-Erweiterung von Memopack. Dabei müssen die Kodierschalter am 16-KByte-RAM richtig eingestellt werden (ausprobieren).

Umfaßt das System eine größere Anzahl von Platinen, so muß besonderes Augenmerk auf die Stromversorgung gelegt werden. Das Netzteil des ZX 81 ist dann in der Regel überlastet. Neben einer thermischen Überbeanspruchung verschlechtert sich auch die Spannungsstabilität, so daß die einwandfreie Funktion von Schaltungen nicht mehr gewährleistet ist. Besonders kritisch sind in diesem Zusammenhang dynamische Zusatzspeicher, wie sie üblicherweise im Handel angeboten werden.

Wird ein externes Netzteil für die Zusatzplatinen verwendet, so müssen die Versorgungsleitungen auf den jeweiligen Grundplatinen zum ZX 81 hin unterbrochen werden, andernfalls können Schäden auftreten.

3 Digitale Ein-/Ausgabekarten (I/O-Ports)

3.1 Sechsfach OUT-Port

3.1.1 Schaltung

Die Grundplatine I enthält nur einen Ausgabe-Port. Auf der Grundplatine II sind zwar zwei Centronics-Ausgänge vorhanden, allerdings dienen sie speziellen Zwecken. Daher taucht sehr schnell der Wunsch nach weiteren Ausgabeleitungen auf. Die hier beschriebene Erweiterungsplatine bietet sechs weitere OUT-Ports mit jeweils acht Bit Datenbreite, insgesamt sind das also 48 einzelne Ausgabeleitungen. Für viele Anwendungsfälle ist man damit bereits ausreichend versorgt. Falls nicht, können selbstverständlich mehrere Platinen verwendet werden, soweit die verfügbaren Adressen ausreichen (beim „großen" System kein Problem).

Im Falle der Grundplatine I wird die Zusatzkarte an die 24polige DIL-Buchse (DIL 1) angeschlossen, bei der Grundplatine II wird der Erweiterungsbus verwendet (vergl. Kap. 2.1). Das Platinenlayout ist in beiden Fällen unterschiedlich. Da beim „großen" System keine dekodierten Adressen vorhanden sind, wird die in Kapitel 2.5 beschriebene Adreßdekodierung vorgeschaltet. Im folgenden wird auf diese Unterschiede nicht weiter eingegangen.

Die Adresse, unter der der jeweilige Ausgang angesprochen wird, kann vom Anwender selbst gewählt werden (beim kleinen System über Lötbrücken, beim großen über DIL-Schalter bzw. Lötbrücken).

Abb.3.1 zeigt das Schaltbild der Platine für System I. Es findet dasselbe Schaltungsprinzip Anwendung wie bei dem OUT-Port auf der Grundplatine. Ein 8fach-Latch 74LS373 übernimmt die auf dem Datenbus liegende Information, solange sein Enable-Eingang (Pin 11) H-Potential führt.

Das soll der Fall sein, wenn die gewünschte Adresse angesprochen wird und gleichzeitig ein Schreibimpuls vom ZX 81 geliefert wird. Die hierzu nötige UND-Verknüpfung dieser beiden Signale erfolgt mit Hilfe eines Sechsfach-Bustreibers 74LS366 *(Abb. 3.2).* Solange seine beiden Enable-Eingänge (Pin 1 und Pin 15) gleichzeitig L-Pegel führen, werden die sechs invertierenden Bustreiber aktiviert. In der Schaltung ist Pin 15 mit der Schreibleitung verbunden, während Pin 1 ständig an Masse gelegt ist. Die dekodierten Adreßsignale werden den Eingängen des 74LS366 zugeführt.

Falls nun die gewünschte Adreßleitung angesprochen wird (L-aktiv), steht am jeweiligen Ausgang des Bustreibers ein H-Signal zur Ansteuerung des dazugehörigen Latch-IC zur Verfügung.

Solange der Pin 15 von IC1 H-Pegel führt, befinden sich die Ausgänge im hochohmigen Zustand. Um zu vermeiden, daß der aus dem TTL-Eingang des Latch-IC (Pin 11) herausfließende Strom einen Spannungsabfall und damit ein H-Signal verursacht, sind alle Enable-Eingänge der Latch-ICs über 1 kΩ mit Masse verbunden.

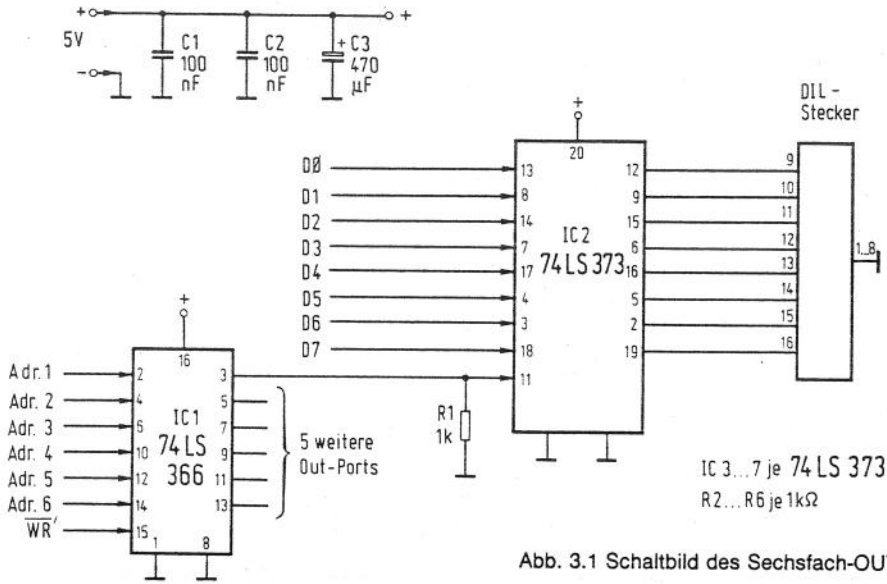


Abb. 3.1 Schaltbild des Sechsfach-OUT-Ports

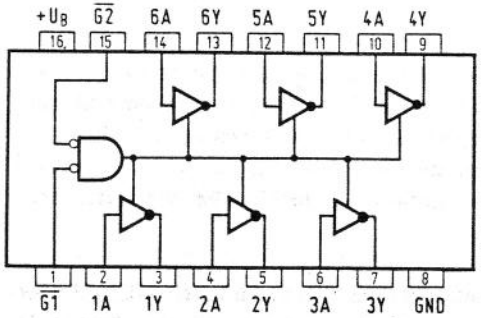


Abb. 3.2 Anschlußbild des Bustreibers 74LS366

Die Ausgänge von IC2 ... IC7 sind jeweils an 16polige DIL-Fassungen geführt, über die die gewünschte Peripherie mit Flachkabeln angeschlossen werden kann. Dabei ist zu beachten, daß die Stecker nicht verdreht eingesetzt werden, da sonst Pin 1 der einen Seite mit Pin 9 der anderen Seite verbunden wird. Unter Umständen kann sich dadurch ein Kurzschluß ergeben.

3.1.2 Aufbau und Inbetriebnahme

Die Schaltung wird auf einer zweiseitig kupferkaschierten Platine aufgebaut. *Abb.3.3* zeigt das Platinenlayout für System I, *Abb.3.4* für System II. Der Bestückungsplan und Stückliste in *Abb.3.5* sind nur auf System I bezogen, die für System II nötigen Zusatzteile sind in Kapitel 2.5 aufgeführt.

Vor dem Einsetzen der Bauteile und IC-Fassungen müssen wie üblich alle von der Bestückungsseite her sichtbaren Lötaugen durchkontaktiert werden.

Ausnahme von dieser Regel sind die Lötaugen zwischen den beiden 24poligen DIL-Buchsen. Sie dienen als eine Art von Kreuzschienenverteiler zur Auswahl der für den jeweiligen Port ge-

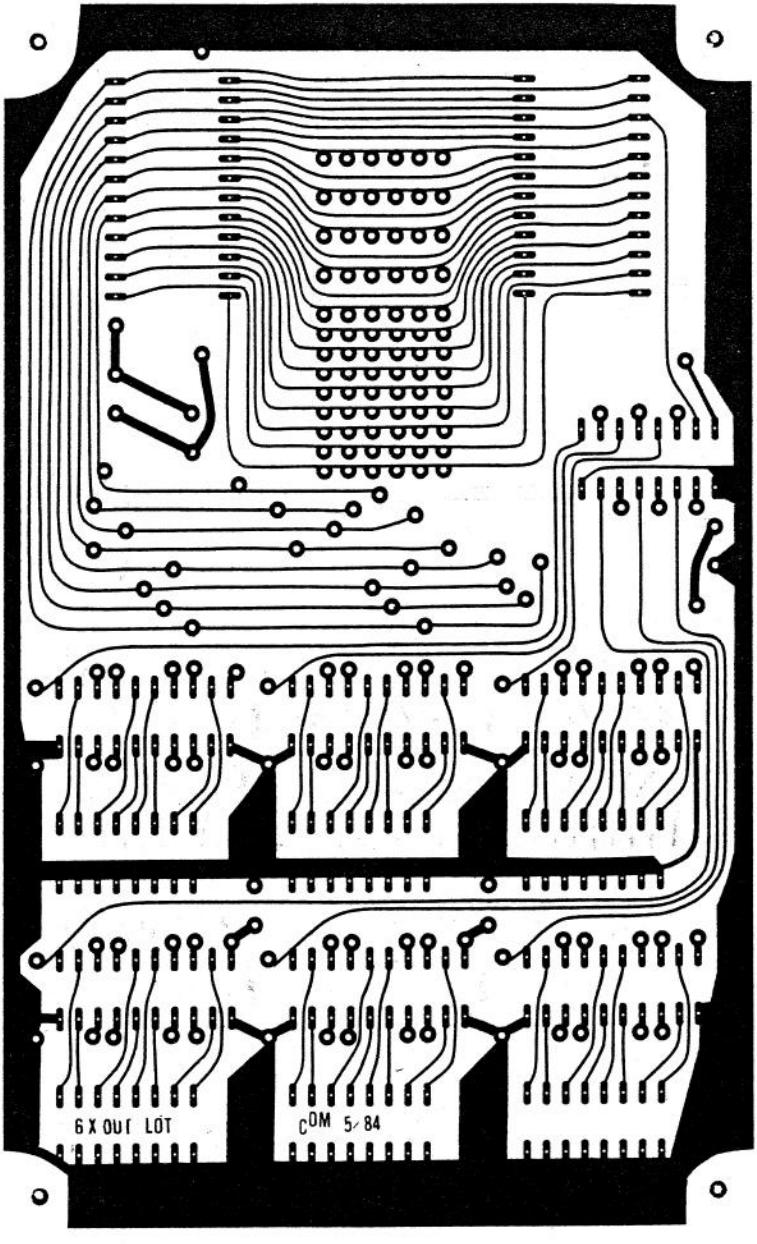


Abb. 3.3a Platinenlayout (System I), Lötseite

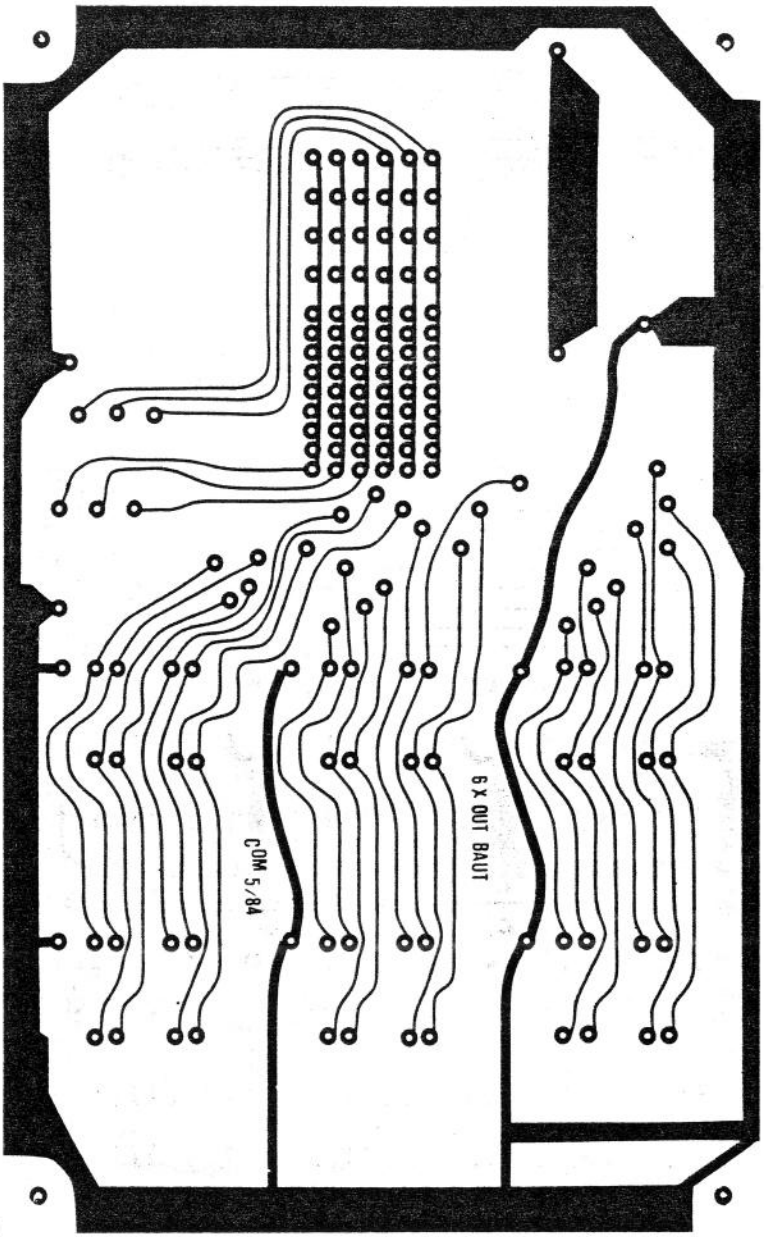


Abb. 3.3b Platinenlayout (System I), Bestückungsseite

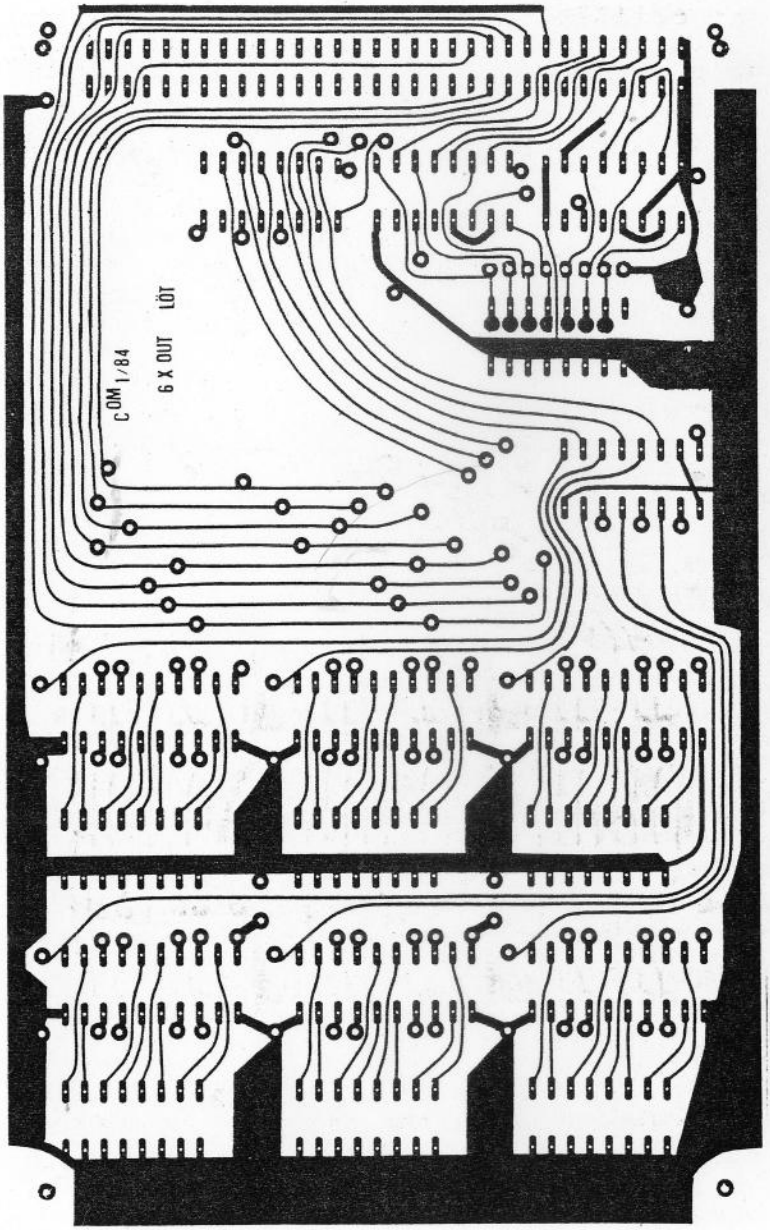


Abb. 3.4a Platinenlayout (System II), Lötseite

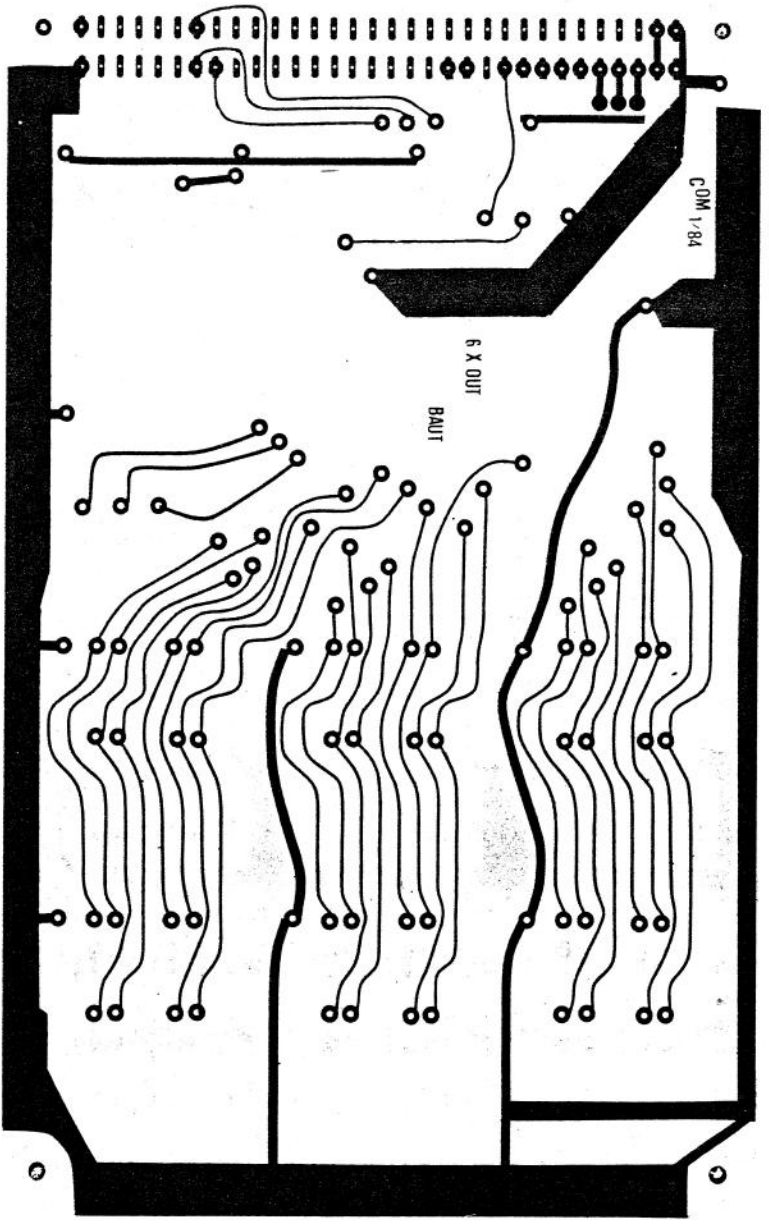


Abb. 3.4b Platinenlayout (System II), Bestückungsseite

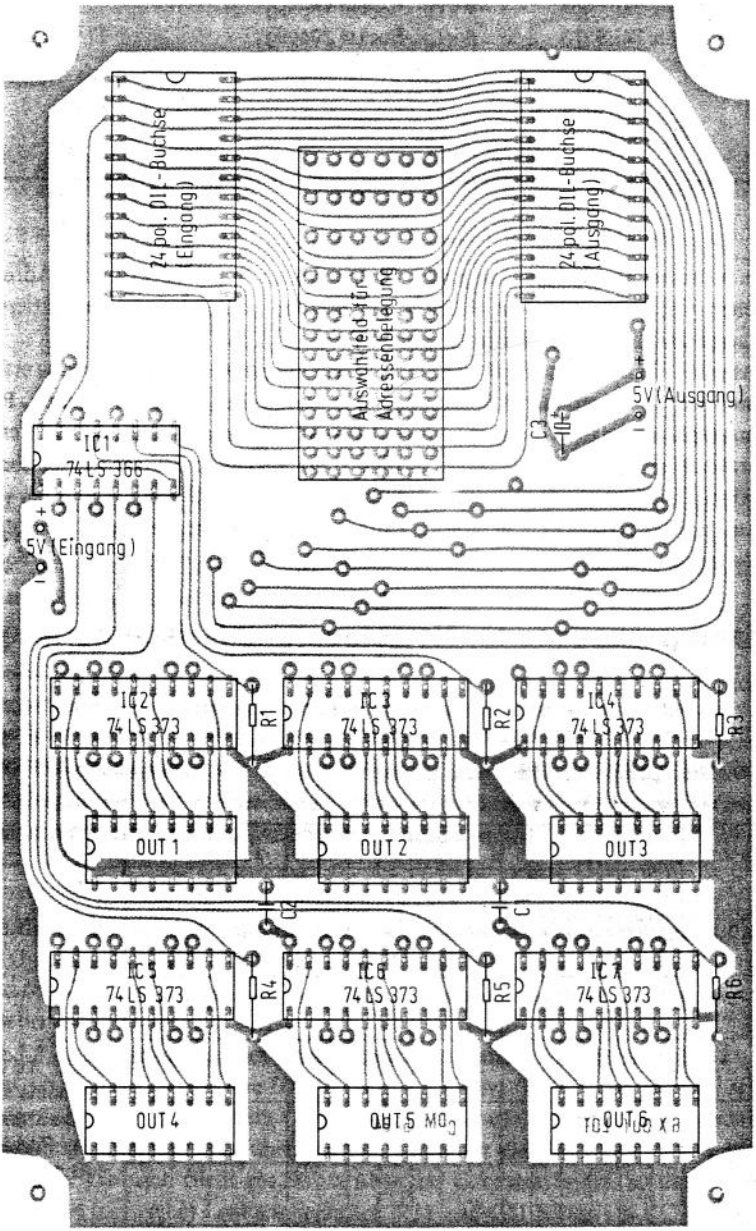


Abb. 3.5 Bestückungsplan (Sechsfach-OUT-Port-Karte)

Stückliste des Sechsfach-OUT-Ports

IC1	1 x 74LS366	7 x DIL-Sockel 16polig
IC2 ... 7	6 x 74LS373	6 x DIL-Sockel 20polig
		2 x DIL-Sockel 24polig
C1, C2	2 x 100 nF	4 x Steckstifte
C3	1 x 470 µF/10 V	
R1 ... 6	6 x 1 kΩ	

wünschten Adresse. Ein Beispiel mag dies verdeutlichen: Soll IC2 unter der Adresse 23559 angesprochen werden, so muß in Reihe 1 das sechste Lötauge durchkontaktiert werden. *Abb. 3.6* zeigt das hierzu gehörende Codierschema.

Beim System I kann der Anschluß der Stromversorgung über die Grundplatine erfolgen, allerdings muß der dort installierte Spannungsregler ausreichend gekühlt werden. Falls noch weitere Zusatzplatinen (z. B. Tongeneratoren) betrieben werden sollen, ist die Verwendung eines separaten Netzteils angezeigt. Beim System II erfolgt die Stromversorgung über den Erweiterungsbus.

Eine Funktionskontrolle der Schaltung kann erfolgen, indem der mit Leuchtdioden bestückte Teststecker verwendet wird, der bereits in Kapitel 2.2.4 besprochen wurde.

3.2 Leistungsausgang für Netzsteuerung (Triac-Interface)

3.2.1 Schaltung des Triac-Interface

Sollen größere Leistungen durch den Computer geschaltet werden, so kann an die einzelnen Bitausgänge einer Parallelschnittstelle (vergl. Kapitel 3.1) ein Relais angeschlossen werden.

Adresse / IC-Nr.	23 552	23 553	23 554	23 555	23 567	23 559	23 566	23 560	23 565	23 561	23 564	23 562	23 563	Adresse / Out-Nr.
2														1
3														2
4														3
7														6
6														5
5														4

Abb. 3.6 Codierschema für die Adreßzuordnung der Port-ICs (System I)

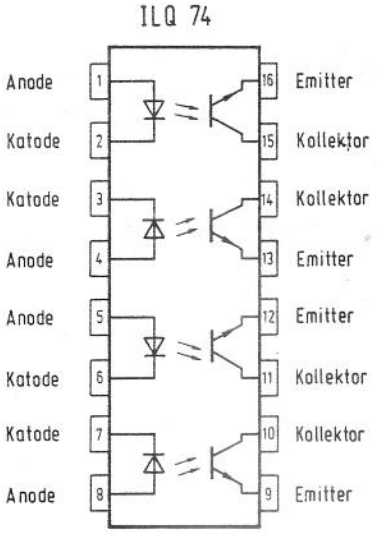


Abb. 3.7 4-fach-Optokoppler ILQ 74

Falls die TTL-Ausgänge nicht den nötigen Strom liefern können, muß noch ein Treibertransistor zwischengeschaltet werden. Insbesondere beim Schalten hoher Lasten unter Netzspannung ist allerdings ein Triac-Schalter wesentlich eleganter. Ein solcher Triac-Schalter wird ebenfalls an einen Ausgabe-Port angeschlossen.

Dabei muß aber eine Warnung ganz deutlich ausgesprochen werden: die hier vorgestellte Schaltung ist nicht für die ersten Gehversuche in Elektronik geeignet. Wenn beim Aufbau und bei der Inbetriebnahme nicht äußerste Vorsicht und Umsicht walten, besteht wegen des direkten Arbeitens an der Netzspannung Lebensgefahr! Daher darf sich nur der im sorgfältigen Arbeiten erfahrene Elektroniker an ihren Aufbau wagen.

Um die Entkopplung des Computers von der Netzspannung zu erreichen, ist die Verwendung von Optokopplern in den Datenleitungen unabdingbar. *Abb.3.7* zeigt das Anschlußbild des verwendeten Optokopplers ILQ-74. Er enthält vier Einzelsysteme, so daß insgesamt zwei dieser ICs benötigt werden.

In *Abb.3.8* ist das Schaltbild des Triac-Interface zu sehen. Die Eingänge IN 1 ... 8 korrespondieren mit den entsprechenden Daten-Ausgängen eines OUT-Ports. Der Widerstand R15 begrenzt den Strom für die LED im Optokoppler. Wird der Fototransistor beleuchtet (d. h. die entsprechende Datenleitung führt H-Pegel), so wird die Basis von T1 negativ und der Transistor leitet. Dadurch fließt ein Strom über das Gate des Triacs, so daß dieser zündet. Dadurch wird die (ohmsche) Last angeschaltet. R11 und C11 bilden ein RC-Glied, das einen zu schnellen Spannungsanstieg am Triac verhindern soll, um ein Selbstzünden zu vermeiden.

Um die Verlustleistung in der Ansteuerschaltung gering zu halten, wird sie nicht kontinuierlich mit der Stromversorgung verbunden, sondern jeweils zu Beginn einer Halbwelle kurzzeitig gepulst. Dies besorgt ein Timer-IC 555, das über T11, T10 getriggert wird. Die bei-

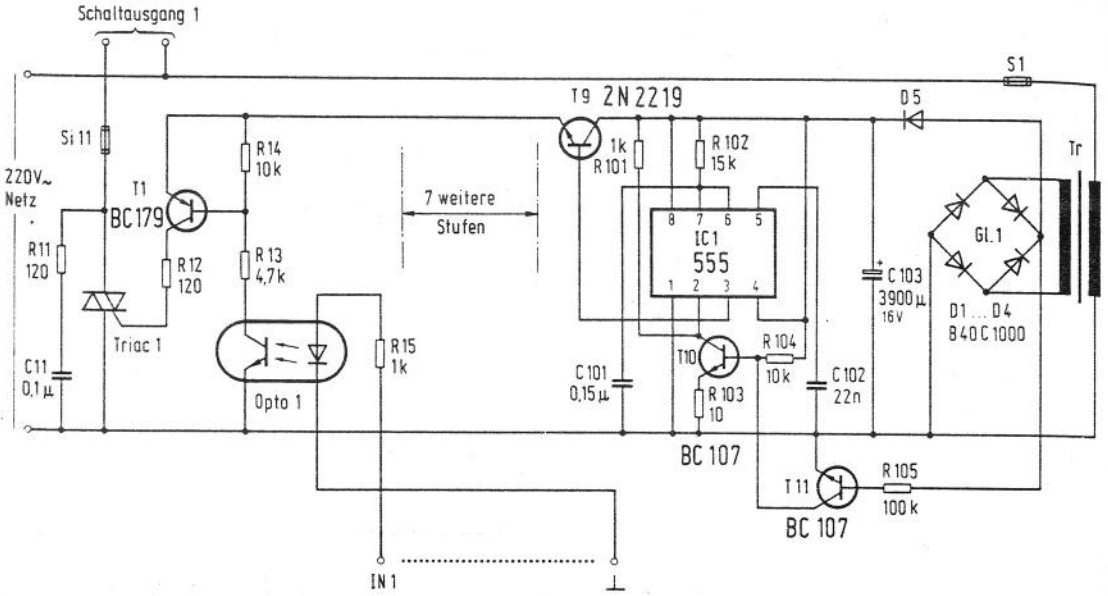


Abb. 3.8 Schaltbild des Triac-Interface

den Transistoren stellen einen Nulldurchgangsdetektor für die Wechselspannung dar, der seine Ansteuerung der Sekundärseite des Trafos Tr entnimmt. Der Ausgangspuls (Zeitdauer ist durch R102 und C101 gegeben) steuert den Längstransistor T9 an, der die Betriebsspannung an die Triac-Ansteuerung anlegt.

Nach Datenblatt verträgt der verwendete Triac maximal 600V und 6A. Allerdings muß bei voller Belastung eine Kühlung des Triacs vorgesehen werden. Aber Vorsicht, der Kühlbolzen ist elektrisch mit dem System verbunden! Es ist also eine sorgfältige Isolierung erforderlich, sonst entsteht ein Kurzschluß. Die eingesetzte Sicherung ist je nach Kühlung zu dimensionieren. Sinnvollerweise sollte ein maximaler Strom von 2A nicht überschritten werden. In diesem Zusammenhang eine Warnung: die übliche „Fingerprüfmethode", ob die Kühlung ausreichend ist und das Bauteil auch nicht zu warm wird, ist natürlich lebensgefährlich.

3.2.2 Aufbau und Inbetriebnahme

Das Triac-Interface wird auf einer einseitig kupferkaschierten Platine aufgebaut. In *Abb.3.9* und *Abb.3.10* sind das Platinenlayout und der Bestückungsplan mit Stückliste zu sehen.

Beim Aufbau ist mit äußerster Sorgfalt vorzugehen, da die Platine im Betrieb direkt mit Netzspannung verbunden ist. Dazu gehört eine besonders gründliche Prüfung auf unerwünschte Verbindungen zwischen Leiterbahnen, die beim Ätzen der Platine unter Umständen entstanden sind. Insbesondere gilt dies für die Ein- und Ausgangsseite der Optokoppler.

Beim Kauf des Transformators muß darauf geachtet werden, ob seine Anschlüsse mit dem Platinenlayout übereinstimmen. Wenn dies nicht der Fall ist, muß die Platine durch Unterbrechen von Leiterbahnen bzw. zusätzliche Verdrahtung mit isoliertem Schaltdraht entsprechend abgeändert werden. Dabei muß mit besonderer Sorgfalt vorgegangen werden, siehe oben.

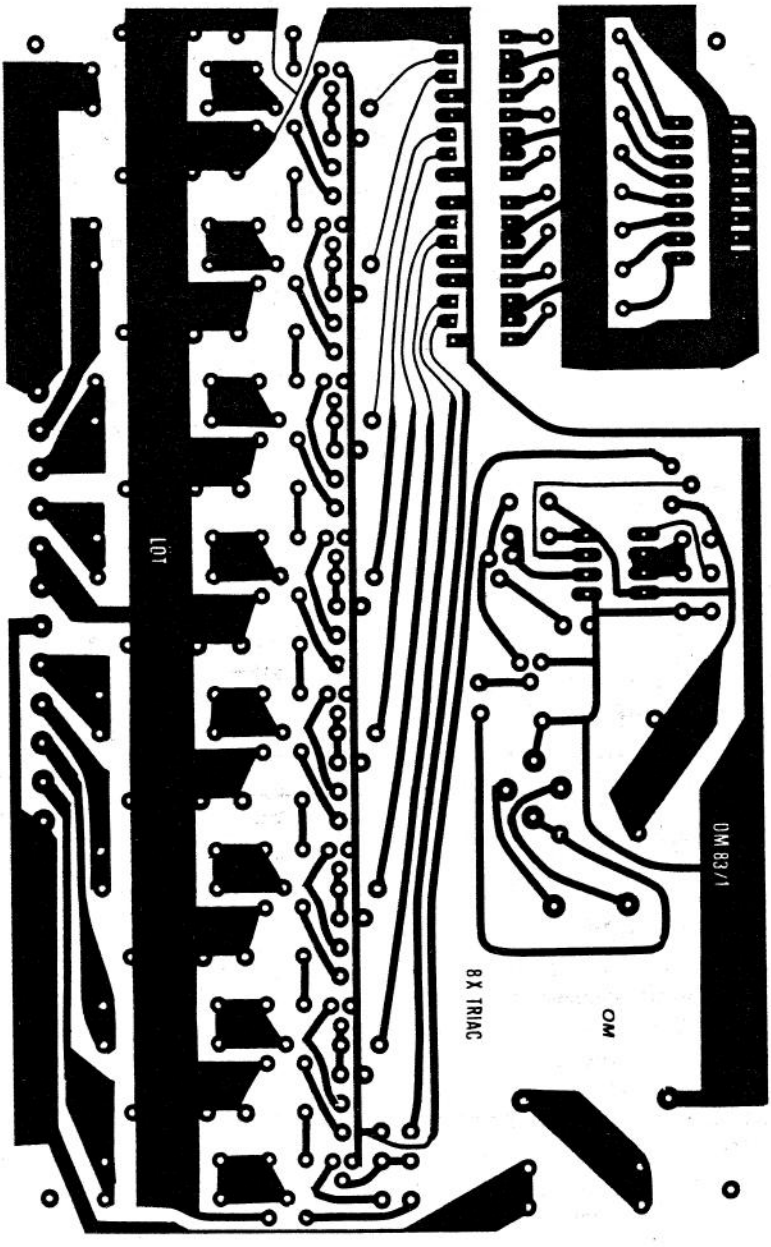


Abb. 3.9 Platinenlayout zum Triac-Interface, Lötseite

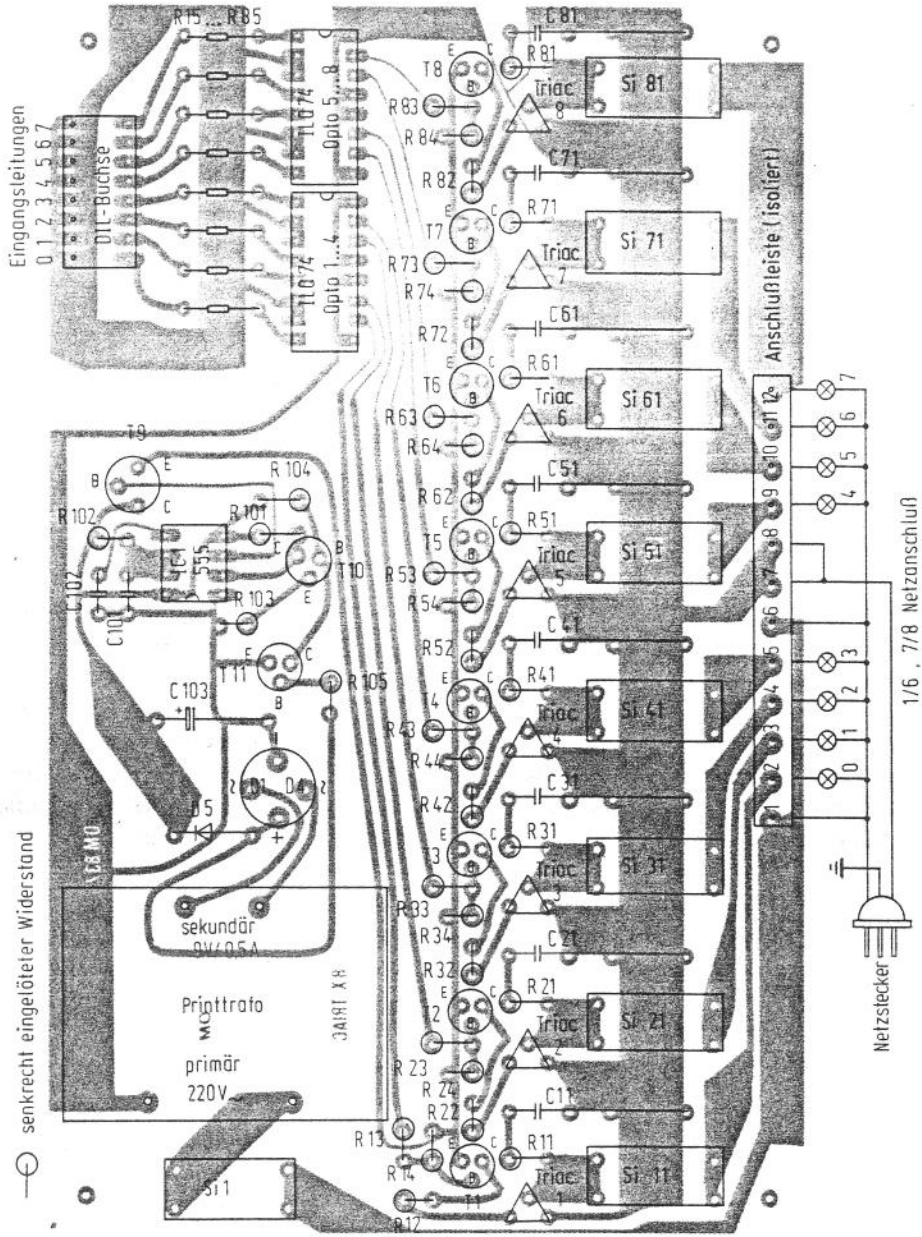


Abb. 3.10 Bestückungsplan (Triac-Interface)

Stückliste der Leistungsplatine (Triac-Netzausgang, achtfach)

R11 ... R81	8 × 120 Ω	R101	1 × 1 kΩ
R12 ... R82	8 × 120 Ω	R102	1 × 15 kΩ
R13 ... R83	8 × 4,7 kΩ	R103	1 × 10 Ω
R14 ... R84	8 × 10 kΩ	R104	1 × 10 kΩ
R15 ... R85	8 × 1 kΩ	R105	1 × 100 kΩ

C11 ... C81	8 × 0,1 µF (unbedingt auf Netzspannungsfestigkeit achten, nur sogenannte Störschutzkondensatoren verwenden, Rastermaß 22,5 mm)
C101	1 × 0,15 µF
C102	1 × 22 nF
C103	1 × 3900 µF/16 V (Elko)
Di1 ... 4	1 × B40/C1000 (Brückengleichrichter)
Di5	1 × 1N4007
T1 ... 8	8 × BC 179
T9	1 × 2N2219
T10, 11	2 × BC107
IC1	1 × NE 555
Opto1 ... 8	2 × ILQ-74 (Vierfach-Optokoppler)
Triac1 ... 8	8 × TIC 216M
Si1	1 × Si 0,1A mit Sicherungshalter (RM 15 mm)
Si11 ... 81	8 × Si (Wert je nach Kühlung der Triacs, bis 2A) mit Sicherungshalter (RM 15 mm)
	3 DIL-Fassungen 16polig
	1 DIL-Fassung 8polig
	1 Anschlußleiste 12polig, RM 5 mm
	1 Printtrafo 9V/0,5 A

Zur ersten Inbetriebnahme muß die Platine über einen Trenntransformator angeschlossen werden. Dadurch ist sie erdfrei, so daß bei eventuellen Messungen mit einem Oszilloskop keine Kurzschlüsse auftreten können. Vor dem Einschalten werden die zu schaltenden Glühbirnen in der gezeigten Weise (Abb. 3.10) angeschlossen. Anstatt sofort den ZX 81 zur Steuerung zu verwenden, kann eine 4,5-V-Batterie mit der entsprechenden Datenleitung an der Eingangsbuchse verbunden werden. Ein H-Signal (bzw. Batterie angeschlossen) veranlaßt die Glühbirne des entsprechenden Kanals aufzuleuchten (wegen der Leuchtdiode richtige Polung beachten). Selbstverständlich ist, daß bei allen Eingriffen an der Platine die Stromversorgung abgeschaltet wird.

Zeigen die Tests, daß die Platine einwandfrei arbeitet, kann ein OUT-Port an den Eingang des Triac-Bausteins angeschlossen werden. Beim Einstecken der Flachkabelverbindung wie üblich darauf achten, daß die Stecker in der richtigen Weise eingesetzt werden, sonst entsteht ein Kurzschluß. Mit der Befehlsfolge

POKE Adresse, Zahl

leuchten die Glühbirnen nun entsprechend dem Binärwert der eingegebenen Zahl auf. Als Adresse wird die Adresse des Ports eingesetzt, an dem das Triac-Interface angeschlossen ist.

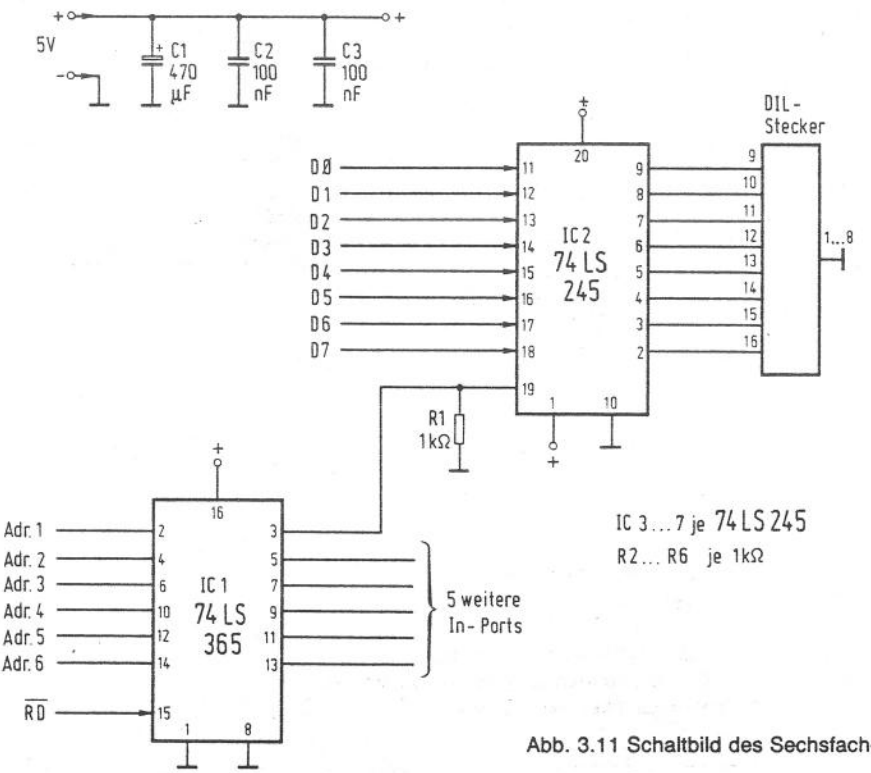


Abb. 3.11 Schaltbild des Sechsfach-IN-Ports

Sind alle Überprüfungen erfolgreich abgeschlossen, muß diese Platine unbedingt in ein geeignetes Gehäuse eingebaut werden, so daß ein versehentliches Berühren spannungsführender Leitungen und Bauteile vermieden wird. Die einschlägigen VDE-Bestimmungen sollten im eigenen Interesse beachtet werden.

3.3 Sechsfach IN-Port

3.3.1 Schaltung der IN-Port-Karte

Die IN-Port-Karte *(Abb.3.11* zeigt die Version für System I) ist ganz entsprechend zur OUT-Port-Karte (Kapitel 3.1.) aufgebaut. Wie bei dem Einzelport auf der Grundplatine I wird auch hier der Bustreiber 74LS245 als Port-IC verwendet. Da dieses IC zum Aktivieren ein L-Signal an Pin 19 benötigt, muß im Unterschied zur Schaltung in Kapitel 3.1 das IC 74LS365 zur Ansteuerung verwendet werden (nichtinvertierender Bustreiber, Anschlußbild siehe *Abb.3.12*).

Wenn ein Lesebefehl auf eine der gewünschten Adressen vorliegt, wird über die $\overline{RD}$-Leitung IC1 aktiviert. Dabei gelangt ein L-Signal an Pin 19 des gewünschten Port-ICs (IC2 ... 7). Dadurch werden die an den Eingängen (Pin 2 ... 9) anliegenden Daten vom Computer übernommen. Es erfolgt also keine Zwischenspeicherung der Eingangsdaten, sie müssen in dem Augenblick anliegen, wenn der Port angesprochen wird. In vielen Fällen ist dies

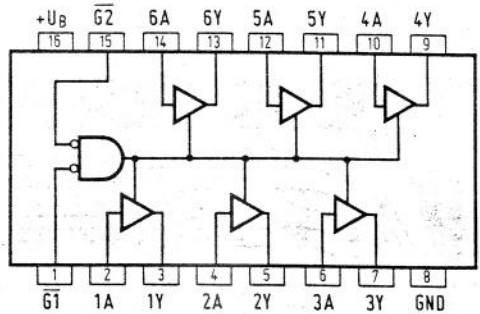


Abb. 3.12 Anschlußbild des Bustreibers 74LS365

kein Nachteil. Wird eine Speicherung der Eingangsdaten gewünscht, so kann die in Kapitel 3.4 beschriebene Karte verwendet werden. Sie enthält drei IN-Ports, wobei ein bitweises Setzen und Rücksetzen der Datenleitungen möglich ist.

Die Widerstände R1 ... 6 sorgen dafür, daß an Pin 19 von IC2 L-Pegel anliegt, solange die Tristate-Ausgänge von IC1 hochohmig sind. Bei der entsprechenden Schaltung für System II entfallen IC1 und R1 ... 6.

Die Adreßauswahl für System I erfolgt in der beschriebenen Weise über Lötbrücken (vergl. Kapitel 3.1.2).

3.3.2 Aufbau und Inbetriebnahme

Für diese Schaltung gelten dieselben Hinweise wie für den OUT-Port in Kapitel 3.1.2. Dies betrifft insbesondere die Auswahl der Adressen. OUT-Port und IN-Port können auf den gleichen Adressen betrieben werden, da sie sich gegenseitig nicht stören. Das Layout der zweiseitigen Platine zeigt *Abb. 3.13* (System I) bzw. *Abb. 3.14* (System II), in *Abb. 3.15* ist der Bestückungsplan und die Stückliste (für System I) aufgeführt.

Wenn die Platine aufgebaut und auf Leiterbahnunterbrechungen bzw. Kurzschlüsse überprüft ist, kann ein Funktionstest erfolgen, indem in die interessierende Eingangsbuchse ein 8poliger DIL-Schalter eingesetzt wird.

PRINT PEEK Adresse

liefert dann den Dezimalwert der eingestellten Binärkombination.

3.4 S/R-IN-Port

3.4.1 Schaltungsgrundlagen

Es gibt eine Reihe von Anwendungsfällen, wo Daten nur kurzzeitig von außen geliefert werden. Wird die Information nicht in genau diesem Augenblick vom Computer abgerufen, so ist sie bei dem in Kapitel 3.3 beschriebenen Port verloren.

Ein Beispiel soll erläutern, was gemeint ist. Um einen Raum zu überwachen, sind verschiedene Kontakte (z. B. Lichtschranken) installiert. Betritt eine Person den Raum, wird ein kur-

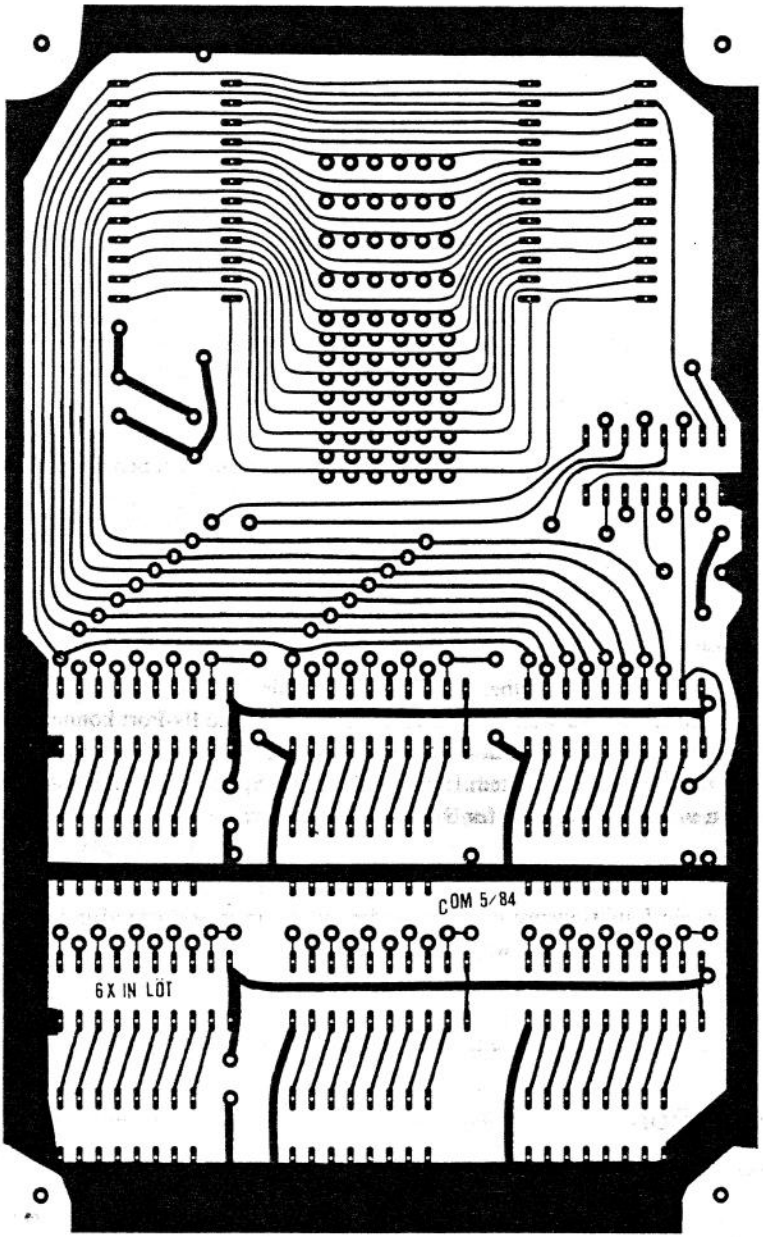


Abb. 3.13a Platinenlayout des Sechsfach-IN-Ports (System I), Lötseite

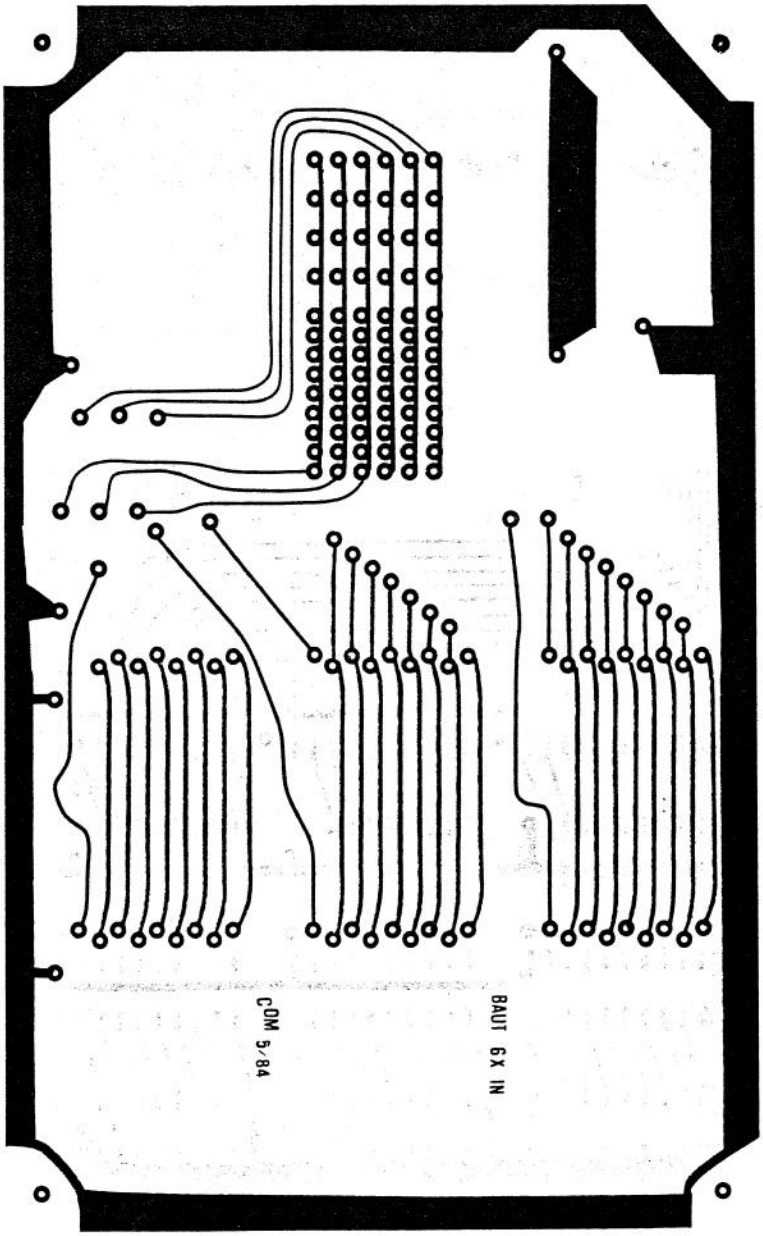


Abb. 3.13b Platinenlayout des Sechsfach-IN-Ports (System I), Bestückungsseite

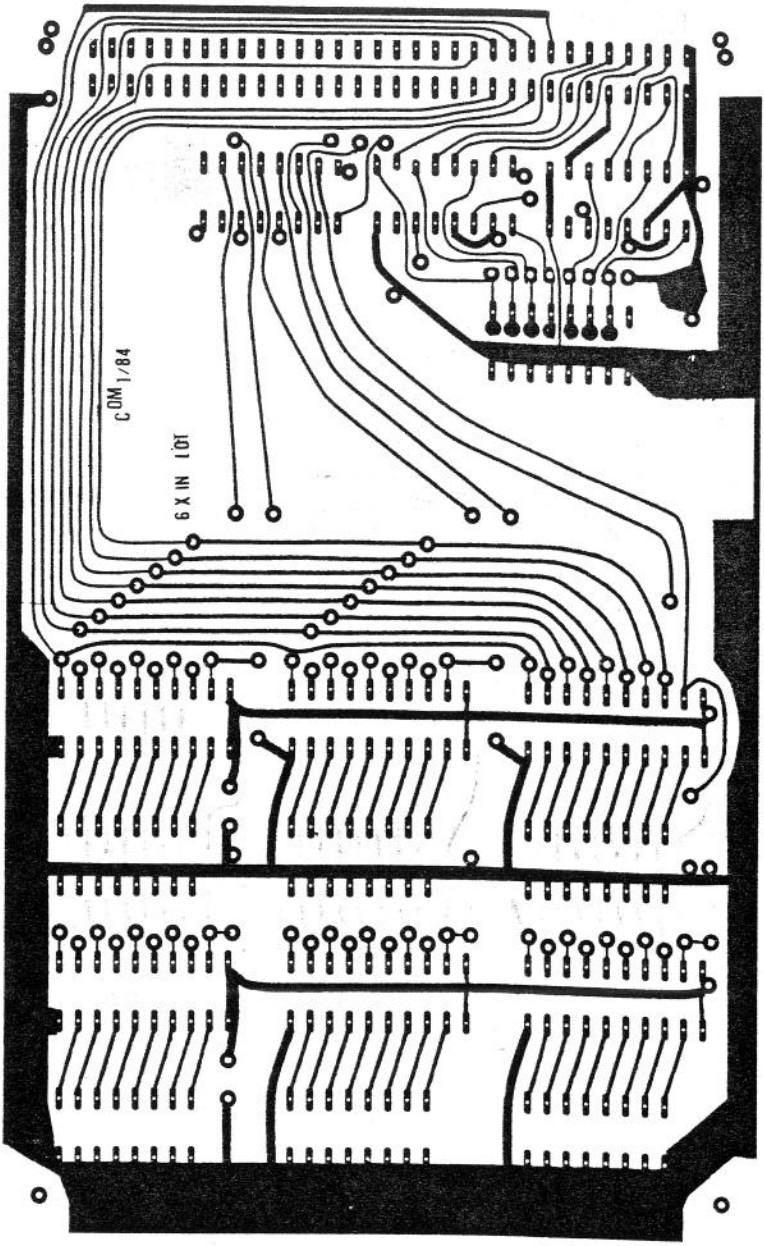


Abb. 3.14a Platinenlayout des Sechsfach-IN-Ports (System II), Lötseite

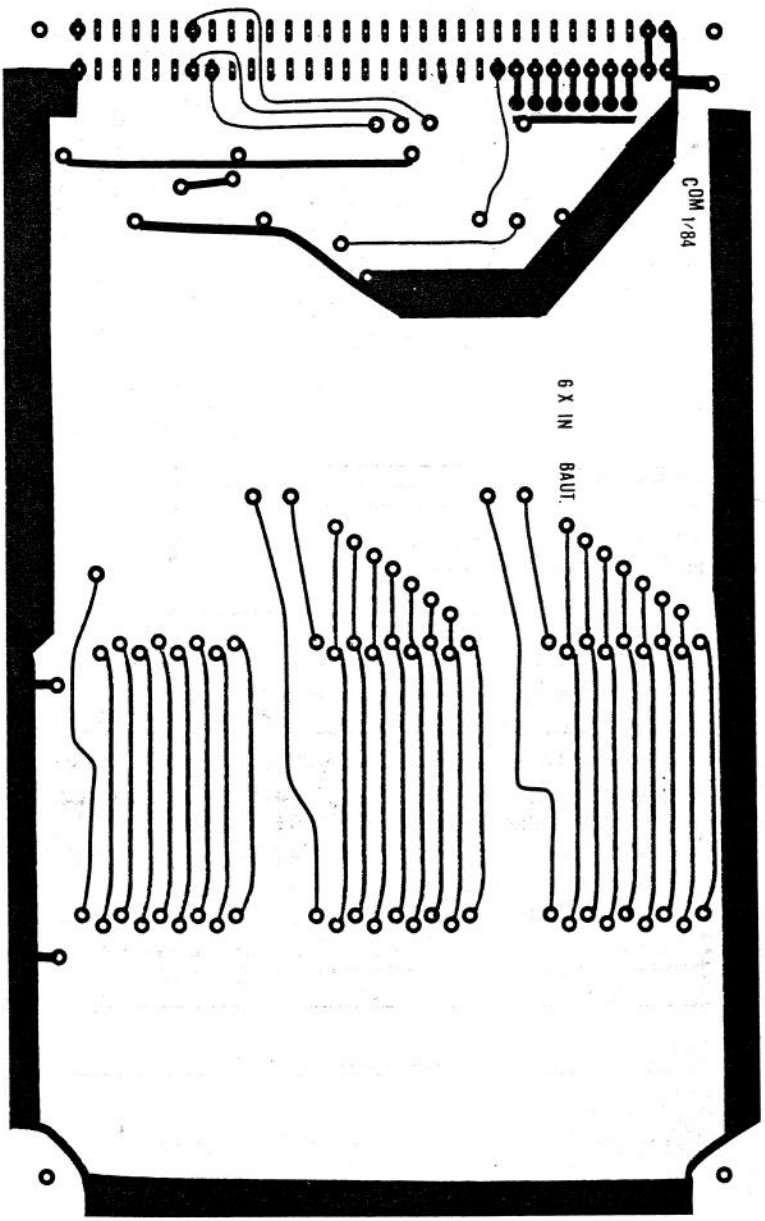


Abb. 3.14b Platinenlayout des Sechsfach-IN-Ports (System II), Bestückungsseite

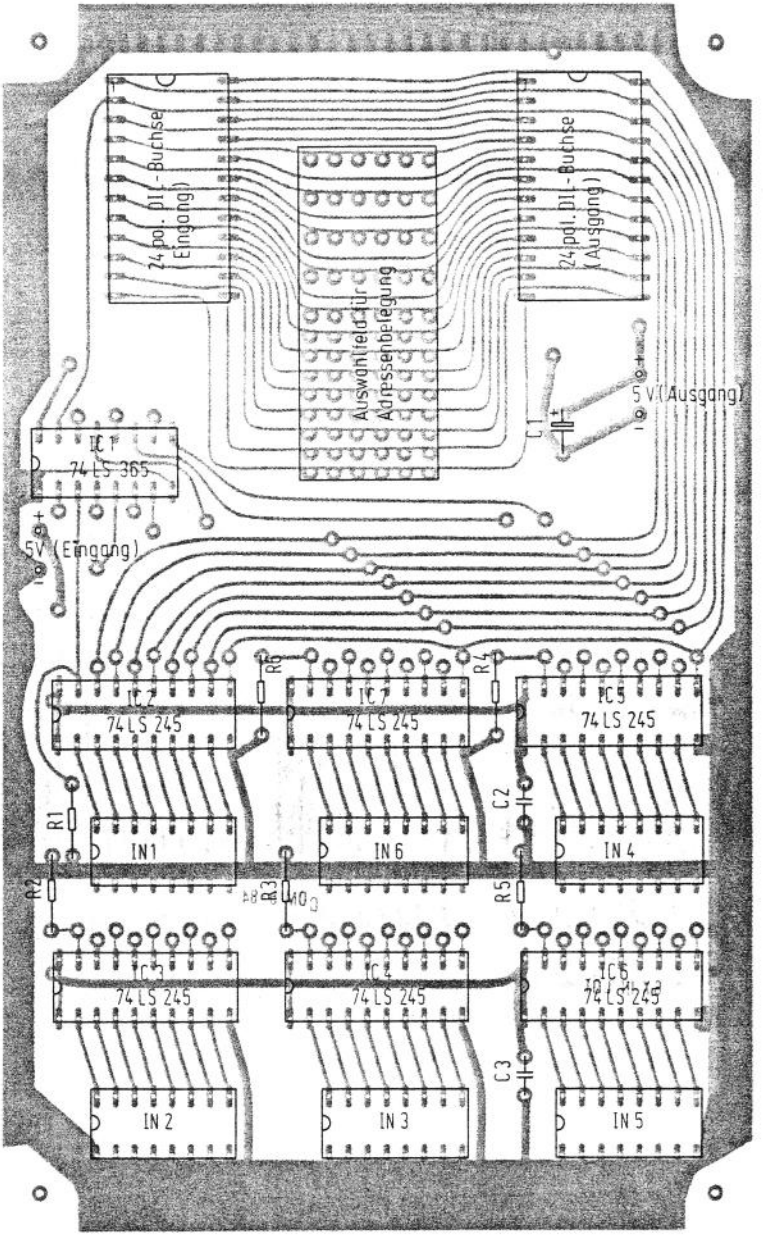


Abb. 3.15 Bestückungsplan (Sechsfach-IN-Port-Karte)

Stückliste Sechsfach-IN-Port

IC1	1 x LS365	6 x IC-Fass. 20polig
IC2 ... IC7	6 x LS245	7 x IC-Fass. 16polig
C1	1 x 470 µF	2 x IC-Fass. 24polig
C2 ... C3	2 x 100 nF	Kleinmaterial
R1 ... R6	6 x 1 kΩ	

zes (Set-)Signal ausgelöst. Verläßt die Person den Raum, so wird ein Rücksetz-Signal (Reset) erzeugt. Ein ähnliches Problem liegt auch in einer Modelleisenbahnanlage vor, wenn ein Zug Kontaktgleise überfährt und dabei einzelne Gleisabschnitte als belegt bzw. frei meldet.

Die Schaltung verwendet acht ICs vom Typ 4044 (*Abb.3.16*). Dieser Baustein enthält jeweils 4 S/R-Flipflops, deren Ausgänge Q Tristate-Verhalten zeigen. Das bedeutet, liegt am CS-Eingang ein H-Signal, so werden die vier Flipflop-Ausgänge durchgeschaltet, andernfalls sind sie hochohmig.

Damit läßt sich das IC direkt am Datenbus betreiben. *Abb. 3.17* zeigt das Gesamtschaltbild. Über IC1 erfolgt die Verknüpfung des $\overline{RD}$-Signals mit dem entsprechenden Adreßsignal. Der Typ 74LS366 invertiert die Eingangssignale, so daß das CS-Signal für den 4044 wie benötigt H-aktiv ist (vergl. Abb. 3.2). R1 ...R3 sorgen dafür, daß bei hochohmigen Ausgängen eindeutig L-Pegel für die Port-ICs vorliegt. Die Eingangsleitungen der S/R-Flipflops werden über jeweils 15kΩ an + 5V gelegt. Setzen bzw. Rücksetzen erfolgt durch einen kurzzeitigen L-Impuls auf der entsprechenden Eingangsleitung. Die Eingänge sind über 16polige DIL-Buchsen zugänglich.

3.4.2 Aufbau und Inbetriebnahme

Abb.3.18 (System I) bzw. *Abb.3.19* (System II) zeigen das Platinen-Layout. In *Abb.3.20* ist der Bestückungsplan mit der Stückliste aufgeführt. Der Schaltungsaufbau erfolgt nach den bereits besprochenen Gesichtspunkten (z. B. Adreßauswahl, Platinenkontrolle usw.). Die Pull-up-Widerstände an den Flipflop-Eingängen werden stehend eingesetzt. Ihre obere Seite wird gemeinsam verlötet und zum Plus-Anschluß geführt (liegt jeweils in der Nähe von Pin 1 der ICs). Falls erhältlich, lassen sich an dieser Stelle natürlich mit Vorteil sogenannte SIL-Widerstände (Single Inline) verwenden.

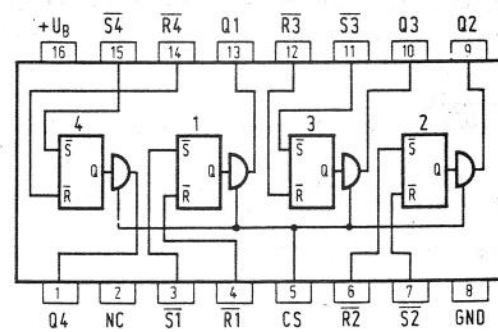


Abb. 3.16 Vierfach SR-Flipflop 4044 mit Tristate-Ausgang

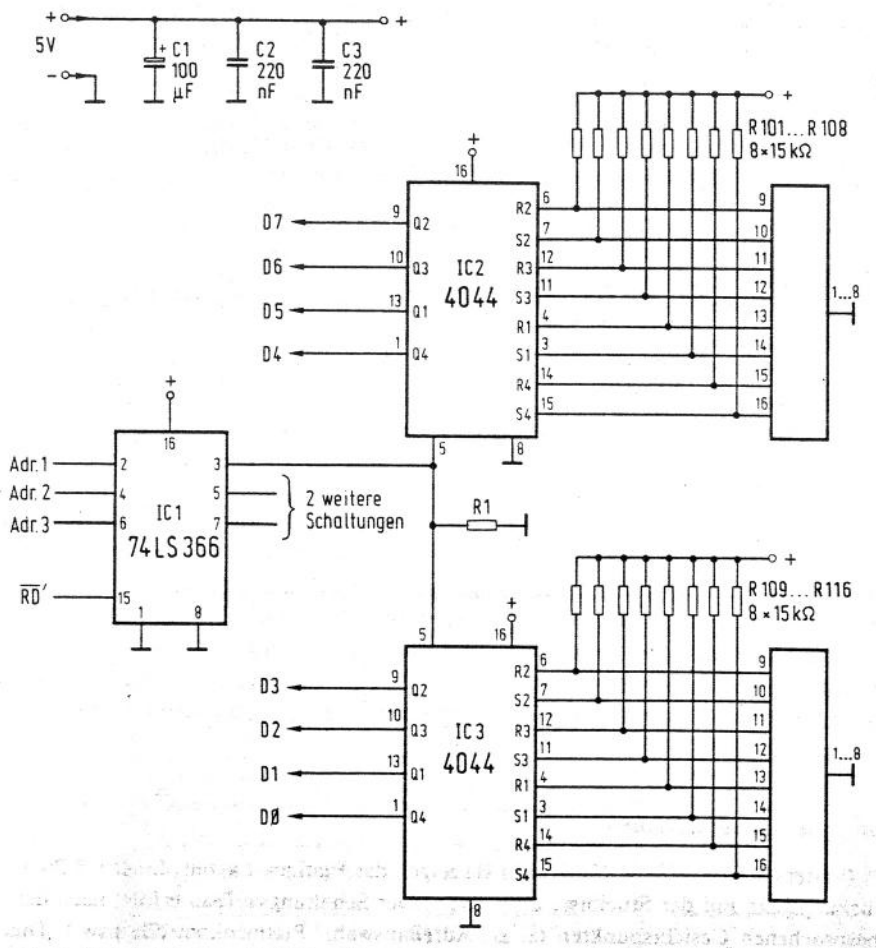


Abb. 3.17 Schaltbild der SR-IN-Port-Karte

Beim Einschalten ist der Zustand der Flipflops zufällig. Wenn das stört, kann durch verschiedene Werte der Pull-up-Widerstände für die Set- und Reset-Eingänge Abhilfe geschaffen werden. Im anderen Fall muß die Schaltung von außen in einen definierten Startzustand gebracht werden.

Getestet wird der Schaltungsaufbau, indem ein kurzes Programm eingegeben wird:

```
100 PRINT PEEK Adresse;" ";
200 GOTO 100
```

Während das Programm läuft, verbindet man die entsprechenden Set- bzw. Reset-Eingänge kurzzeitig mit Masse. Dabei muß sich eine entsprechende Veränderung der angezeigten Zahl ergeben.

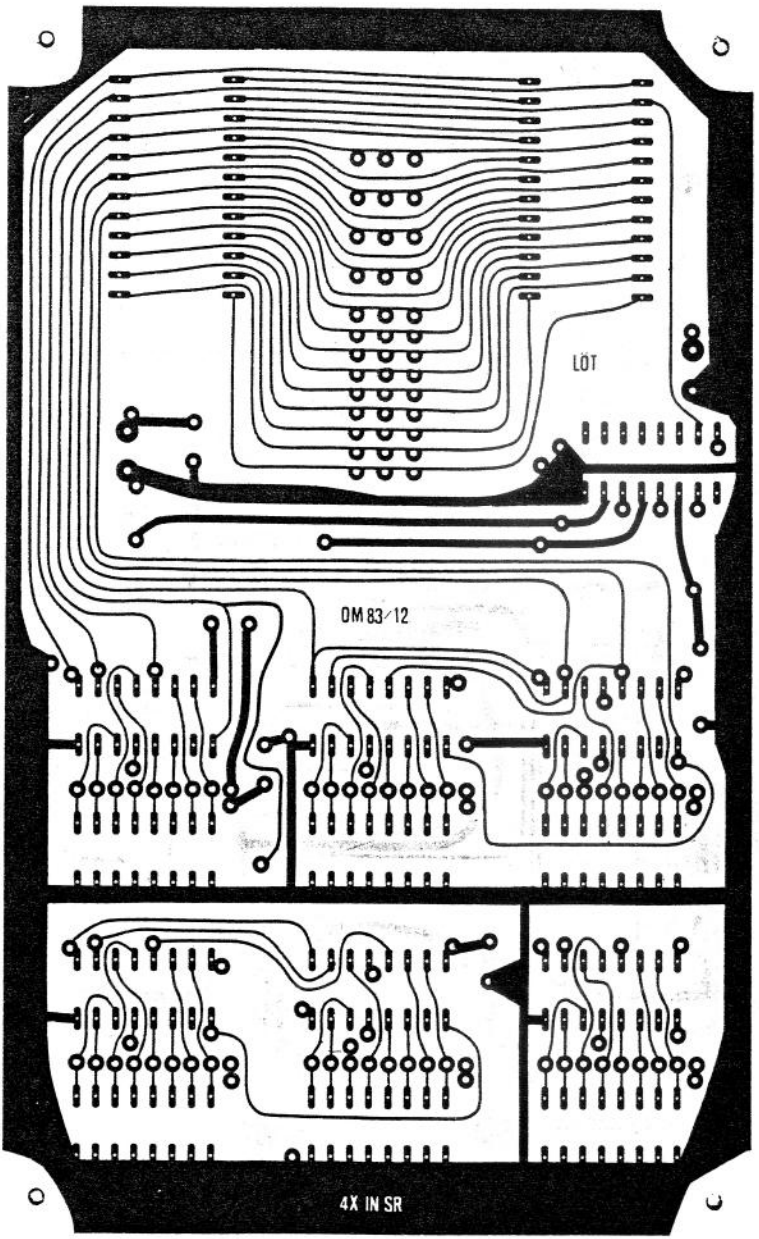


Abb. 3.18a Platinenlayout des SR-IN-Ports (System I), Lötseite

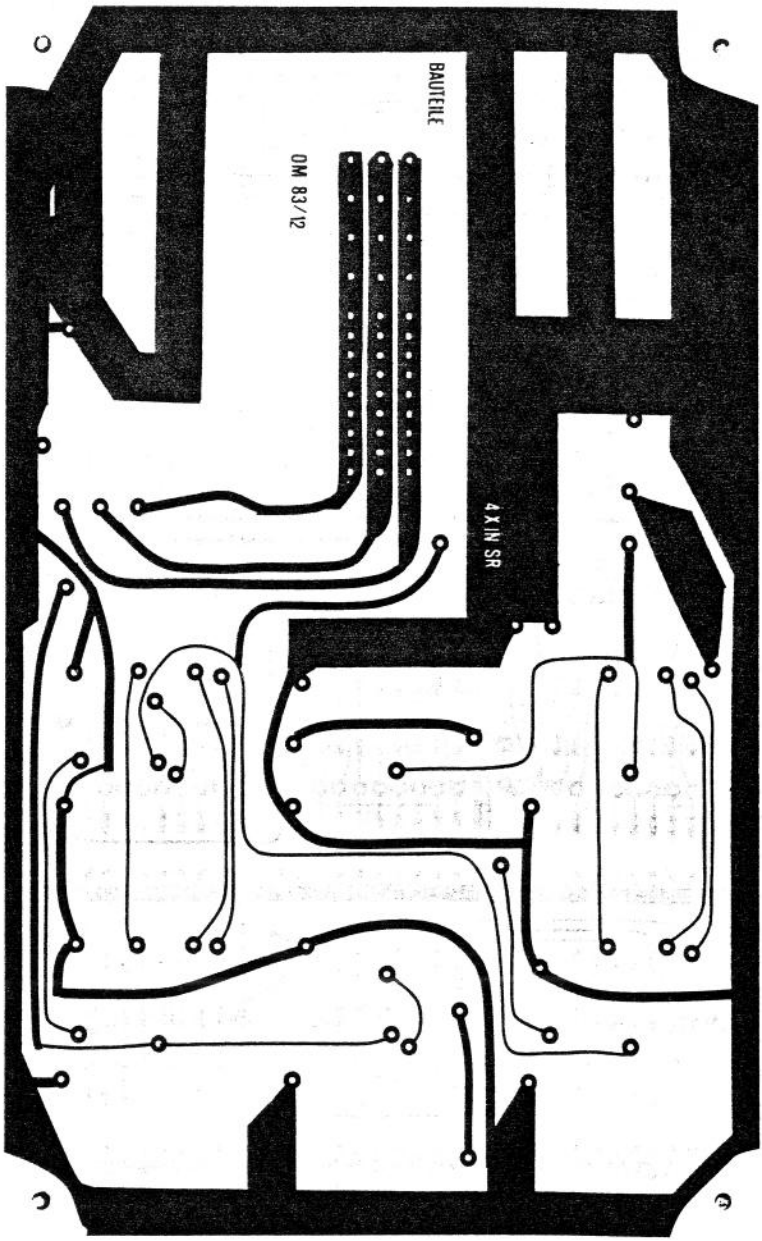


Abb. 3.18b Platinenlayout des SR-IN-Ports (System I), Bestückungsseite

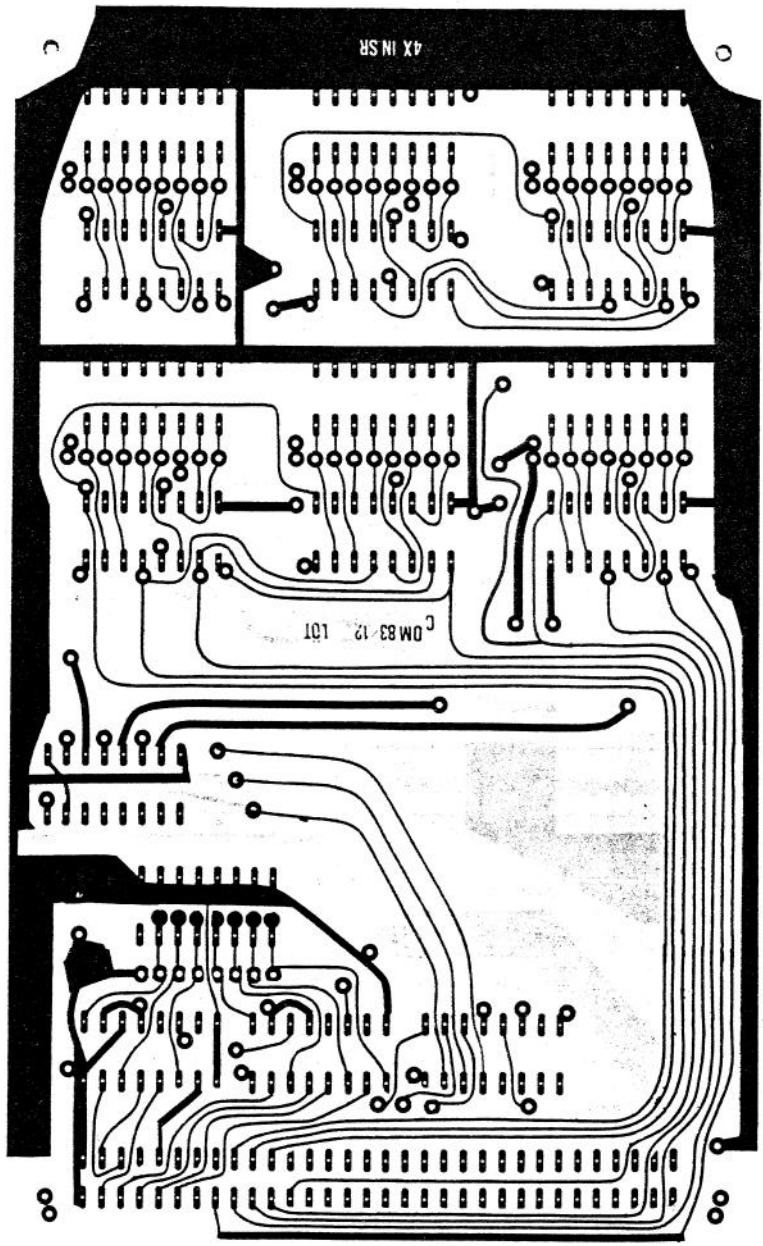


Abb. 3.19a Platinenlayout des SR-IN-Ports (System II), Lötseite

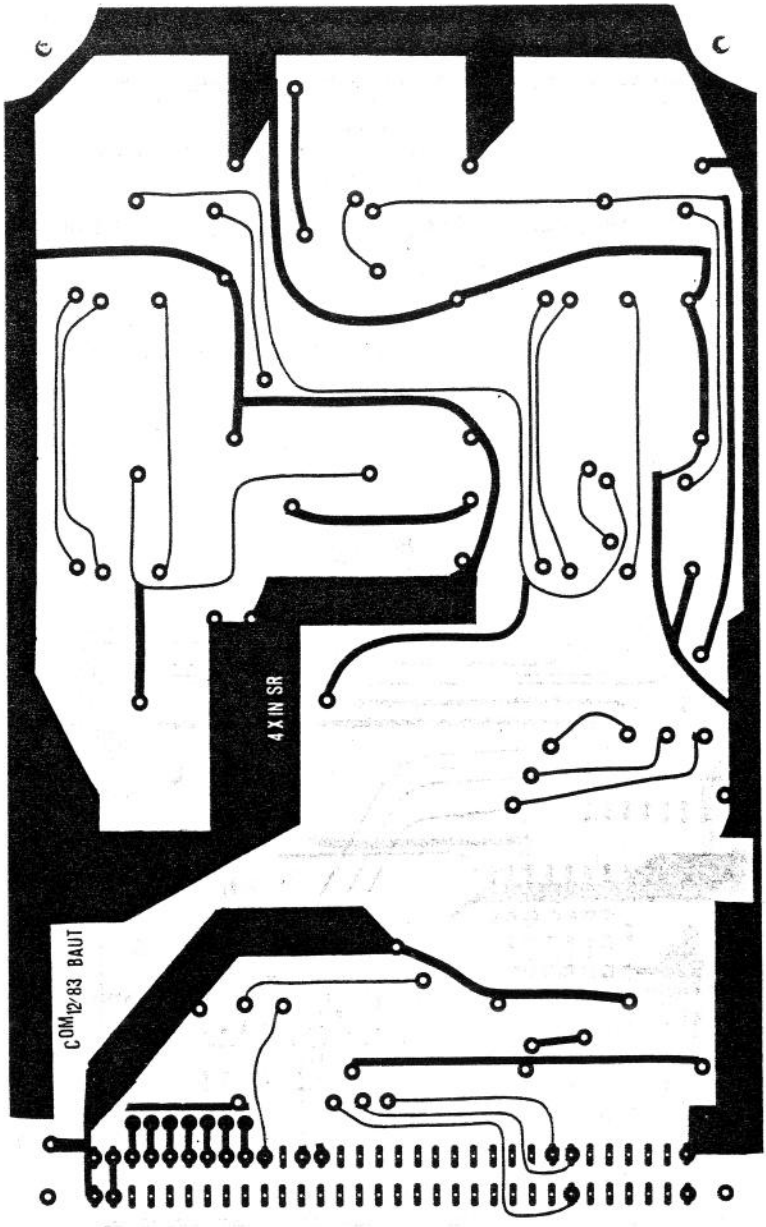
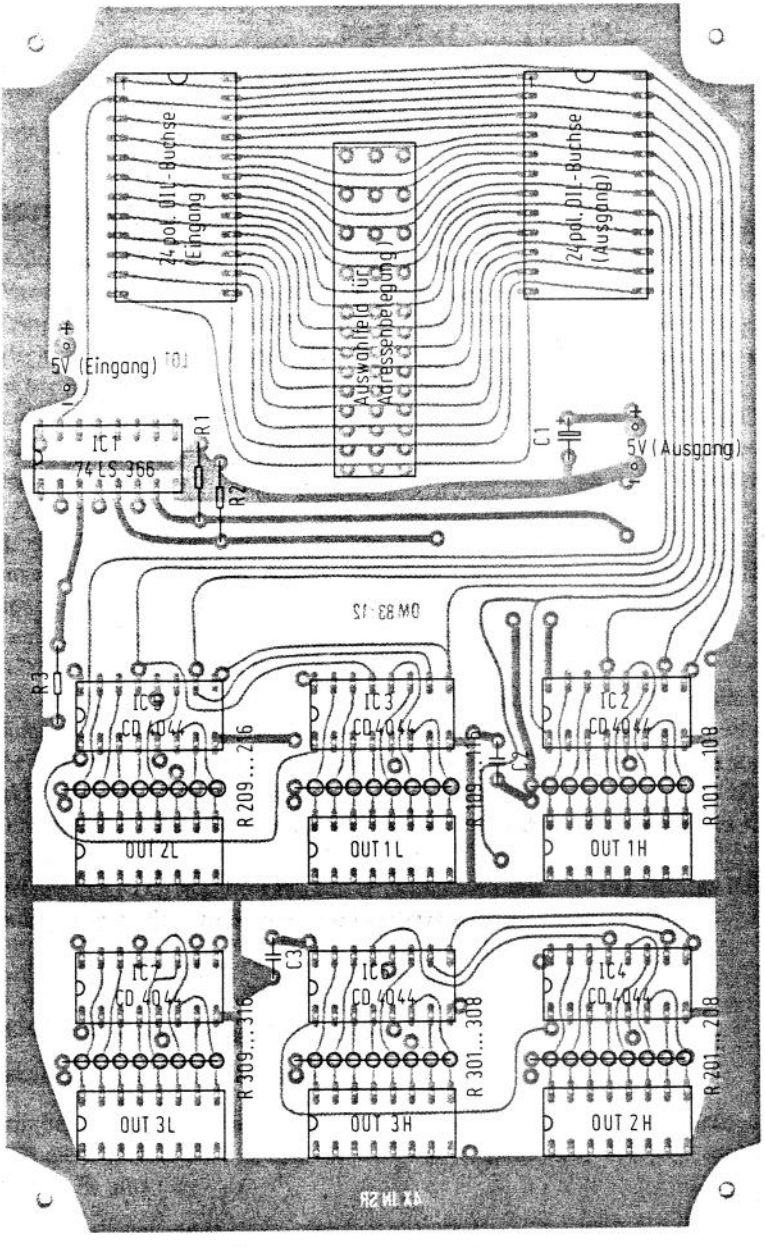


Abb. 3.19b Platinenlayout des SR-IN-Ports (System II), Bestückungsseite

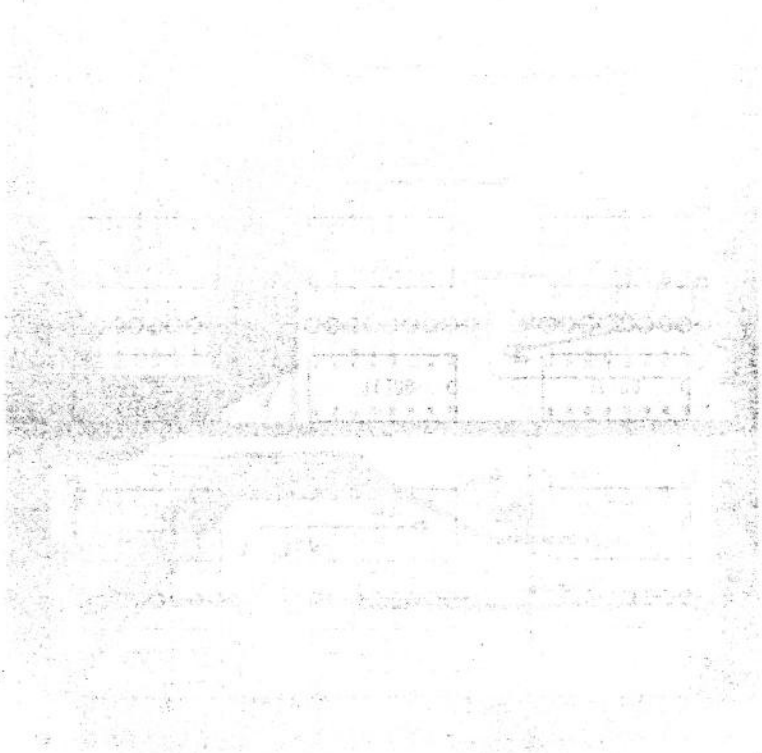


R 101... / 201... / 301... werden stehend eingelötet und oben miteinander verlötet. Falls erhältlich, kann auch ein SIL-Widerstandsnetzwerk verwendet werden.

Abb. 3.20 Bestückungsplan (SR-IN-Port-Karte)

Stückliste Vierfach-IN-Port (S/R)

IC1	1 x 74LS366
IC2 ... 7	6 x 4044
C1	1 x 100 µF
C2 ... 3	2 x 220 nF
R1 ... R3	3 x 1 kΩ (entfallen bei System II)
R101 ... R116	
R201 ... R216	
R301 ... R316	48 x 15 kΩ
	13 x IC-Fass. 16polig
	2 x IC-Fass. 24polig
	Steckstift



4 Analoge Ein/Ausgabe-Karten

4.1 Übersicht

In der Meßtechnik ist auch ein Messen nichtelektrischer Größen möglich, wenn entsprechende Wandler eingesetzt werden. Solche Wandler gibt es für fast alle Meßprobleme, z. B. Druck, Temperatur, Geschwindigkeit, Helligkeit und vieles mehr. An ihrem Ausgang steht ein elektrisches Signal zur Verfügung, das in einem bestimmten Zusammenhang mit der eigentlichen Meßgröße steht (idealerweise ein proportionaler Zusammenhang).

Will man einen Computer direkt für Meßaufgaben einsetzen, so müssen die analogen elektrischen Signale am Ausgang des jeweiligen Wandlers in digitale Form umgesetzt werden (analog bedeutet, daß diese Signale kontinuierlich veränderlich sind, d. h. keine Sprünge aufweisen). Zu diesem Zweck benötigt man einen Analog-Digital-Wandler (A/D-Wandler, Kap. 4.4).

Soll umgekehrt der Computer gewisse elektrische Signalgrößen vorgeben (beispielsweise eine Vorspannung für den Arbeitspunkt eines Transistors), so muß das digitale Computersignal in eine analoge Größe übersetzt werden. Diese Aufgabe erfüllt ein Digital-Analog-Wandler (D/A-Wandler, Kap.4.3).

Da in vielen Fällen der Meßwandler den Charakter eines ohmschen Widerstands besitzt, läßt sich auch eine direkte Wandlung eines Widerstandswertes in einen Digitalwert durchführen. Solche R/D-Wandler werden auch gerne als Paddles („Steuerknüppel") in sogenannten „Joy-Sticks" für Computerspiele eingesetzt. Ebenfalls realisierbar wird ein Grafik-Tablett, mit dem ein direktes Übertragen vorgegebener Skizzen in den Bildspeicher des Computers möglich wird.

Der Vorteil eines R/D-Wandlers ist, daß er mit handelsüblichen, preiswerten Bauteilen arbeitet. Leider ist er nicht ganz so universell einsetzbar (Kap.4.2.).

4.2 Paddle-Steuerung (R/D-Wandler)

4.2.1 Funktionsweise der Paddle-Steuerung

Das Prinzip der Umwandlung eines Widerstandswertes in eine digitale Zahlenangabe beruht darauf, daß die Impulsdauer t eines monostabilen Multivibrators 74LS123 von einem RC-Glied abhängt. Es gilt:

$$t = 0{,}45 \times R \times C.$$

Das bedeutet, daß die Impulsdauer proportional zum eingesetzten Widerstand ist.

Abb.4.1 zeigt das Gesamtschaltbild der Paddle-Steuerung. Sie enthält insgesamt vier Wandler, bei geringerem Bedarf kann selbstverständlich auch nur ein Teilausbau erfolgen.

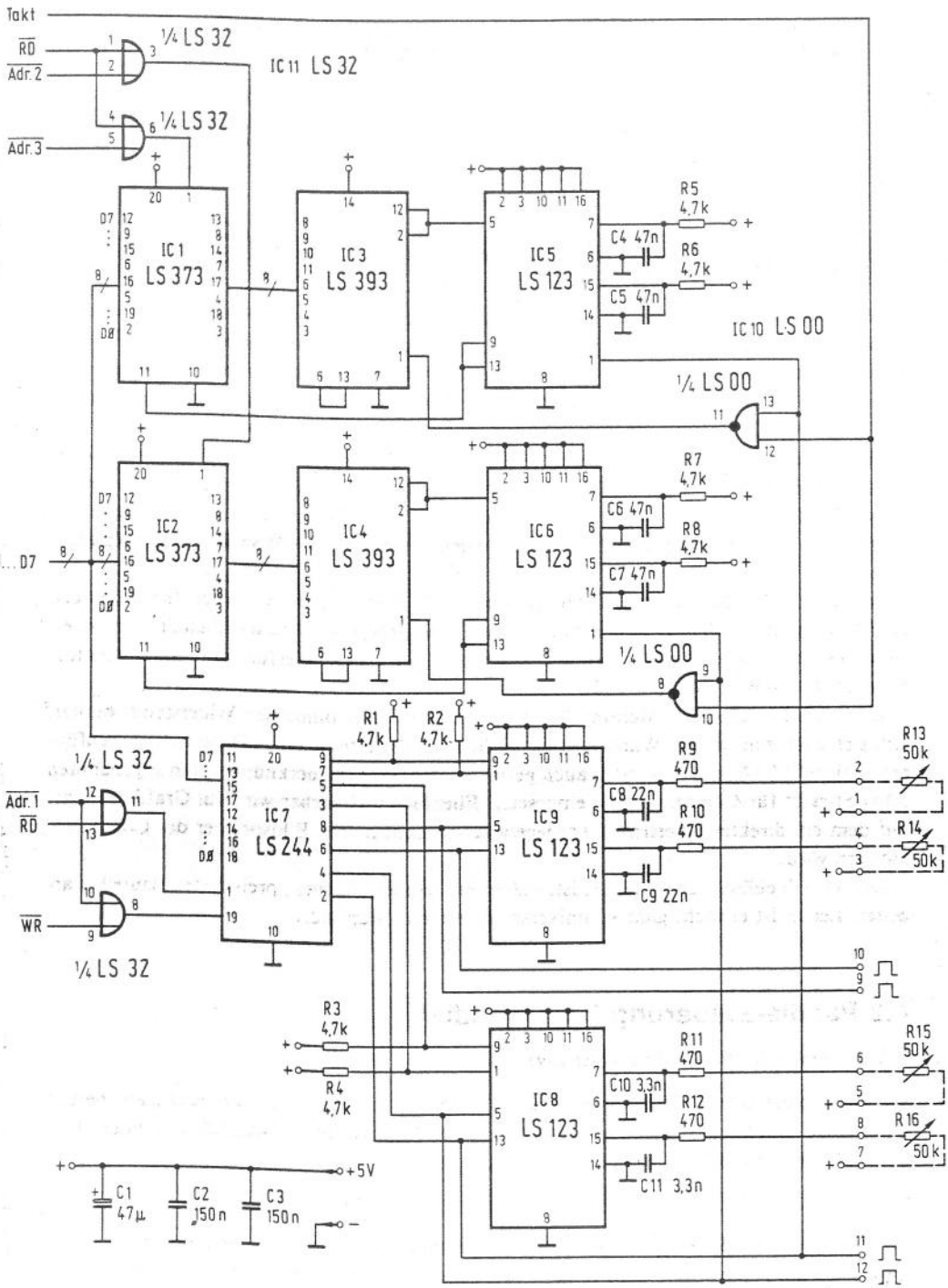


Abb. 4.1 Schaltbild der Paddle-Steuerung

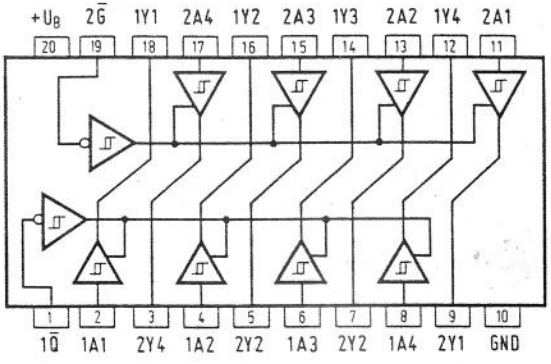


Abb. 4.2 Anschlußbild des Bustreibers 74LS244

Gestartet werden die monostabilen Multivibratoren durch einen Schreibimpuls auf der entsprechenden Datenleitung von Adresse 1. Durch ein Beispiel läßt sich dies am besten verdeutlichen. Mit dem BASIC-Befehl

POKE Adresse 1, 127

wird auf Datenleitung 7 der Adresse 1 ein L-Pegel erzeugt (alle anderen Datenleitungen führen H-Pegel). Über Pin 19 wird dabei der Bus-Treiber IC7 aktiviert (74LS244, Anschlußbild siehe *Abb.4.2*). Dadurch erhält auch seine Ausgangsleitung Pin 9 L-Pegel und das dazugehörige Monoflop (1/2 IC9) wird getriggert. Seine Impulsdauer wird durch R9, R13 und C8 bestimmt. Während dieser Zeit liegt die Ausgangsleitung (Pin 5) auf logisch 1.

Über die Datenleitungen 6, 5 und 4 lassen sich in entsprechender Weise die übrigen Monoflops triggern. Statt 127 lauten die dazugehörigen Dezimalzahlen 191, 223 bzw. 239. Die Periodendauer wird jeweils durch R10/R14/C9, R11/R15/C10 bzw. R12/R16/C11 bestimmt. Die Widerstände R1 ... R4 sorgen dafür, daß bei nicht angesprochenem IC7 (Ausgänge hochohmig) keine Triggerung der Monoflops erfolgt.

Die zu den einzelnen Monoflops gehörenden Ausgangssignale können auf den niederwertigen 4 Bit von Adresse 1 gelesen werden. Die Digitalwandlung des zugehörigen Widerstandswertes ist möglich, indem der Computer eine programmierte Zählschleife durchläuft, solange der interessierende Monoflop-Ausgang H-Pegel besitzt. Die auf diese Weise erhaltene Zahl ist dann direkt dem Widerstandswert proportional. Bei dieser Methode ist der Computer demnach vollständig durch das Abarbeiten der Zählschleife ausgelastet, solange der Widerstandswert ermittelt wird. Das bedeutet beim ZX 81 außerdem, daß der FAST-Modus verwendet werden muß, da sonst falsche Zählwerte ermittelt werden (zeitkritische Anwendung). Für viele Anwendungsfälle ist das allerdings lästig. Günstig wäre, wenn nach Triggerung der Zahlenwert sofort ausgelesen werden könnte.

Diesem Zweck dient ein kleiner Schaltungszusatz. Die Monoflop-Ausgänge von IC8 triggern jeweils ein Doppel-Monoflop (IC5 bzw. IC6). Diese ICs bewerkstelligen eine Ablaufsteuerung für zwei 8-Bit-Zähler (IC3 bis IC4) sowie die dazugehörigen Datenspeicher (IC1 bzw. IC2). Sind die entsprechenden Ausgänge von IC8 auf H-Pegel, so gelangen über eine Torschaltung (jeweils 1/4 74LS00) Zählimpulse (Systemtakt, 3,2 MHz) auf den Eingang des zugeordneten 8-Bit-Zählers. Die negative Flanke des Monoflop-Impulses von IC8 triggert am Pin 1 von IC5 bzw. IC6 ein Monoflop, dessen positiver Impuls am Pin 11 von IC1 bzw. IC2

für die Übernahme des Zählerstands in den Datenspeicher 74LS373 sorgt. Die negative Flanke dieses Übernahme-Impulses wiederum sorgt über Pin 9 und Pin 5 des IC5 bzw. IC6 für die Rücksetzung des Zählers. Damit der Zählbereich dieser Zähler optimal ausgenützt wird, muß die maximale Impulsdauer der Monoflops in IC8 der Taktfrequenz angepaßt sein. Da die Periodendauer des Taktes 0,31 μs ist, sollte die maximale Impulsdauer etwa 80 μs betragen ($256 \times 0{,}31\ \mu s = 80\ \mu s$).

Nach Ablauf des Zählzyklus repräsentiert der Zählerstand den Wert des betreffenden Widerstands (R15 bzw. R16). Der Zählerstand kann über einen Lesebefehl auf Adresse 2 (IC2) bzw. Adresse 3 (IC1) ermittelt werden.

An der zwölfpoligen Kontaktleiste werden die Potentiometer angeschlossen, außerdem stehen die Ausgangsimpulse der Monoflops zur Verfügung. Die Anschlußbelegung ist in Abb. 4.1 eingetragen. Falls die Monoflops für andere Zwecke verwendet werden sollen, läßt sich die Impulsdauer dem jeweiligen Verwendungszweck durch entsprechende Wahl der zeitbestimmenden Bauteile anpassen.

4.2.2 Aufbau und Inbetriebnahme

Die Platinen-Layouts für System I *(Abb.4.3)* und System II (*Abb.4.4*) unterscheiden sich nur in der zusätzlichen Adreß-Dekodierung, die bei der Platine für System II hinzukommt. *Abb.4.5* zeigt den Bestückungsplan und die Stückliste.

Nachdem Herstellung und Aufbau sowie Adressenauswahl in üblicher Weise erfolgt sind, kann die Platine getestet und in Betrieb genommen werden. Dazu wird sie über die entsprechende Grundplatine mit dem ZX 81 verbunden. An die zwölfpolige Kontaktleiste werden die Potentiometer für die einzelnen Kanäle angeschlossen.

Mit folgendem BASIC-Programm kann sofort die Funktion der beiden mit Hardware-Zählern ausgestatteten Kanäle überprüft werden:

```
100 POKE Adresse 1,0
200 PRINT "2: ";PEEK Adresse 2;
300 PRINT "3: ";PEEK Adresse 3;
400 GOTO 100
```

Für Adresse 1/2/3 werden die selbstgewählten Adreßwerte eingesetzt.

Nachdem das Programm mit RUN gestartet ist (SLOW-Modus), werden in Zeile 100 alle vier Monoflops gleichzeitig getriggert. Da das BASIC-Programm relativ langsam arbeitet, ist bei der Abfrage in Zeile 200 und 300 die Impulsdauer der Monoflops bereits abgelaufen, so daß der ausgelesene Zahlenwert korrekt ist. Durch Verdrehen der Potentiometer sollten sich Zahlenwerte zwischen (nahezu) 0 und 255 einstellen lassen.

Gelingt dies nicht, so weicht die Impulsdauer infolge Exemplarstreuungen zu sehr von den berechneten Werten ab. Hierbei spielen nicht nur die Widerstands- und Kondensatorwerte, sondern auch das IC selbst eine Rolle. Abhilfe ist durch Anpassen der Werte der zeitbestimmenden Bauteile möglich. Wird der Zählbereich nicht vollständig ausgenützt, so ist die maximale Impulsdauer zu kurz. Es muß also entweder der Widerstandswert (z. B. R9 ... R12) oder der Kondensatorwert (C8 ... C11) angehoben werden.

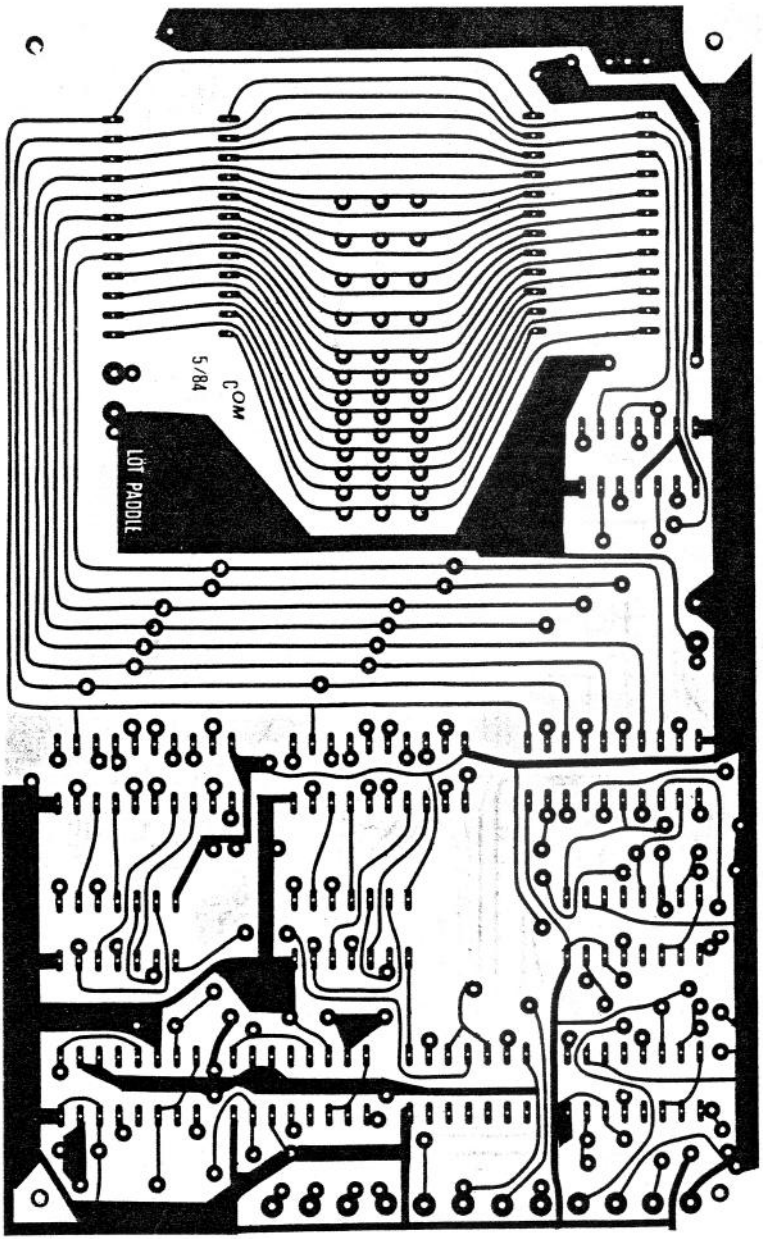


Abb. 4.3a Platinenlayout der Paddle-Steuerung (System I), Lötseite

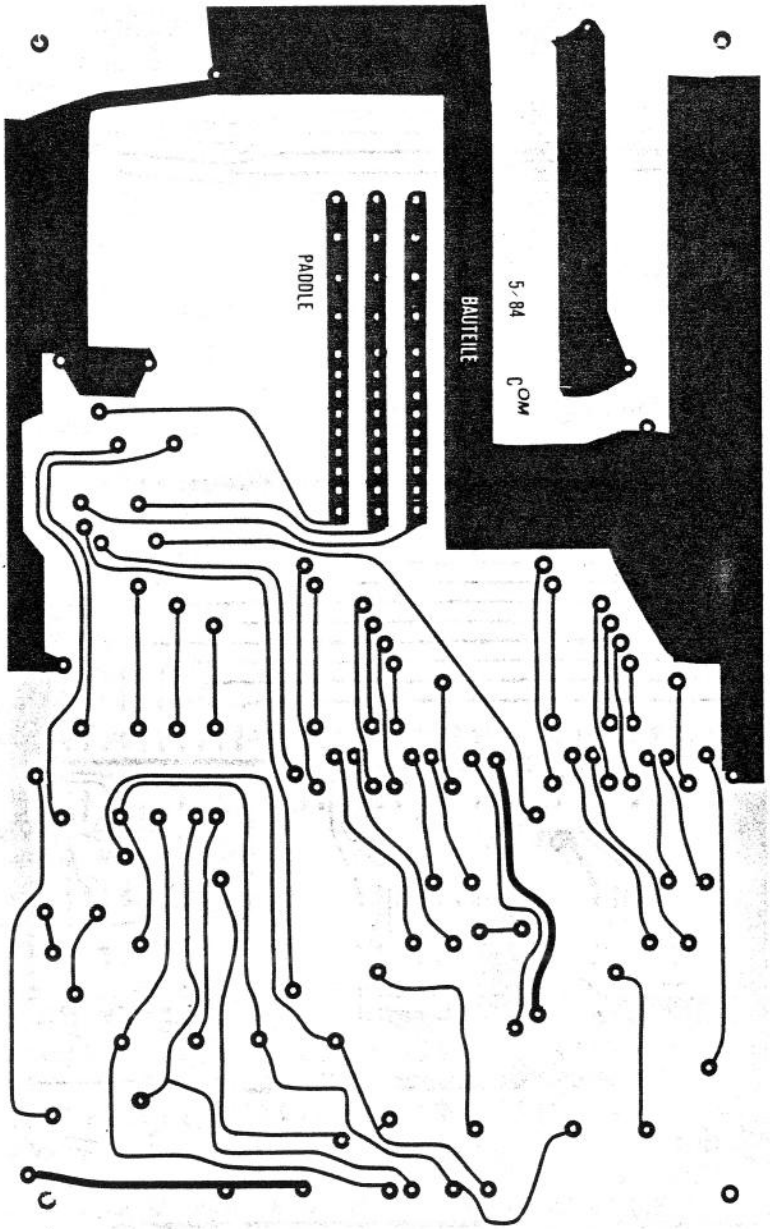


Abb. 4.3b Platinenlayout der Paddle-Steuerung (System I), Bestückungssteuerung

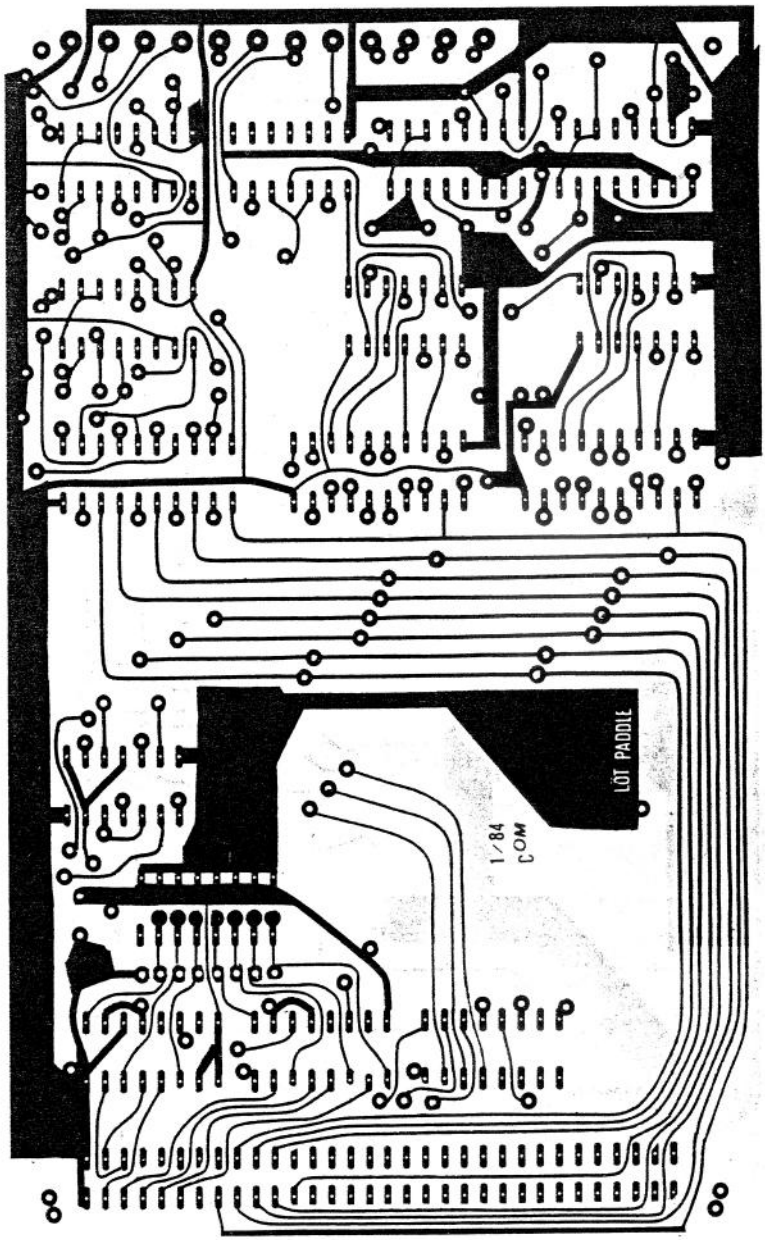


Abb. 4.4a Platinenlayout der Paddle-Steuerung (System II), Lötseite

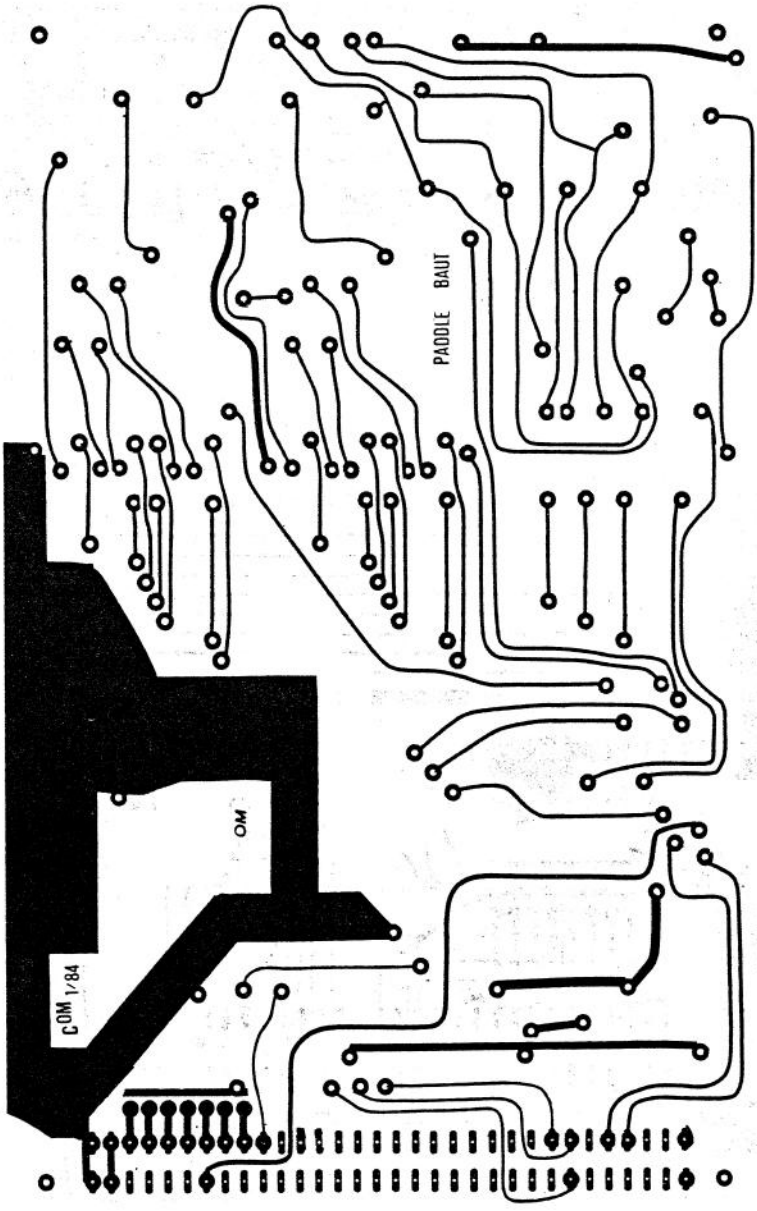


Abb. 4.4b Platinenlayout der Paddle-Steuerung (System II), Bestückungsseite

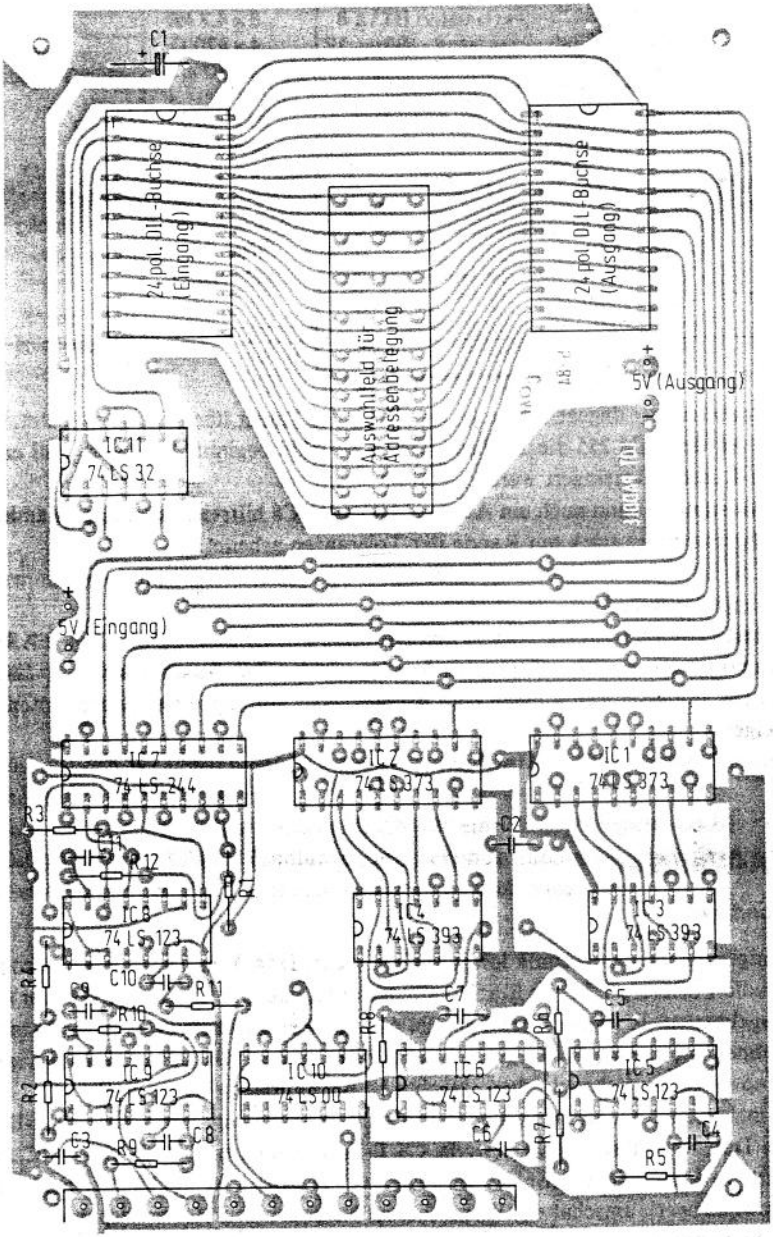


Abb. 4.5 Bestückungsplan (Paddle-Steuerung)

Stückliste zur Vierfach-Paddle-Steuerung

IC1, 2	2 x 74LS373	R1 ... 8	8 x 4,7 kΩ
IC3, 4	2 x 74LS393	R9 ... 12	4 x 470 Ω
IC5, 6, 8, 9	4 x 74LS123	R13 ... 16	4 x 50 kΩ (Potentiometer)
IC7	1 x 74LS244		
IC10	1 x 74LS00		3 x IC-Fass. 20polig
IC11	1 x 74LS32		4 x IC-Fass. 16polig
			4 x IC-Fass. 14polig
C1	1 x 47 µF		2 x IC-Fass. 24polig
C2, 3	2 x 150 nF		1 x Anschlußleiste 12polig
C4 ... 7	4 x 47 nF		(RM 2,54 mm)
C8, 9	2 x 22 nF		Steckstifte
C10, 11	2 x 3,3 nF		

Ist die Impulsdauer dagegen zu lang, so „läuft der Zähler über". Das bedeutet, daß nach Überschreiten der Zahl 255 die Zählung wieder bei Null beginnt. In diesem Fall müssen die Werte von R und C verringert werden.

Unter Umständen kann auch ein Auswechseln von IC8 hilfreich sein, dann nämlich, wenn das IC möglicherweise stark am Rande der Toleranzen arbeitet.

Um die richtige Funktion von IC9 zu überprüfen, muß ein Maschinenprogramm mit Zählschleife eingesetzt werden.

Steht ein Oszillograf zur Verfügung, so läßt man das obige Programm im FAST-Modus ablaufen und untersucht die entsprechenden Impulse an den Anschlüssen 9 ... 12 der Kontaktleiste. Die Impulslängen müssen eine Abhängigkeit von den eingestellten Potentiometerwerten zeigen.

4.3 D/A-Wandler

4.3.1 Grundlagen

Nach heutigem Stand der Technik lohnt es nicht mehr, D/A-Wandler (und genauso natürlich A/D-Wandler) diskret, d. h. aus Einzelbauteilen aufzubauen. Aus der Vielzahl der für diesen Zweck angebotenen ICs wurde der Typ ZN 428 von Ferranti ausgewählt. Er gestattet einen Aufbau ohne viele externe Bauelemente, kann leicht an den Mikroprozessorbus angekoppelt werden und ist einfach abzugleichen. Nachteilig ist sein relativ hoher Preis.

Das Prinzip-Schaltbild des Wandler-ICs ist aus *Abb.4.6* zu erkennen. Das 8-Bit-Datenwort wird vom Datenbus übernommen (wenn $\overline{CS}$ L-Pegel führt) und im Wandler-IC zwischengespeichert. In Abhängigkeit von diesem Datenwort werden CMOS-Schalter betätigt, die ein Widerstandsnetzwerk umschalten. Wenn diesem Netzwerk eine Referenzspannung zugeführt wird, ist die Spannung am Analog-Ausgang ein entsprechender Bruchteil der Referenzspannung. Als Referenzspannung kann eine externe Quelle Verwendung finden oder die auf dem Chip integrierte 2,5-V-Referenz. Digitale und analoge Masseleitung können im Regelfall miteinander verbunden werden.

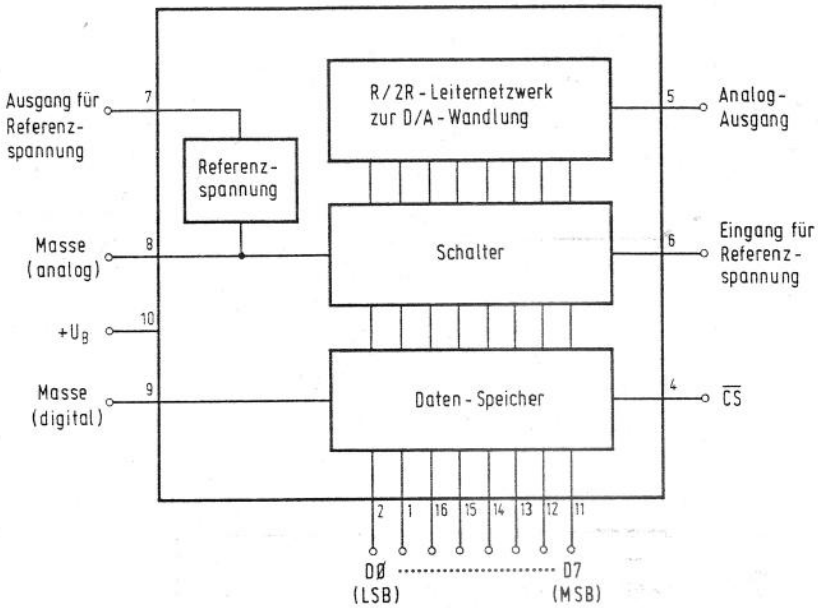


Abb. 4.6 Prinzip-Schaltbild und Anschlußschema der D/A-Wandlers ZN 428

In *Abb.4.7* ist das R/2R-Leiternetzwerk nochmals separat dargestellt. An seinem Ausgang erhält man im unbelasteten Fall die Ausgangsspannung

$U = S \times (U_{ref}/256)$,

wobei S die mit den Schaltern S0 ... S7 eingestellte Binärzahl ist. Als Schalter dienen auf dem Chip integrierte CMOS-Schalter.

Das Datenblatt nennt für das Wandler-IC ZN 428 eine Einschwingzeit von 800 ns. Das bedeutet, daß rund eine Million Wandlungen pro Sekunde durchgeführt werden können. Dadurch ist ein Einsatz bei Tonfrequenzanwendungen problemlos möglich.

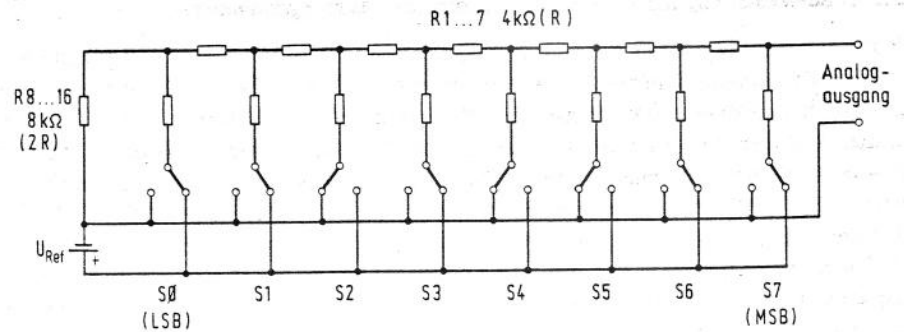


Abb. 4.7 R/2R-Leiternetzwerk zur D/A-Wandlung

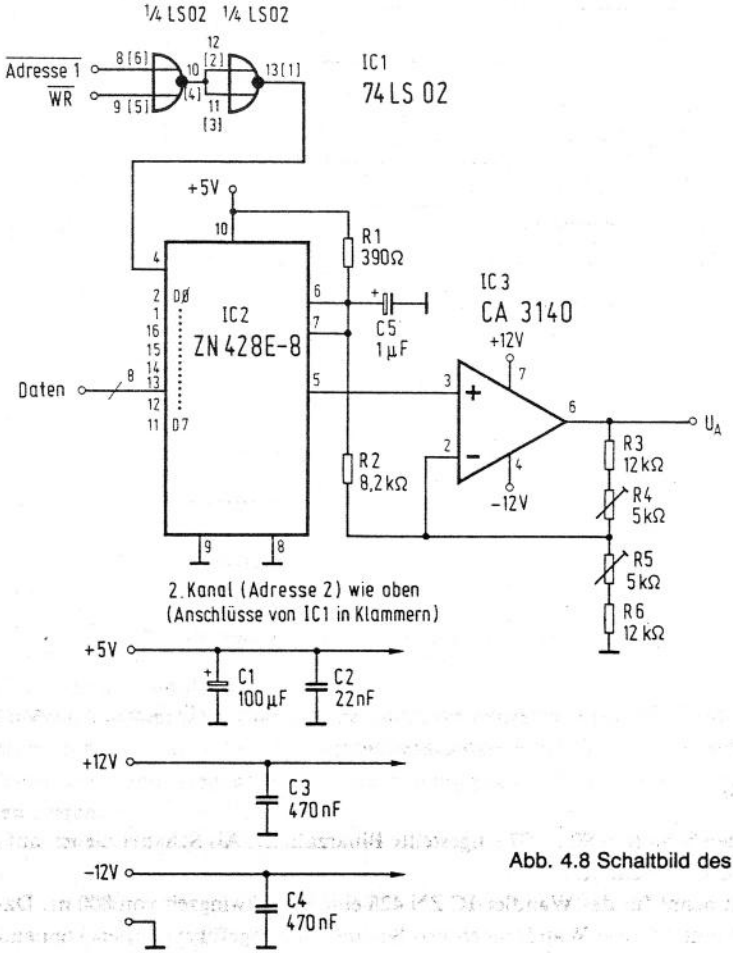


Abb. 4.8 Schaltbild des D/A-Wandlers

4.3.2 Gesamtschaltung des D/A-Wandlers (bipolare Ausgangsspannung)

Die Gesamtschaltung *(Abb. 4.8)* enthält zwei D/A-Wandler. Selbstverständlich kann auch hier wieder ein Kanal allein bestückt werden. Es entfallen dann IC4 und IC5 sowie die dazugehörigen Widerstände und Kondensatoren. Bei System I erfolgt eine Verknüpfung des Adreß- mit dem WR-Signal über ein NOR-Gatter. Da das Wandler-IC zum Aktivieren L-Pegel an Pin 4 benötigt, wird mit einem weiteren NOR-Gatter das Signal invertiert. Das hierfür eingesetzte IC1 (1/2 74LS02, die andere Hälfte ist für Kanal 2 zuständig) ist bei System II nicht nötig und entfällt.

IC2 übernimmt die Daten vom Bus und wandelt sie in eine dem Datenwort proportionale Ausgangsspannung. Als Referenzspannung wird die intern erzeugte 2,5-V-Versorgung benützt. R1 ist Vorwiderstand für die Speisung der Referenzquelle aus der + 5-V-Versorgung, C5 dient zur Glättung.

Damit schwankt die Ausgangsspannung an Pin 5 zwischen 0 V und 2,55 V. Um ein nullsymmetrisches Ausgangssignal zu erhalten, wird der nachgeschaltete Pufferverstärker (IC3) in seinem Ausgangspotential verschoben. Da er einen Verstärkungsfaktor von zwei besitzt, genügt es hierzu, seinen Minus-Eingang mit der Referenzspannung (+2,5 V) zu verbinden. R4 und R5 dienen zum Abgleich.

4.3.3 Aufbau und Inbetriebnahme

In *Abb.4.9* (System I) und *Abb.4.10* (System II) ist das Platinenlayout zu sehen, *Abb.4.11* enthält den Bestückungsplan und die Stückliste. Der Aufbau erfolgt wie üblich. Bei den Potentiometern handelt es sich um Spindeltrimmer (z. B. Cermet). Beim Kauf ist darauf zu achten, daß das Raster mit der Platinenauslegung übereinstimmt, andernfalls muß die Platine überarbeitet werden. Für den Anschluß der Ausgangsleitungen sind zweipolige Kontaktleisten im Rastermaß 2,54 mm vorgesehen. Genausogut können aber auch einfache Steckstifte verwendet werden.

Um die Schaltung in Betrieb zu nehmen, ist neben der üblichen Verbindung zum Computer (über die entsprechende Grundplatine) auch eine symmetrische Betriebsspannung für die Operationsverstärker notwendig (+/− 8 ... 15 V).

Zum Test wird ein Voltmeter (Digitalvoltmeter ist der Genauigkeit wegen vorzuziehen) an den Ausgang des zu testenden Kanals angeschlossen. Nach Ausführung des Befehls

POKE Adresse, 0

muß sich eine Ausgangsspannung in der Gegend von −5 V einstellen, mit

POKE Adresse, 128

sollten es ca. 0 V sein und

POKE Adresse, 255

sollte eine Anzeige von etwa +5 V bewirken.

Falls dies nicht der Fall ist, sollte überprüft werden, ob der Ausgang der Wandler-ICs reagiert (daß alle Betriebsspannungen und Leitungsführungen überprüft wurden, wird vorausgesetzt). Hierzu schließt man das Voltmeter an Pin 5 von IC2 bzw. IC4 an und arbeitet wie oben angegeben. Da jetzt Offset-Verschiebung und Verstärkung fehlen, liegen die Spannungen zwischen 0 V und 2,5 V.

Wenn auch jetzt nicht die erwartete Anzeige erfolgt, muß die Adreßselektion überprüft werden. Dazu ist ein Oszilloskop notwendig. Während das nachfolgende kleine Programm im FAST-Modus abläuft, wird Pin 4 des Wandler-ICs überwacht. Es müssen die $\overline{CS}$-Impulse (L-aktiv) zu erkennen sein.

```
100 POKE Adresse, 0
200 GOTO 100
```

Sind die ersten Überprüfungen zufriedenstellend abgelaufen, so wird der Abgleich durchgeführt. Nach dem Befehl

POKE Adresse, 0

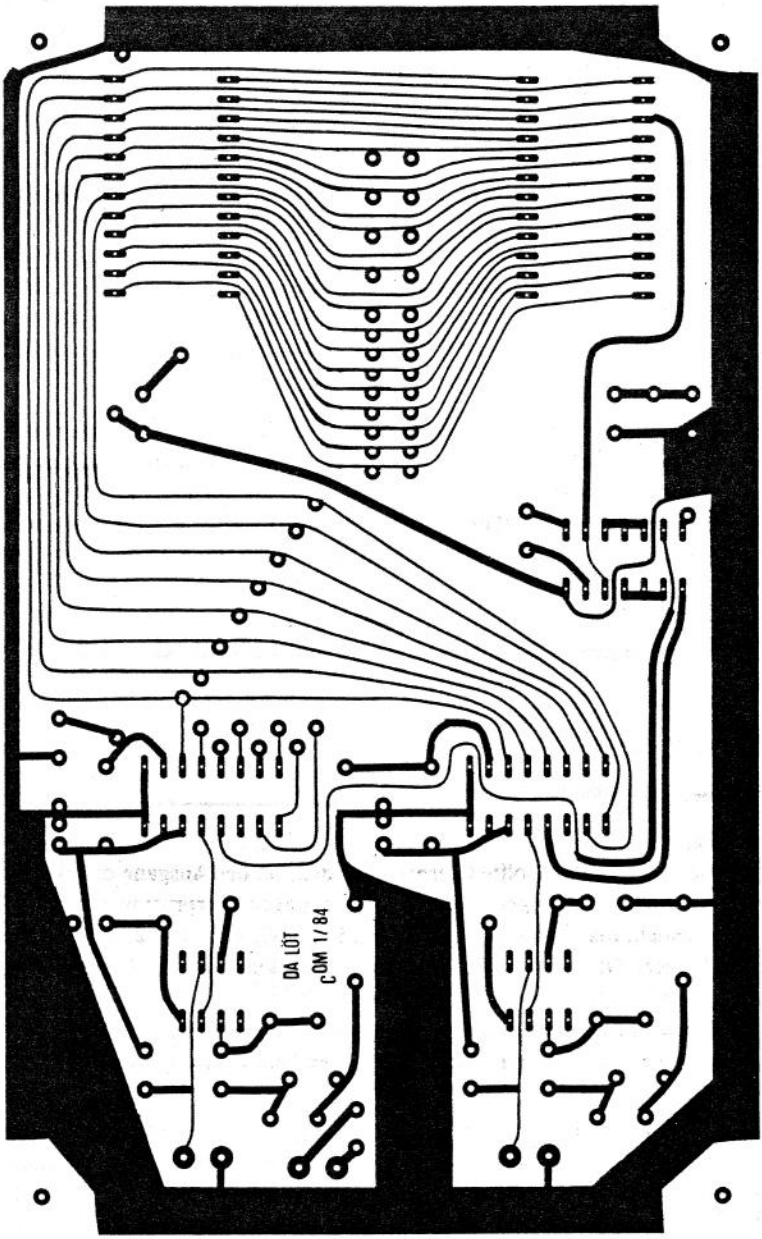


Abb. 4.9a Platinenlayout des D/A-Wandlers (System I), Lötseite

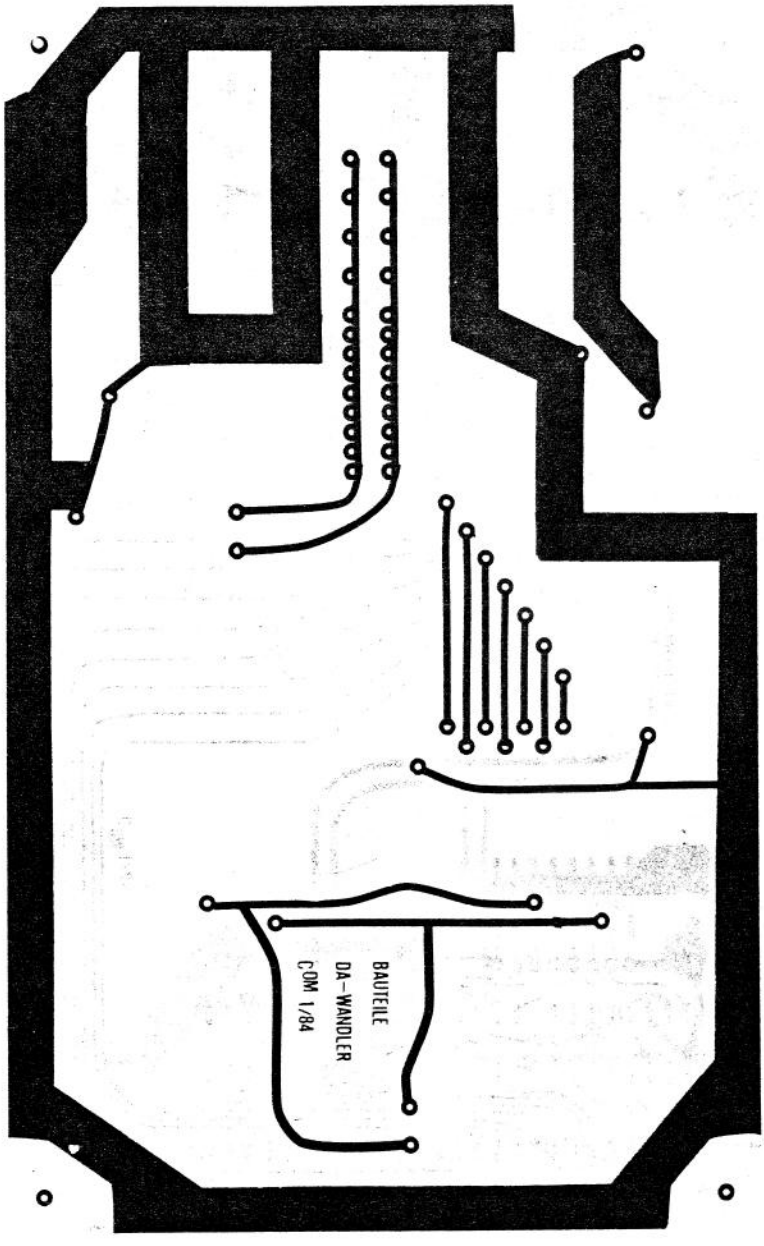


Abb. 4.9b Platinenlayout des D/A-Wandlers (System I), Bestückungsseite

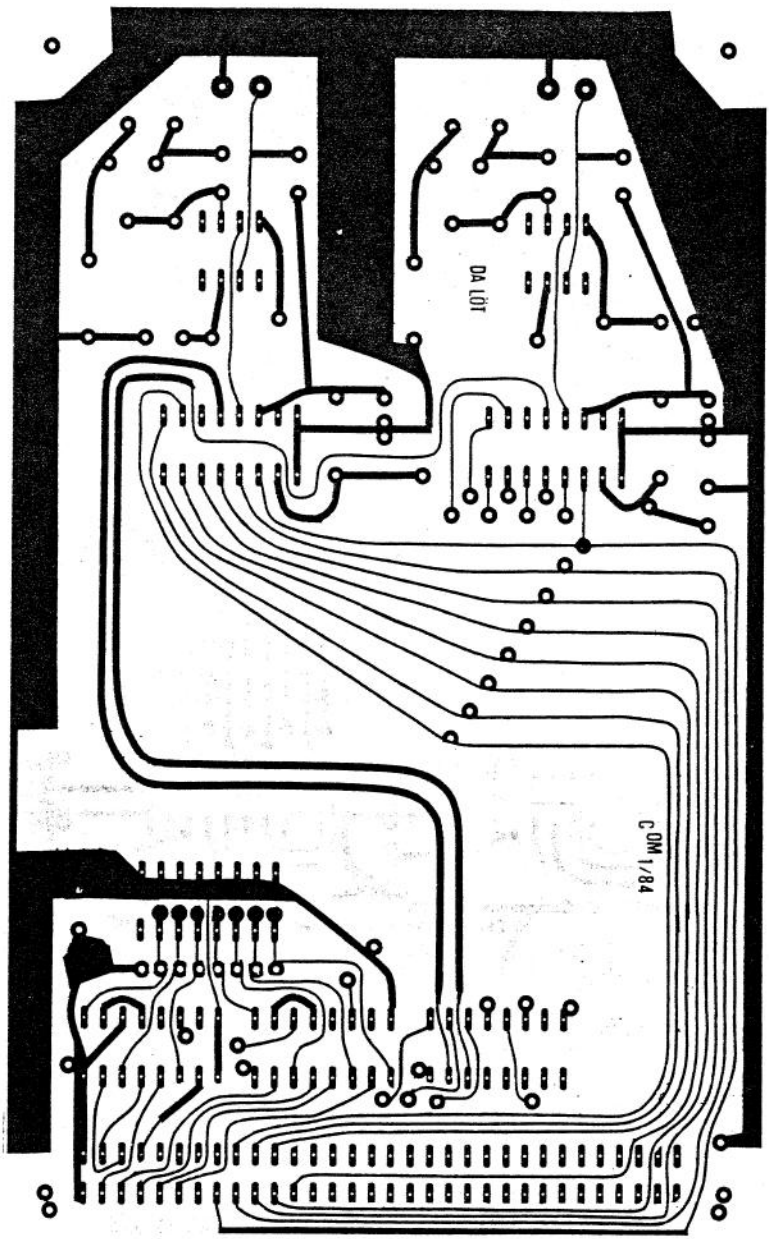


Abb. 4.10a Platinenlayout des D/A-Wandlers (System II), Lötseite

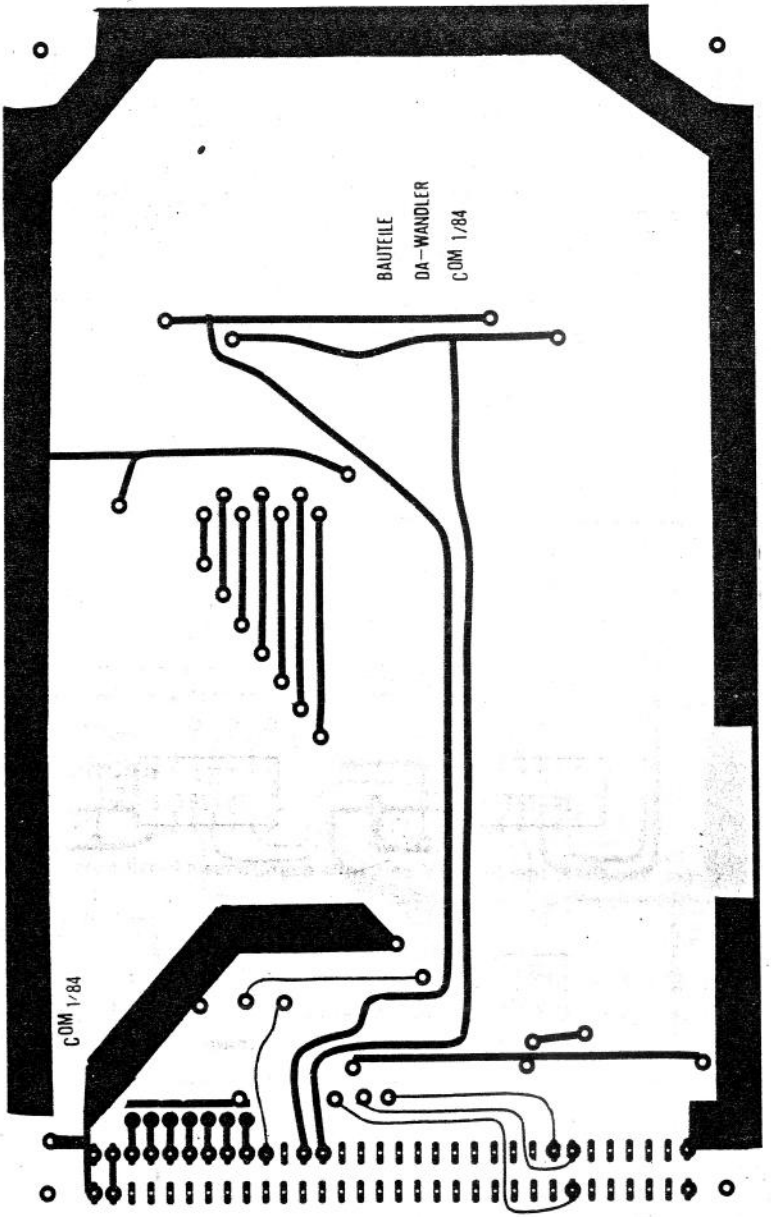


Abb. 4.10b Platinenlayout des D/A-Wandlers (System II), Bestückungsseite

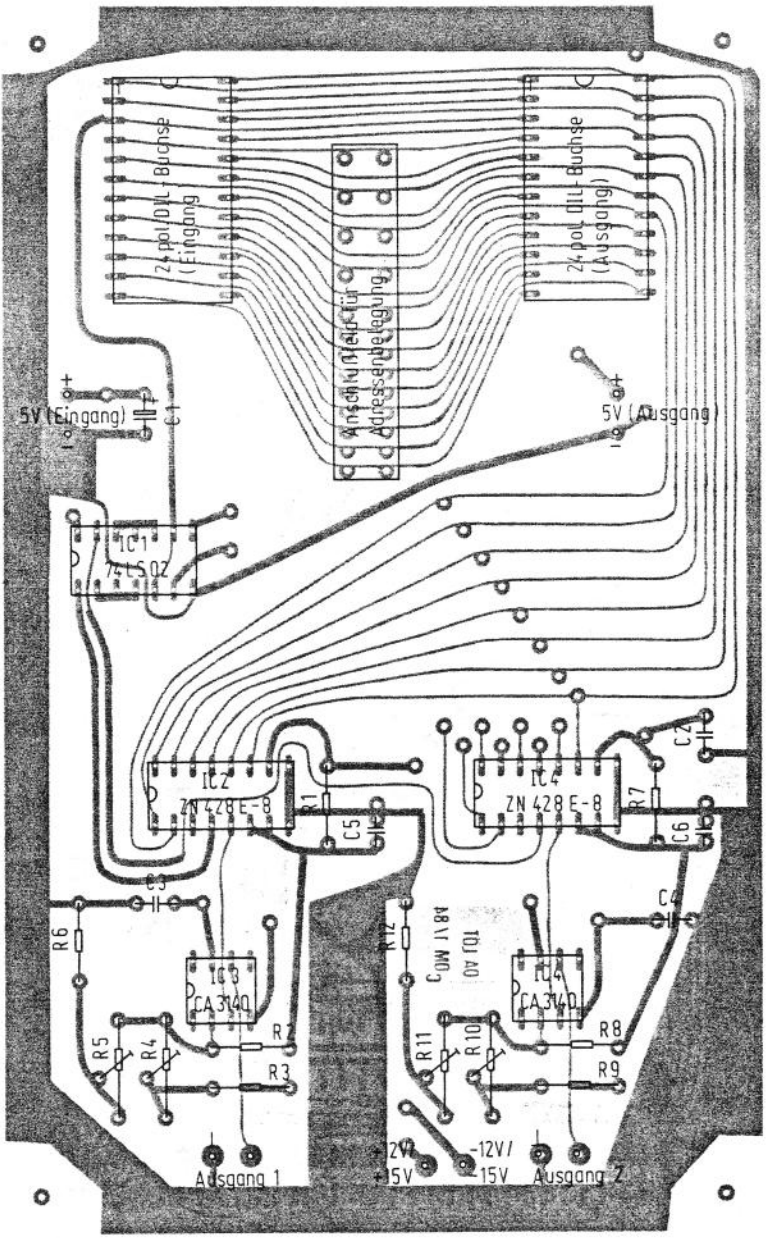


Abb. 4.11 Bestückungsplan des D/A-Wandlers

Stückliste zum D/A-Wandler

IC1	1 x 74LS02 (entfällt bei System II)	R3, 6, 9, 12	4 x 4,7 kΩ
IC2, 4	2 x ZN 428E-8	R4, 5, 10, 11	4 x Spindeltrimmer 5 kΩ
IC3, 5	2 x CA 3140		
			2 x IC-Fass. 16polig
C1	1 x 100 µF		1 x IC-Fass. 14polig
C2	1 x 22 nF		2 x IC-Fass. 8polig
C3, 4	2 x 470 nF		2 x IC-Fass. 24polig
C5, 6	2 x 1 µF		Steckstifte
R1, 7	2 x 390 Ω		
R2, 8	2 x 8,2 kΩ		

wird mit R4 die Ausgangsspannung auf −5,00 V eingestellt. Anschließend gibt man

POKE Adresse, 255

ein und gleicht mit R5 auf +5,00 V ab. Dieser Abgleich wird nun wechselweise noch ein paar Mal wiederholt, bis sich keine Veränderung mehr ergibt.

POKE Adresse, 128

sollte nun 0,0 V liefern.

Damit ist der Abgleich beendet und die D/A-Karte kann eingesetzt werden.

4.4 A/D-Wandler

4.4.1 Grundlagen

Auch für A/D-Wandler-Anwendungen wird eine Vielzahl von IC-Typen angeboten. Hier fiel die Wahl auf den 8-Bit-Wandler ZN 427. Wie sein D/A-Kollege ZN 428 gestattet er einen einfachen Schaltungsaufbau ohne viel Peripherie.

In *Abb. 4.12* ist das Anschlußbild und Innenleben des A/D-Wandlers zu erkennen. Überraschend erscheint wohl, daß dieser Wandler intern einen D/A-Wandler enthält. Grund dafür ist das hier verwendete Wandlerprinzip: „Sukzessive Approximation". Man könnte diesen Ausdruck übersetzen mit „schrittweise Annäherung (an den wirklichen Wert)". Dabei wird an den internen D/A-Wandler eine Binärzahl übergeben. Nachdem sie in die zugehörige Spannung gewandelt ist, wird sie im Komparator mit der Eingangsspannung verglichen. Abhängig davon, ob die Eingangsspannung größer oder kleiner als der intern vorgegebene Wert ist, wird die Binärzahl erhöht oder erniedrigt, solange bis die Eingangsspannung erreicht ist.

Um den Annäherungsvorgang möglichst schnell (und auch immer gleich schnell) beenden zu können, verbietet sich der an sich naheliegende Gedanke, die Binärzahl von Null aus zu erhöhen, bis Gleichheit zwischen Eingangsspannung und intern erzeugter Spannung besteht. Grund: Liegt die Eingangsspannung an der oberen Grenze, so sind 256 Wandlungsschritte notwendig, im anderen Extrem wäre es nur ein Wandlungsschritt.

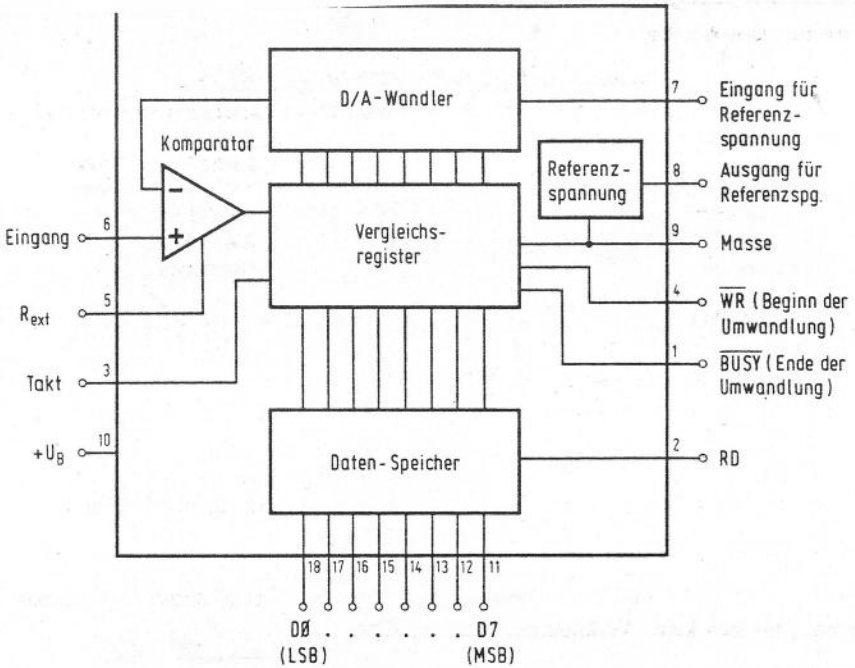


Abb. 4.12 Prinzip-Schaltbild und Anschlußschema des A/D-Wandlers ZN 428

Daher wird ein anderes Verfahren angewendet. In *Abb. 4.13* ist gezeigt, wie ein solcher Wandlungszyklus abläuft. Trifft ein $\overline{WR}$-Impuls (Start der Wandlung) an Pin 4 ein, so wird mit der nächsten fallenden Flanke des Taktsignals (Pin 3) die $\overline{BUSY}$-Leitung aktiv (L-Pegel) und signalisiert den Beginn der Umwandlung. Gleichzeitig wird die Leitung D7 auf logisch 1 gesetzt (alle anderen Datenleitungen werden logisch 0). Nun testet der Komparator, ob die Eingangsspannung größer oder kleiner als die dem Datenwort 128 entsprechende Spannung ist. In Abhängigkeit von dieser Entscheidung wird bei der nächsten fallenden Flanke des Taktes die Datenleitung D7 entweder auf 0 gesetzt oder bleibt logisch 1. Im hier gezeigten Beispiel ist die Eingangsspannung niedriger, so daß D7 wieder zurückgesetzt wird.

Gleichzeitig mit der Entscheidung für D7 erhält nun D6 H-Pegel. Jetzt wiederholt sich das Entscheidungsspiel erneut. Im Beispiel war die Eingangsspannung höher als der intern erzeugte Vergleichswert, so daß D6 gesetzt bleibt.

In derselben Weise werden die übrigen Datenleitungen überprüft. Nach der Entscheidung über D0 ist der Wandlungsprozeß abgeschlossen. Daher erhält die $\overline{BUSY}$-Leitung mit der nächsten fallenden Taktflanke wieder H-Pegel. Der Zustand dieser Leitung signalisiert also, ob die Wandlung beendet ist und ob der Computer auf den gespeicherten Datenwert zugreifen kann. In unserem Beispiel wurde der Datenwert 4Ch bzw. 76d ermittelt.

Man erkennt, daß nach maximal zehn Taktperioden die Wandlung durchgeführt ist und der Digitalwert zur Verfügung steht. Diese Umwandlungszeit ist darüber hinaus unabhängig von der Größe der Eingangsspannung stets konstant. Für den ZN 427 wird die maximale Umwandlungszeit mit 10 μs angegeben. Damit sind rund 100 000 Wandlungen pro Sekunde

durchführbar, d. h. Audio-Anwendungen sind ohne weiteres möglich. Zu beachten ist allerdings, daß die Wandlungszeit von der Taktfrequenz abhängt. Da hier höchstens 1 MHz erlaubt ist, kann der Systemtakt des ZX 81 (3,2 MHz) keine Verwendung finden.

4.4.2 Gesamtschaltung des A/D-Wandlers

Die A/D-Karte enthält ebenfalls zwei Kanäle. In *Abb.4.14* ist hiervon nur ein Kanal gezeichnet. Wie üblich kann die Bestückung auf einen Kanal beschränkt werden, indem die entsprechenden Bauteile weggelassen werden. Der hier verwendete Bustreiber 74LS240 stimmt mit dem in Abb.4.2 aufgeführten Typ 74LS244 in der Anschlußbelegung vollständig überein. Im

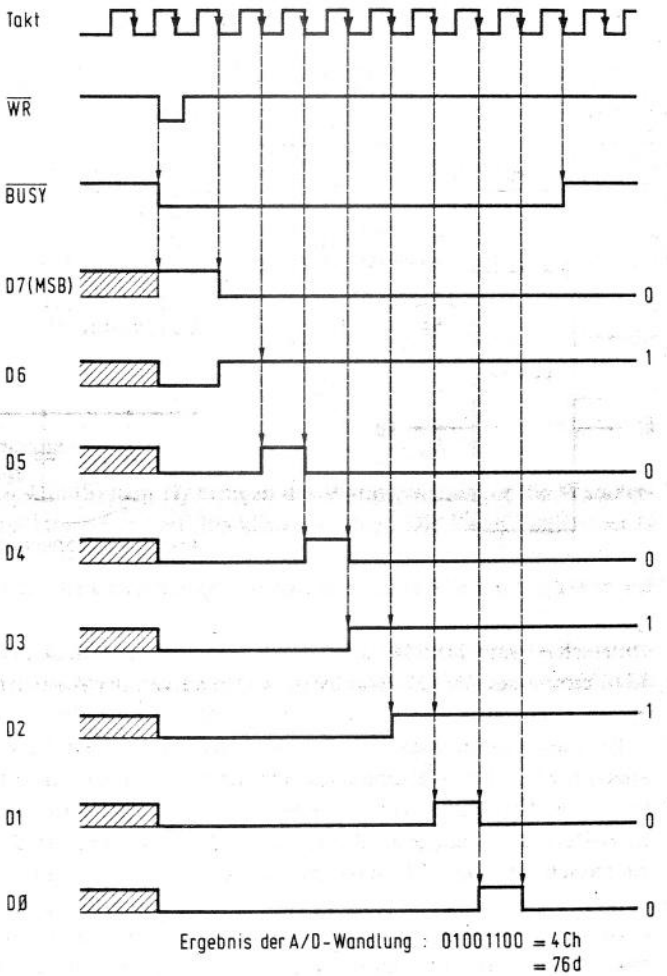


Abb. 4.13 Impulsdiagramm für einen Wandlungszyklus

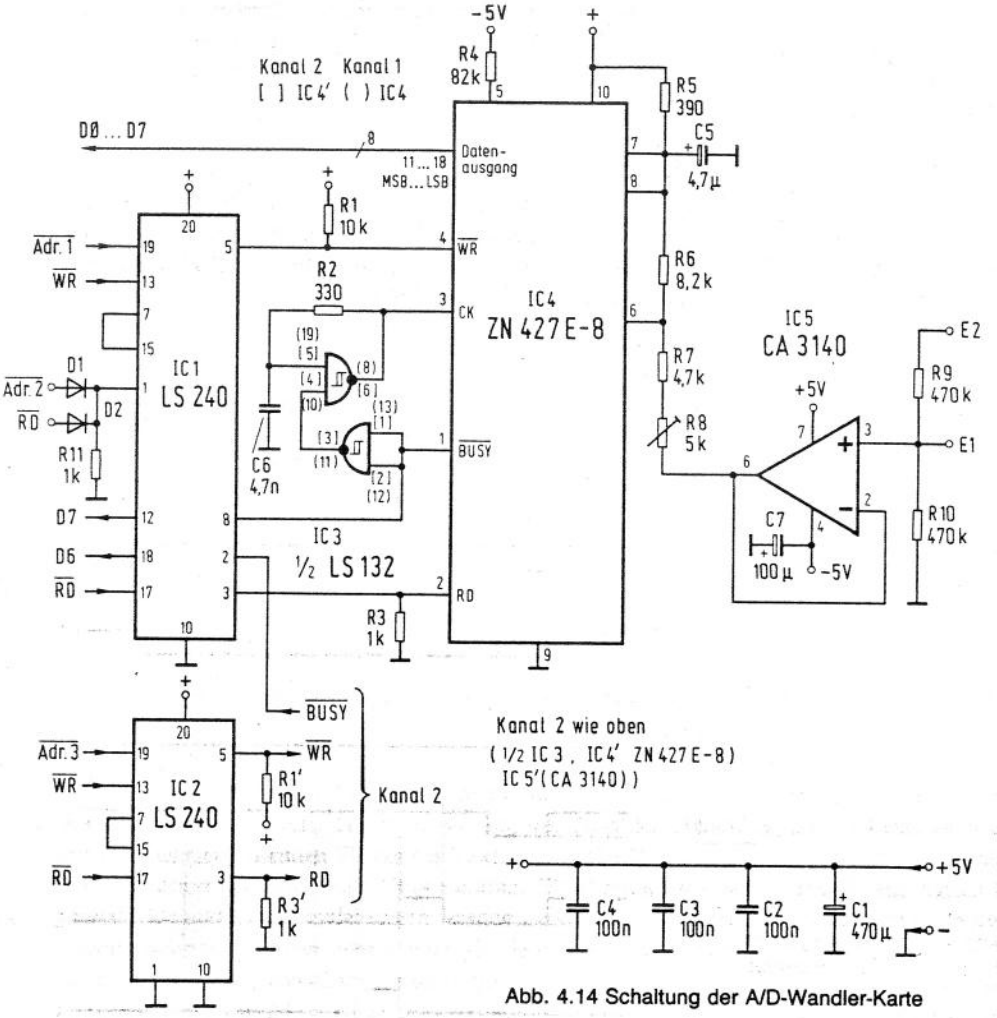


Abb. 4.14 Schaltung der A/D-Wandler-Karte

Unterschied zum 74LS244 invertiert er allerdings die Bussignale. Dies ist notwendig, da die RD-Leitung des ZN 427 H-aktiv ist, während von der Ansteuerung her L-aktive Signale zur Verfügung stehen.

Bei einem Schreibzugriff auf Adresse 1 wird die eine Hälfte des Bustreibers IC1 über Pin 19 angesprochen. Der $\overline{WR}$-Impuls gelangt nach zweimaliger Invertierung an Pin 4 von IC4 und startet den Wandlungszyklus, wie bereits erläutert. R1 sorgt dafür, daß diese Leitung eindeutig H-Pegel hat, solange die Ausgänge von IC1 hochohmig sind. Über die $\overline{BUSY}$-Leitung wird ein freischwingender TTL-Oszillator gestartet, der in der angegebenen Dimensionierung auf ca. 600 kHz schwingt und die Taktfrequenz für den Wandler erzeugt.

Der $\overline{BUSY}$-Ausgang wird gleichzeitig an Pin 8 von IC1 geführt. Dieser Anschluß gehört zur zweiten Hälfte des Bustreibers, die über Pin 1 aktiviert wird. Dies erfolgt bei einem Lesezu-

griff auf Adresse 2. Dabei wird die Information der $\overline{\text{BUSY}}$-Leitung invertiert auf die Datenleitung D7 geschaltet. Auf Bit 7 von Adresse 2 kann also der Zustand des Wandlers abgefragt werden. Ist D7 von Adresse 2 logisch 1, so erfolgt gerade eine Umwandlung (vergl. Abb. 4.13, wegen der Invertierung durch den Bustreiber sind die Signal-Pegel gegenüber dort vertauscht). Die Daten des A/D-Wandlers sind gültig, sobald D7 von Adresse 2 logisch 0 wird. Dies läßt sich für ein Programm zur kontinuierlichen Abfrage des Wandlerwertes ausnützen. Man vermeidet dadurch das Auslesen ungültiger Werte. Auf D6 von Adresse 2 steht die entsprechende Information für den zweiten Wandler-Kanal zur Verfügung.

Das Wandlungsergebnis wird unter Adresse 1 (für Kanal 1) bzw. Adresse 3 (Kanal 2) gelesen. Bei einem Lesevorgang auf der jeweiligen Adresse ist der zu den Pins 17 und 3 gehörende Teil des Bustreibers angesprochen, so daß nach Invertierung des $\overline{\text{RD}}$-Signals IC4 veranlaßt wird, die Daten an den Anschlüssen 11 ... 18 für den Datenbus bereitzustellen. Ansonsten sind die Datenausgänge von IC4 hochohmig und belasten den Datenbus nicht, können also direkt mit ihm verbunden werden.

IC5 ist als Impedanzwandler für die zu messende Eingangsspannung geschaltet. R9 und R10 bilden einen Spannungsteiler, so daß eine maximale Eingangsspannung von $+/-2{,}5$ V (E1) bzw. $+/-5$ V (E2) möglich ist. Über R6 ergibt sich ein Offset der Eingangsspannung für den Wandler, so daß nullsymmetrische Spannungen gemessen werden können. Mit R8 wird abgeglichen (Spindeltrimmer).

Da das Wandler-IC als Eingangsspannung nicht wesentlich mehr als 5 V verträgt, sollte die Versorgungsspannung des Impedanzwandlers nur $+/-5$ V betragen, wenn ein Überschreiten dieses Maximalwertes nicht auf andere Weise sicher vermieden werden kann.

R4 hängt von der Höhe der negativen Versorgungsspannung ab. Der angegebene Wert von 82 kΩ gilt für -5 V (-10 V: 150 kΩ, -15 V: 220 kΩ).

Kanal 2 ist entsprechend aufgebaut.

4.4.3 Aufbau und Inbetriebnahme

Abb.4.15 (System I) und *Abb.4.16* (System II) bringen das Platinenlayout für die Wandler-Karte, in *Abb.4.17* sind Bestückungsplan und Stückliste zu sehen. Die Karte für System II enthält zusätzlich noch die Adreßdekodierung. Außerdem wird hier die negative Versorgungsspannung über die Anschlußleiste zugeführt, so daß der entsprechende Steckstift auf der Platine entfällt.

Nachdem Aufbau und Kontrolle wie gewöhnlich durchgeführt wurden, wird die Karte angeschlossen. An den Eingang E1 des gewünschten Kanals wird eine variable Spannungsquelle sowie ein (idealerweise Digital-) Voltmeter angelegt. Dann gibt man das nachfolgende Testprogramm in den Rechner ein und läßt es im SLOW-Modus ablaufen:

```
100 POKE Adresse 1, 0
200 PRINT PEEK Adresse 1; " ";
300 GOTO 100
```

In Zeile 100 wird die Wandlung gestartet. Da das BASIC-Programm relativ langsam abläuft, ist bei Ausführung der Zeile 200 die Wandlung bereits beendet, so daß keine Abfrage des Wandlerstatus nötig ist.

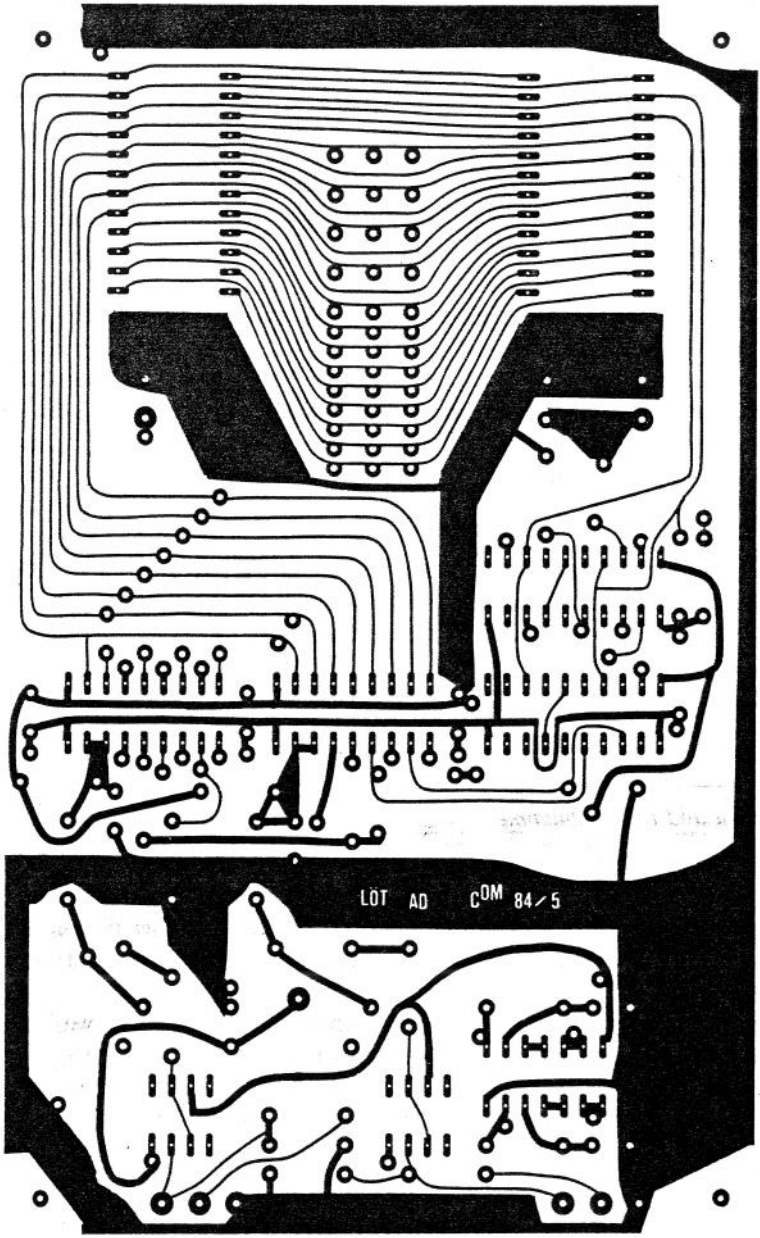


Abb. 4.15a Platinenlayout des A/D-Wandlers (System I), Lötseite

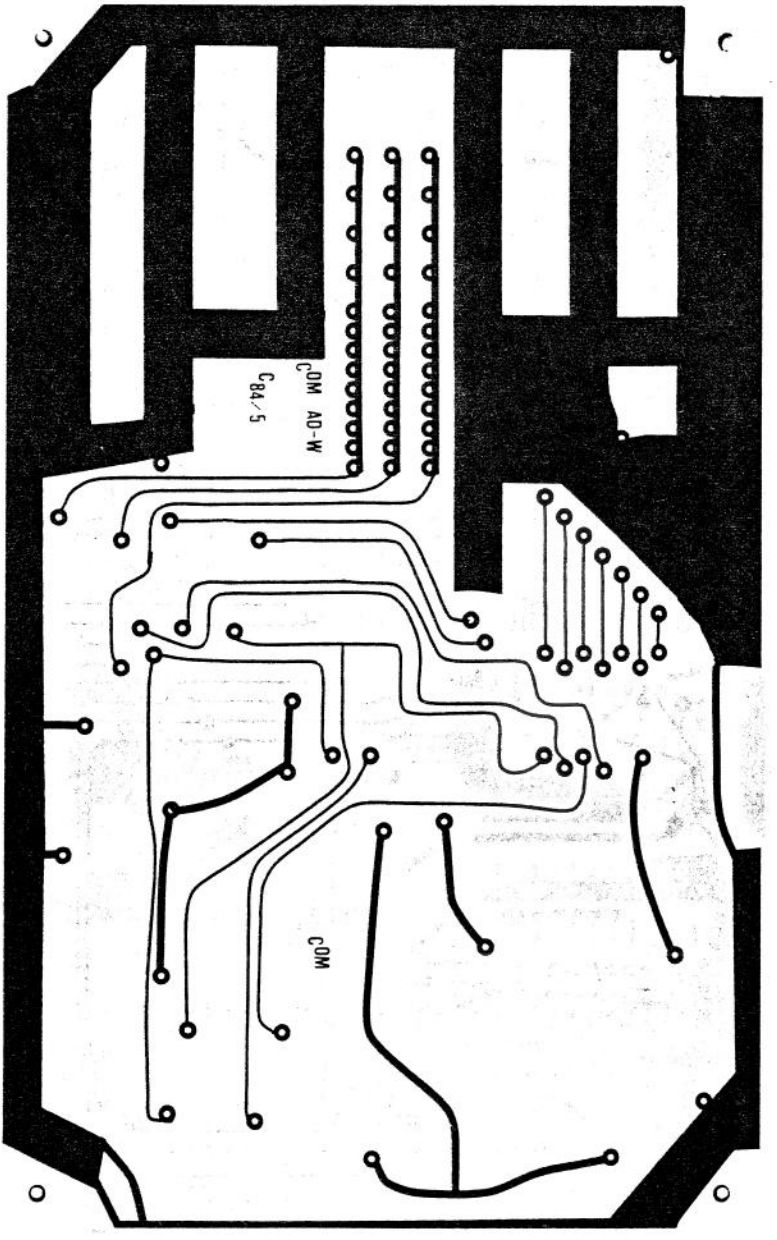


Abb. 4.15b Platinenlayout des A/D-Wandlers (System I), Bestückungsseite

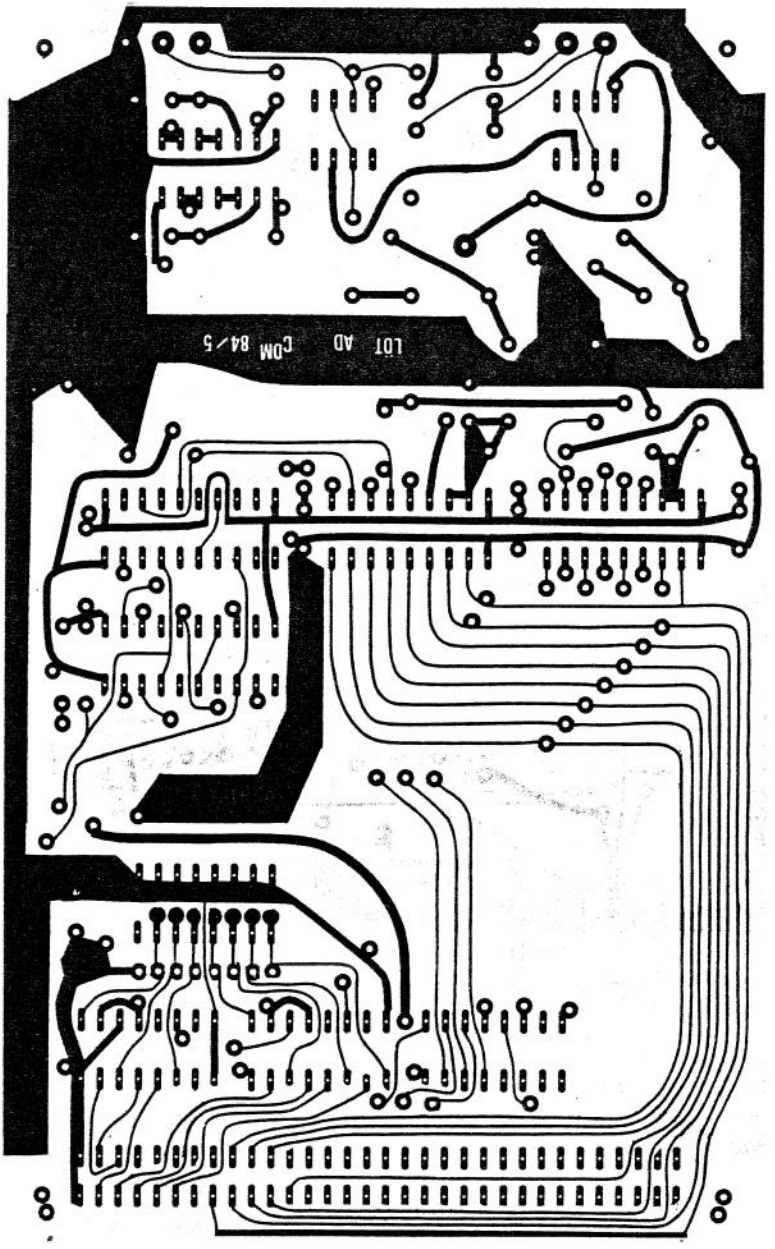


Abb. 4.16a Platinenlayout des A/D-Wandlers (System II), Lötseite

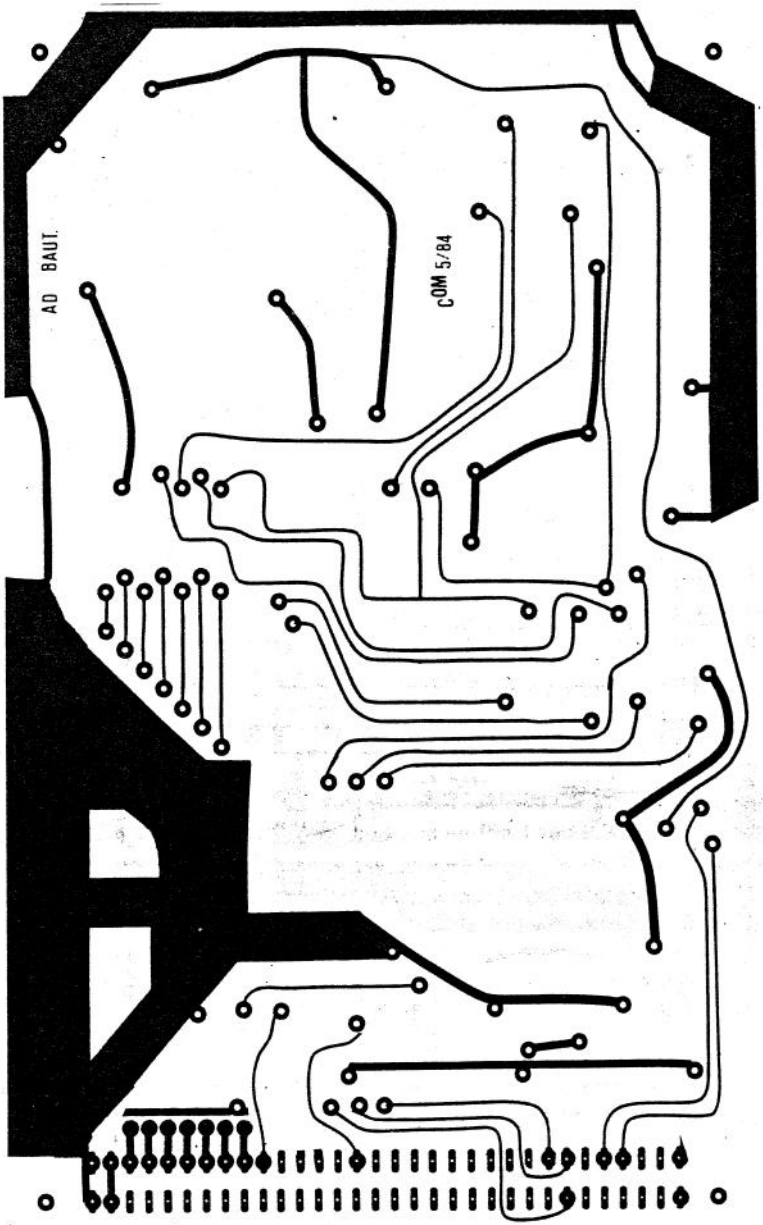


Abb. 4.16b Platinenlayout des A/D-Wandlers (System II), Bestückungsseite

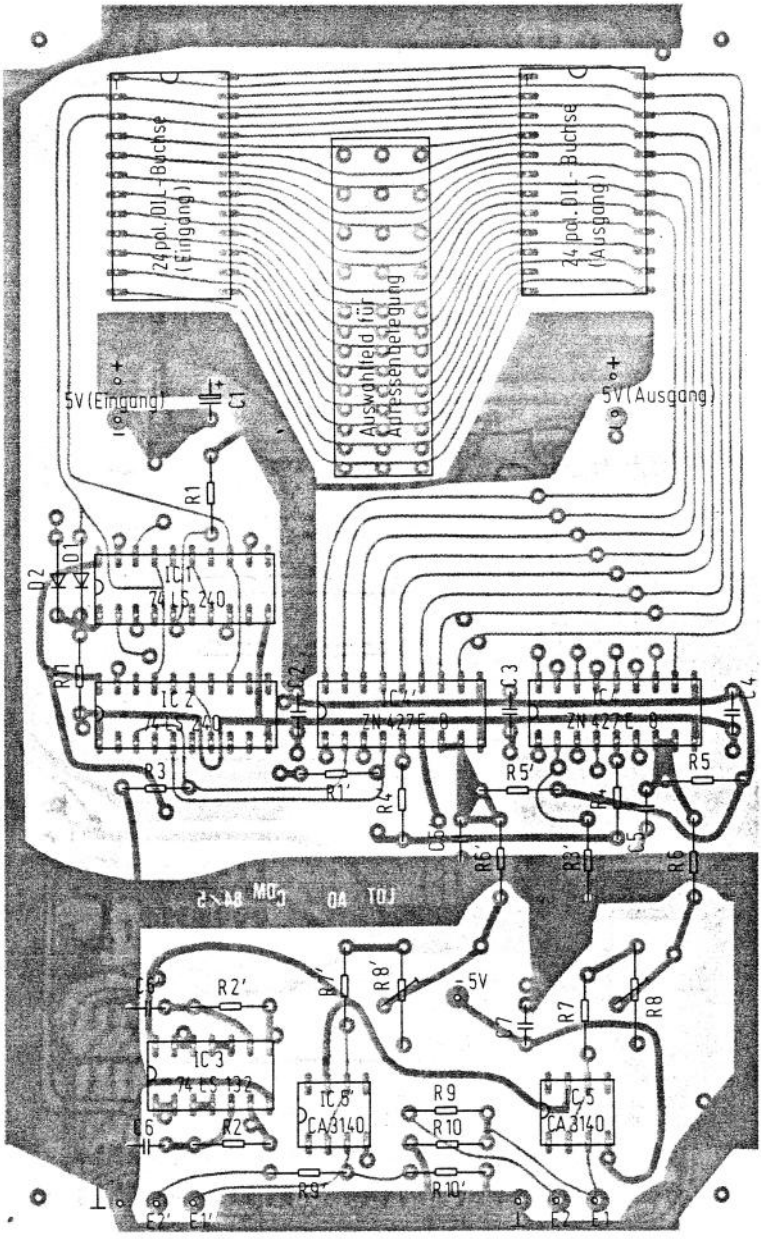


Abb. 4.17 Bestückungsplan (A/D-Wandler)

Stückliste A/D-Wandler

IC1, 2	2 × 74LS240	R9, R9', R10, R10'	4 × 470 kΩ
IC3	1 × 74LS132	R11	1 × 1 kΩ
IC4, 4'	2 × ZN 427E-8	C1	1 × 470 µF/10 V
IC5, 5'	2 × CA 3140	C2 ... C4	3 × 0,1 µF
D1, 2	2 × AA 116 o. ä. Ge-Diode	C5, C5'	2 × 4,7 µF
R1, R1'	2 × 10 kΩ	C6, C6'	2 × 4,7 nF
R2, R2'	2 × 330 Ω	C7	1 × 100 µF/16 V
R3, R3'	2 × 1 kΩ		2 × Stecksockel 24polig
R4, R4'	2 × 82 kΩ (vgl. auch Text)		4 × Stecksockel 20polig
R5, R5'	2 × 390 Ω		1 × Stecksockel 14polig
R6, R6'	2 × 8,2 kΩ		2 × Stecksockel 8polig
R7, R7'	2 × 4,7 kΩ		Kleinteile
R8, R8'	2 × Spindeltrimmer 5 kΩ		

Zum Abgleich stellt man die Spannung auf +2,5 V ein. Dabei sollte ein Zählwert von 255 registriert werden, bei −2,5 V Eingangsspannung muß Null angezeigt werden. Die Eingangsspannung 0 V sollte zu einer Anzeige von 128 führen. Mit dem Spindeltrimmer R8 wird abgeglichen.

Falls die Schaltung nicht wie gewünscht arbeitet, sollte mit Hilfe eines Oszilloskops geprüft werden, ob überhaupt die Adreßkodierung richtig arbeitet. Selbstverständlich werden zuvor sämtliche Betriebsspannungen, auch unmittelbar an den IC-Anschlüssen, überprüft. Darauf läßt man im FAST-Modus ein Testprogramm laufen:

```
100 POKE Adresse 1, 0
200 GOTO 100
```

Jetzt müssen sich an Pin 19 und Pin 5 von IC1 die L-aktiven $\overline{CS}$-Impulse finden lassen, das gleiche gilt für Pin 4 von IC4. Außerdem muß nun an Pin 3 von IC4 die Taktfrequenz nachweisbar sein. Möglicherweise sind auch nur Impulspakete feststellbar, wenn nämlich die Wandlung schon abgelaufen ist, bevor der nächste Startimpuls eintrifft.

Mit dem Programm

```
100 LET A = PEEK Adresse 1
200 GOTO 100
```

müssen sich an Pin 3 von IC1 bzw. Pin 2 von IC4 H-Pulse nachweisen lassen.

Ist der Fehler damit noch nicht ermittelt, so muß der Analogteil überprüft werden. Dazu verändert man die Eingangsspannung des Impedanzwandlers und kontrolliert an Pin 6 die Ausgangsspannung.

In entsprechender Weise wird auch Kanal 2 überprüft und angeglichen. Nach erfolgreich abgeschlossenem Test ist die A/D-Wandler-Karte einsatzbereit.

5 Zeit-Erfassung

5.1 Der Zeitzeichen-Sender DCF 77

Exakte Zeitangaben spielen in einer modernen Industriegesellschaft eine entscheidende Rolle. Die Genauigkeit normaler Uhren, auch wenn es sich um hochpräzise Quarzuhren handelt, reicht dafür längst nicht mehr aus. Aus diesem Grund wurden Zeitzeichensender installiert, die ihre Ganggenauigkeit von Atomuhren ableiten. Dabei wird eine Präzision erreicht, die geradezu phantastisch ist: mit einer Toleranz in der Gegend von 10^{-15} ergibt sich in 10 000 Jahren nur eine Abweichung von 1/10 000 s. Viele Uhren an öffentlichen Gebäuden, Bahnhöfen und auch bei den Rundfunk- und Fernsehanstalten werden von diesem Zeitnormal gesteuert.

So exotisch einen das Ganze anmuten mag, diese hochpräzise Zeitskala kann recht einfach von jedermann genutzt werden. Die Zeitinformation wird in der Bundesrepublik durch einen Längstwellensender in der Nähe von Frankfurt übertragen. Dieser Zeitzeichensender DCF 77 arbeitet auf einer Frequenz von 77,5 kHz und ist ohne größere Schwierigkeiten zu empfangen. Geeignete Empfängerschaltungen wurden schon verschiedentlich in den einschlägigen Fachzeitschriften veröffentlicht.

Am Ausgang des Empfängers erhält man eine Spannung, deren prinzipieller Verlauf in *Abb.5.1* gezeigt ist. Jeweils mit Sekundenbeginn wird das Sendersignal auf 1/4 seiner Maximalstärke abgesenkt. Dabei ist die Zeitdauer der Absenkung nicht einheitlich, sie kann 0,1 s oder 0,2 s betragen. Ein Impuls von 0,1 s bedeutet den binären Wert 0, ein solcher von 0,2 s Dauer den Wert 1. Auf diese Weise können im Laufe einer Minute 60 Bit übertragen werden. Da das Minutenende und damit der Start einer neuen Übertragungssequenz durch eine fehlende Sekundenabsenkung erkennbar ist, bleiben davon noch 59 Bit an Information übrig.

Wie die Zuordnung der einzelnen Bits zur Uhrzeit- und Datumsangabe erfolgt, ist aus *Abb.5.2* zu ersehen. Der Sekundenimpuls 20 zeigt den Beginn der eigentlichen Zeitübertragung an, er ist stets 0,2 s lang. Die verschiedenen Prüfbits gestatten es, Übertragungsfehler zu erkennen.

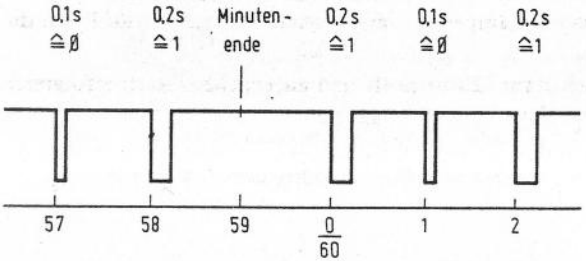


Abb. 5.1 Signalverlauf bei DCF 77

Codierungsschema für DCF-77-Übertragung

Sekunde	Wert	Bedeutung	Beispiel	Auswertung
0 ... 19	—	Verschiedene Senderinformationen		
20	—	Start der Zeitcodierung		
		Minuten		
21	1		1	
22	2		0	
23	4	Einer	0	9
24	8		1	
25	10		1	
26	20	Zehner	0	5
27	40		1	
28	—	Prüfbit	0	
		Stunden		
29	1		0	
30	2		1	2
31	4	Einer	0	
32	8		0	
33	10	Zehner	0	
34	20		0	0
35	—	Prüfbit	0	
		Kalendertag		
36	1		1	
37	2		0	
38	4	Einer	0	1
39	8		0	
40	10	Zehner	0	
41	20		1	2
		Wochentag		
42	1		1	
43	2		1	3
44	4		0	
		Kalendermonat		
45	1		0	
46	2		1	
47	4	Einer	0	2
48	8		0	
49	10	Zehner	1	1
		Kalenderjahr		
50	1		0	
51	2		0	
52	4	Einer	1	4
53	8		0	

Sekunde	Wert	Bedeutung	Beispiel	Auswertung
54	10		0	
55	20		0	
56	40	Zehner	0	8
57	80		1	
58	—	Prüfbit	1	
59	—	Marke fehlt - Identifizierung des neuen Minutenbeginns		

Beispiel: Mi (=3), 21. 12. 1984, 2.59 Uhr

Abb. 5.2 Codierung des Zeitzeichensenders DCF 77

Für eine Auswertung der codierten Zeitinformation bietet sich ein Mikrocomputer geradezu an. Softwaremäßig kann das geschehen, indem der Ausgang des Empfängers (zeitkritisch) auf L- oder H-Pegel abgefragt wird und aus den daraus gewonnenen Informationen die Daten rekonstruiert werden. Idealerweise triggert man das Programm über die Interrupt-Eingänge, so daß der Rechner nur einen Bruchteil seiner Rechenzeit für die Auswertung benötigt.

Leider scheidet beim ZX 81 diese Technik aus, da die Interrupt-Leitungen schon für Zwecke des Betriebssystems belegt sind. Damit bliebe nur noch die Möglichkeit, den Empfängerausgang ständig vom Computer überwachen zu lassen. Da dies im FAST-Modus geschehen müßte (zeitkritisches Programm), wäre sogar eine Bildschirmanzeige unmöglich.

Aus diesem Grunde wurde für den ZX 81 eine teilweise hardware-orientierte Lösung gefunden. Dabei erfolgt die Speicherung der übertragenen Informationen hardwaremäßig in einer Art Schieberegister, die Auslesung und Dekodierung der Speicherdaten übernimmt der Computer. Damit wird nur verhältnismäßig wenig Rechenzeit beansprucht.

5.2 Schaltung zur Hardware-Dekodierung

In *Abb.5.3* ist das Schaltbild der Dekodierung zu sehen. Der angeschlossene Empfänger muß die Trägerabsenkung in Form TTL-kompatibler, negativer Impulse (L-aktiv) liefern. Die Schaltung läßt sich für positive Signale umrüsten, indem bei FF2 (IC1) und FF1 (IC2) jeweils die Anschlüsse 1 und 2 vertauscht werden, außerdem muß die Software bei der Auswertung die invertierten Signale berücksichtigen.

Die Platine ist so ausgelegt, daß nach Anschluß eines DCF-77-Empfängers die Zeitinformation selbsttätig in einen zum Pseudo-Schieberegister zweckentfremdeten Speicher 2114 eingeschrieben wird. Obwohl dieses IC (Speicherkapazität 1 K×4) durch die Speicherung von 59 Bit bei weitem nicht ausgelastet ist, ist dies eine kostengünstige Lösung.

Um die Wirkungsweise der Schaltung zu verstehen, sollte man sich gleichzeitig das Impulsprogramm aus *Abb.5.4* vor Augen halten. Nachdem sich L- und H-Information in unterschiedlich langen Zeitmarken verbergen, muß aus dieser Codierung erst das eigentliche Zeitsignal mit L- und H-Pegel gewonnen werden. Hierzu triggert die fallende Flanke des Zeittakts das Monoflop FF1. Da FF1 eine Impulsdauer von 0,15 s besitzt, ist bei seinem Zurückklappen auf der Taktleitung entweder noch H-Pegel (wenn es sich um eine logische 1 mit 0,2 s Im-

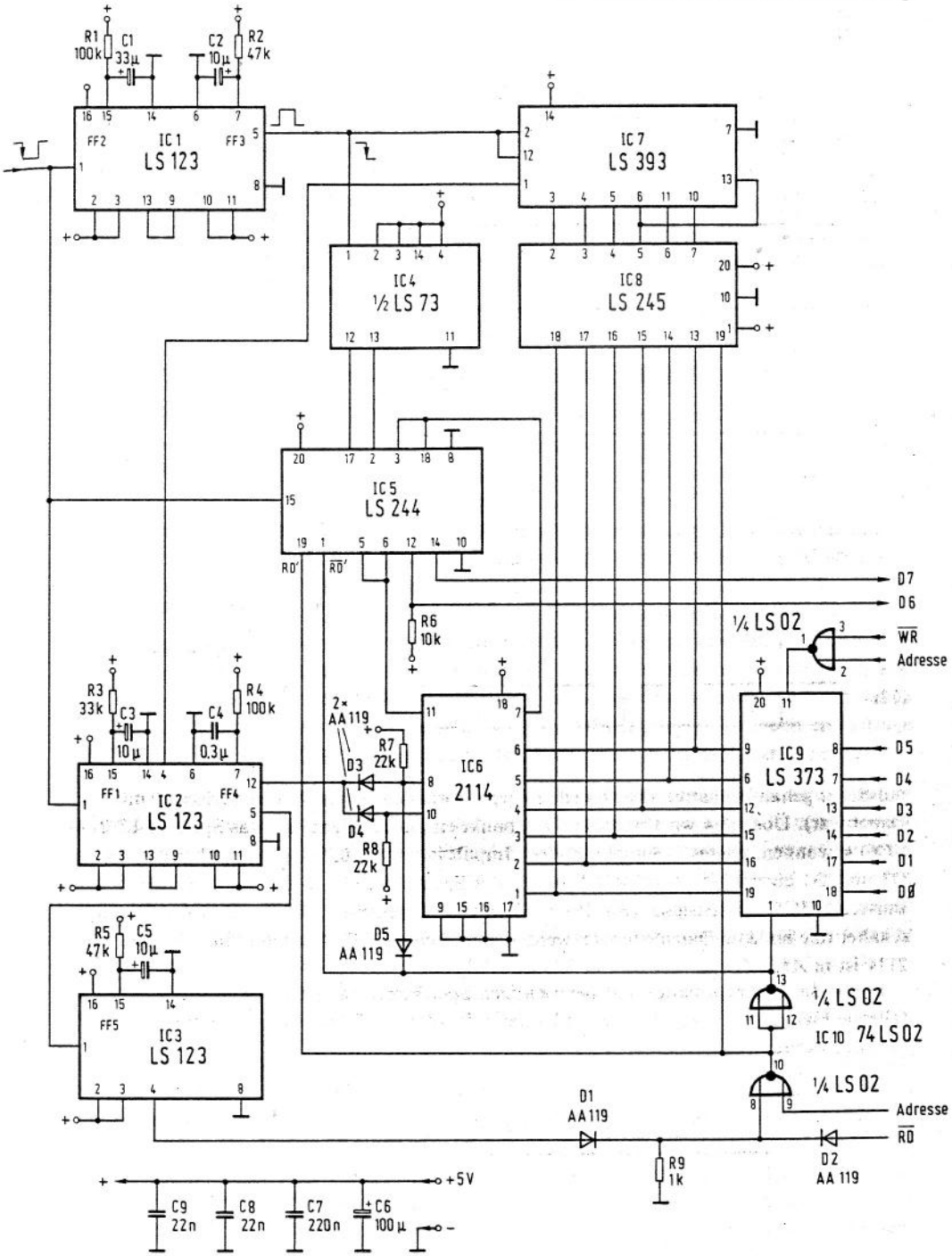


Abb. 5.3 Schaltbild der Dekoder-Platine

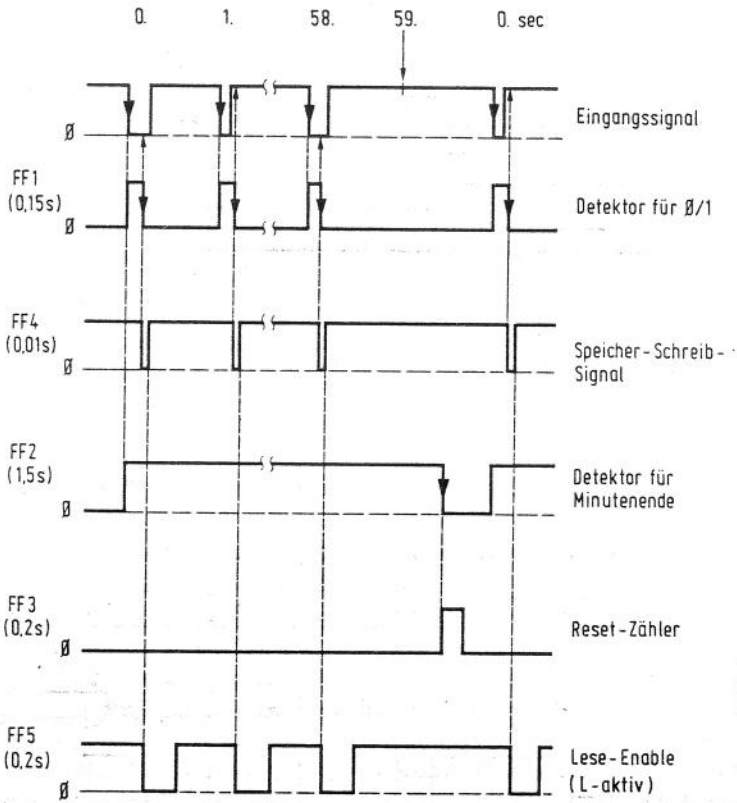


Abb. 5.4 Impulsdiagramm zum Dekoder

pulsdauer gehandelt hatte), oder bereits L-Pegel vorhanden (wenn es eine logische 0 mit 0,1 s Dauer war). Über FF4 wird zu diesem Zeitpunkt ein Schreibsignal für das Speicher-IC 2114 (IC6) gewonnen. Dieses L-Signal mit einer Impulsdauer von 0,01 s gelangt über die Dioden D3 und D4 an den $\overline{CS}$- bzw. $\overline{WR}$-Eingang des Speichers 2114. Gleichzeitig wird über den Bustreiber IC5 das Zeitsignal (Pin 15/Pin 5) auf die Datenleitung D4 des Speichers durchgeschaltet (die anderen Datenleitungen werden nicht benützt). Das Anschlußbild des Speichers 2114 ist in *Abb. 5.5* aufgeführt.

Damit die Zeitinformation auf der richtigen Speicheradresse gefunden wird, schaltet die fallende Flanke des Q-Ausgangs von FF1 den 8-Bit-Zähler IC7 (von dem allerdings nur 6 Bit benötigt werden) weiter.

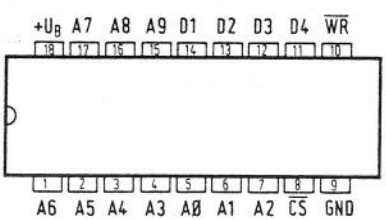


Abb. 5.5 Anschlußbild des 1 K x 4-Speichers 2114

FF2 ist für eine Impulsdauer von 1,5 s dimensioniert. Das bewirkt, daß sein Ausgang dauernd erneut getriggert wird, so daß er durchgehend H-Pegel führt. Das ändert sich erst mit dem Sekundentakt 59. Da dieser Impuls ausgelassen wird, kann FF2 zurückkippen. Dadurch wird FF3 getriggert und setzt den 8-Bit-Zähler auf Null. Auf diesem Wege läßt sich also der Minutenbeginn erkennen.

Da die Schaltung zum gerade betrachteten Zeitpunkt im Speicherbetrieb arbeitet, ist der Bustreiber IC8 über die Steuerleitung (Pin 19) so geschaltet, daß der Zählerstand von IC7 an den Adreßanschlüssen des Speichers IC6 anliegt. Dabei darf nicht verwirren, daß das niederwertigste Bit des Zählers (IC7 bzw. IC8) nicht mit der Adreßleitung A0 des Speichers verbunden ist (für die übrigen Leitungen gilt sinngemäß das gleiche). Auf diese Weise läßt sich ein einfacheres Platinen-Layout erreichen. Für die Funktion des Speichers ist es bedeutungslos, ob die Datenbits 1 ... 59 auf „seinen" Adressen 1 ... 59 oder auf irgendwelchen anderen Adressen liegen, wenn nur die Zuordnung bei Schreiben und Lesen in gleicher Weise erfolgt.

Mit IC 4 hat es eine besondere Bewandtnis. Da beim Auslesen stets die komplette Zeitinformation benötigt wird, man andererseits zum Auslesen aber nicht jedesmal das Minutenende abwarten will (vorher ist die Zeitinformation ja noch unvollständig bzw. fehlerhaft), werden stets zwei aufeinanderfolgende Minuten abgespeichert. Die weiter zurückliegende Information ist dabei stets vollständig und für den aktuellen Zeitpunkt gültig. Das Umschalten zwischen den hierfür nötigen zwei Speicherbereichen besorgt das bistabile Flipflop IC4. Es wird vom Minutenpuls (aus FF3) umgeschaltet. Über IC5 (Pin 17/Pin 3) gelangt das Ausgangssignal von IC4 auf die Adreßleitung A2 des Speichers. An einem Beispiel sei die Funktionsweise verdeutlicht. Die zurückliegende Minuteninformation liege auf den Adressen 100 ... 159, die aktuelle Information wird auf die Adressen 0 ... 59 eingeschrieben. Nach dem Minutenende ist es gerade umgekehrt. Nun wird auf 100 ... 159 eingeschrieben, während die vollständige Information bei 0 ... 59 zur Verfügung steht (aus den oben erläuterten Gründen und infolge der binären Adressenkodierung ist die tatsächlich verwendete Adreßlage komplizierter).

Ein Auslesen des Speichers ist nur erlaubt, wenn gerade nicht eingeschrieben wird. Um hier Kollisionen zu vermeiden, erzeugt FF5 nach dem Einschreiben der Daten in den Speicher (IC6) einen Impuls von 0,2 s Dauer. Während dieser Zeit ist ein Lesezugriff vom Computer her erlaubt. In der übrigen Zeit ist ein solcher Zugriff blockiert. Dies geschieht, indem über die Diode D1 ein H-Pegel an Pin 8 des NOR-Gatters anliegt, so daß dieser Eingang über die $\overline{RD}$-Leitung nicht auf L-Pegel gezogen werden kann. Damit der Computer die Blockierung erkennt, wird bei erfolgreichem Lesezugriff über IC5 (Pin 8/Pin 12) L-Pegel auf die Datenleitung D6 geschaltet. War der Zugriff ungültig, so wird auf D6 H-Pegel gelesen.

Vor dem Auslesen muß erst festgelegt werden, welches Zeitbit erfaßt werden soll. Hierzu wird auf die Dekoderadresse die Nummer des gewünschten Bits geschrieben (eine Zahl zwischen 1 und 59). Sie wird in IC9 zwischengespeichert. Bei einem anschließend durchgeführten (nicht blockierten) Lesebefehl auf der Dekoderadresse liegt diese Zahl an den Adreßeingängen von IC6 an (Pin 1 von IC9 aktiviert). Gleichzeitig ist IC8 über Pin 19 abgeschaltet, so daß kein Doppelzugriff (IC7 und IC9) möglich ist. Auch IC5 schaltet um, dadurch ist der Zeittakt abgekoppelt (Pin15/Pin5), so daß auf D7 (über Pin 6/Pin 14 von IC5) das ausgelesene Bit ansteht. Damit die bereits vollständige Zeitinformation erreicht wird, gelangt das komplementäre Ausgangssignal von IC4 über Pin 2/Pin 18 von IC5 auf die Leitung A2 des Speichers (vergleiche hierzu das oben angeführte Beispiel).

5.3 Aufbau und Inbetriebnahme

Platinenlayout für System I und II finden sich in *Abb. 5.6* bzw. *Abb. 5.7,* der Bestückungsplan mit Stückliste ist in *Abb. 5.8* zu sehen.

Adreßdekodierung und Aufbau erfolgen in üblicher Weise. Für eine erste Inbetriebnahme wird die Platine (ohne Computer) an die Betriebsspannung angeschlossen. Dann führt man das Signal eines DCF-77-Empfängers auf den Eingang (Signalpolarität beachten). Nun überprüft man mit dem Oszilloskop, ob alle Monoflops die erforderlichen Impulse erzeugen. Insbesondere muß auch auf die richtige Impulslänge geachtet werden, da sie für die Dekodierung entscheidend ist. Hier können durch Bauteiletoleranzen u. U. unzulässige Abweichungen auftreten. Neben der Korrektur von Widerstands- und Kondensatorwerten kann auch ein IC-Austausch zum Erfolg führen.

Wird über einen längeren Zeitraum beobachtet, so läßt sich auch der fortschreitende Zählerstand von IC7 verfolgen. Dieser Zählerstand muß auch an den Adreßeingängen des Speichers auftreten.

Sind diese Überprüfungen erfolgreich beendet, so legt man an den $\overline{RD}$- und Adreßeingang L-Pegel (Achtung, offene TTL-Eingänge „empfinden" H-Pegel). Jetzt muß der Zähler vom Speicher abgekoppelt sein, auch die Taktimpulse dürfen nicht mehr an Pin 11 von IC6 erscheinen. D6 muß im Gegensatz zu vorher L-Pegel besitzen.

Haben sich bisher keine Anhaltspunkte für einen Fehler ergeben, kann die Platine mit dem Computer zusammengeschaltet werden (vorher natürlich Betriebsspannung unterbrechen). Dabei können sich u. U. schwer zu beseitigende Störprobleme ergeben. Grund: Hochfrequenz aus dem Computer gelangt in den Zeitzeichen-Empfänger und überlagert die Empfangsfrequenz. Besonders problematisch ist zudem die fünfte Oberwelle der Zeilenfrequenz des Fernseh-Monitors, sie liegt bei 78 kHz und damit in unangenehmer Nähe der Empfangsfrequenz. Abhilfe schafft in der Regel eine möglichst große räumliche Entkopplung der Geräte.

Nach Inbetriebnahme muß man mindestens zwei Minuten warten, bis sicher eine vollständige Zeitinformation gespeichert ist. Diese Information läßt sich mit dem folgenden Programm auslesen:

```
100 FOR N = 1 TO 59
200 POKE Adresse, N
300 PRINT N;" – "; PEEK Adresse
400 NEXT N
```

Bei der Auswertung sollte man sich allerdings die erhaltenen Werte in binärer Form aufschreiben, da ja nur Bit 7 von Bedeutung ist. Vorsicht, wenn Bit 6 logisch 1 ist: dann war die Auslesung ungültig. Durch Vergleich mit dem aktuellen Datum bzw. der Uhrzeit läßt sich allerdings schon erkennen, ob die Schaltung im wesentlichen richtig funktioniert. Für den endgültigen Test geeigneter ist allerdings das in Kaptiel 8.7 vorgestellte Maschinenprogramm.

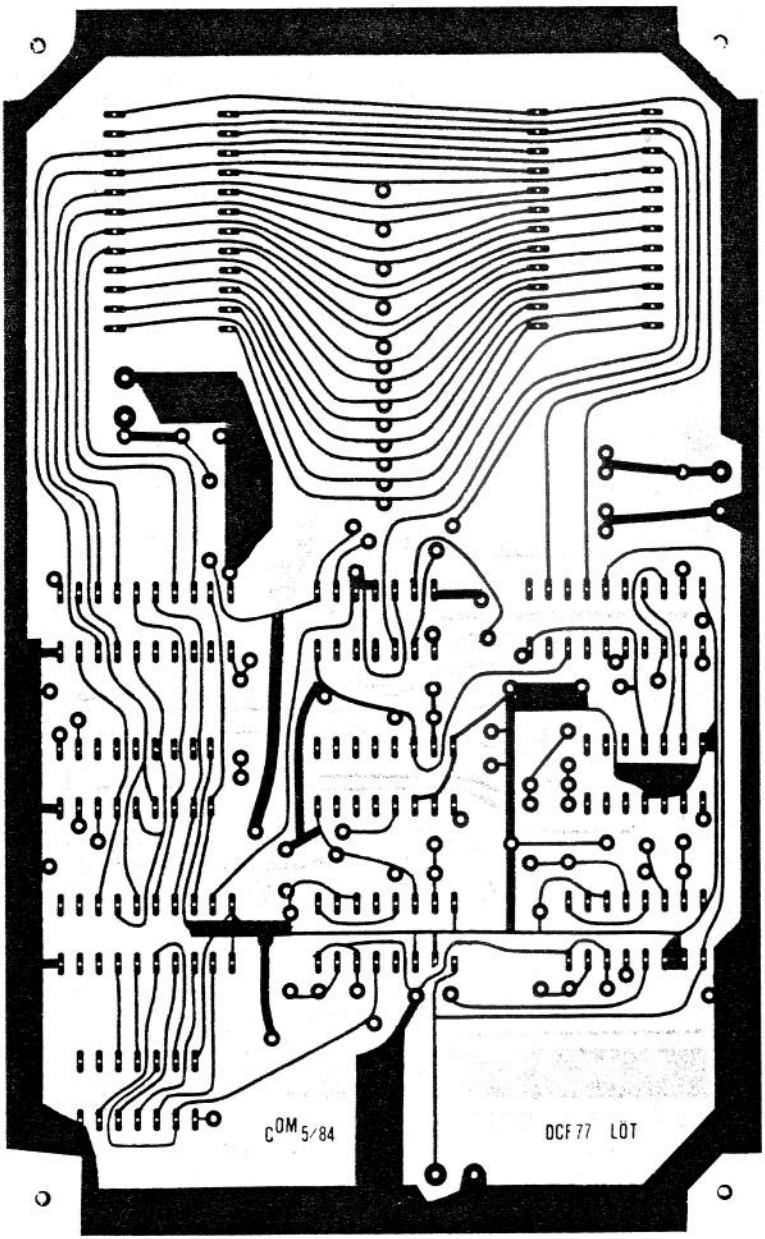


Abb. 5.6a Platinenlayout des Dekoders (System I), Lötseite

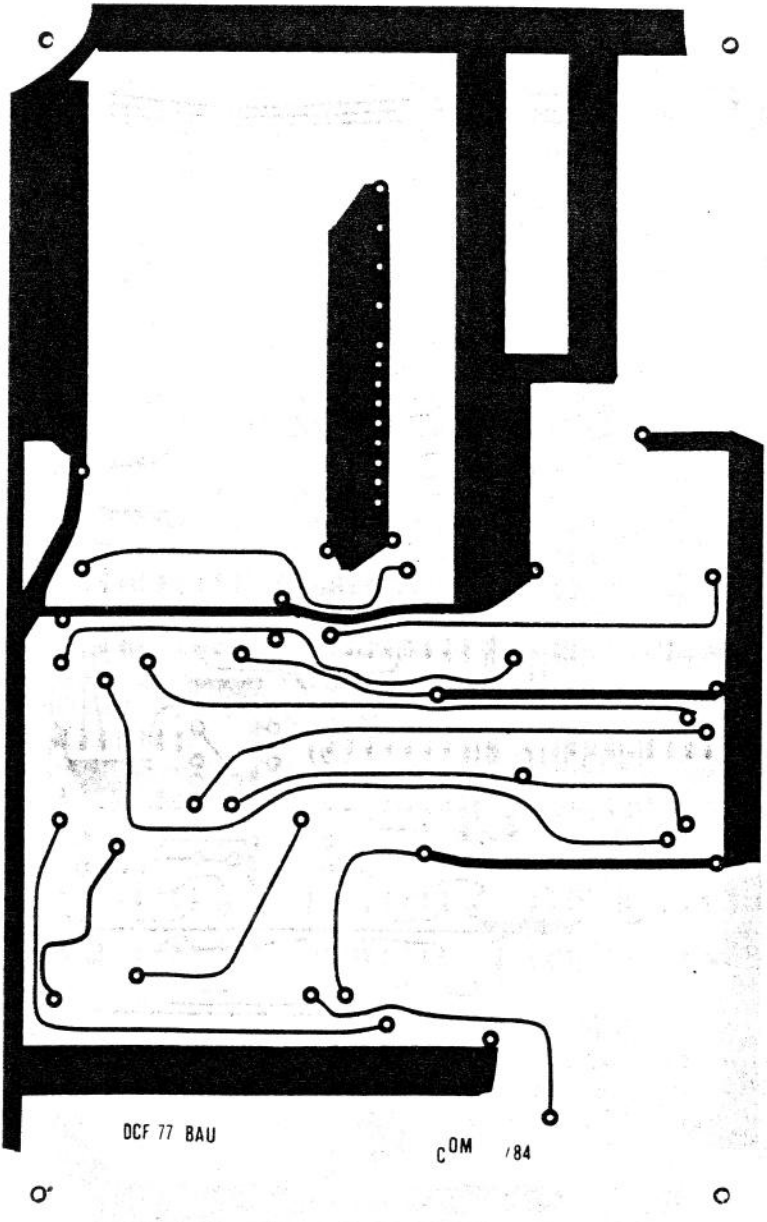


Abb. 5.6b Platinenlayout des Dekoders (System I), Bestückungsseite

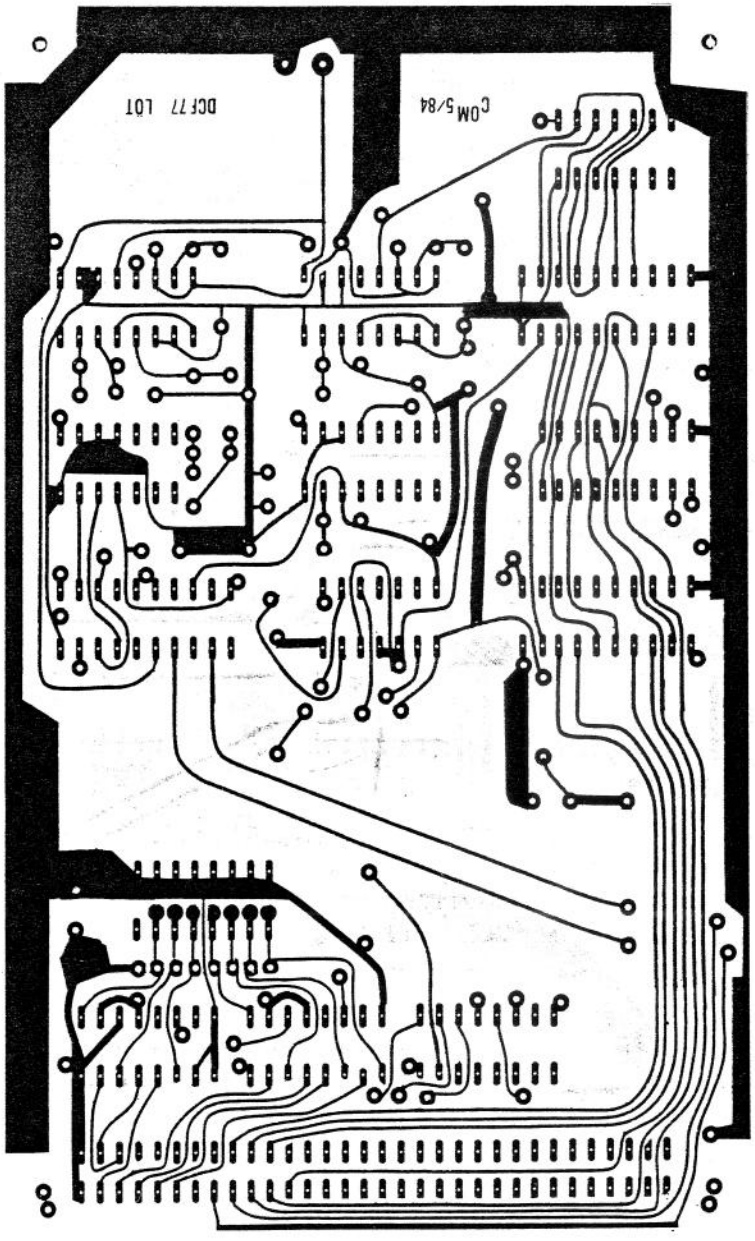


Abb. 5.7a Platinenlayout des Dekoders (System II), Lötseite

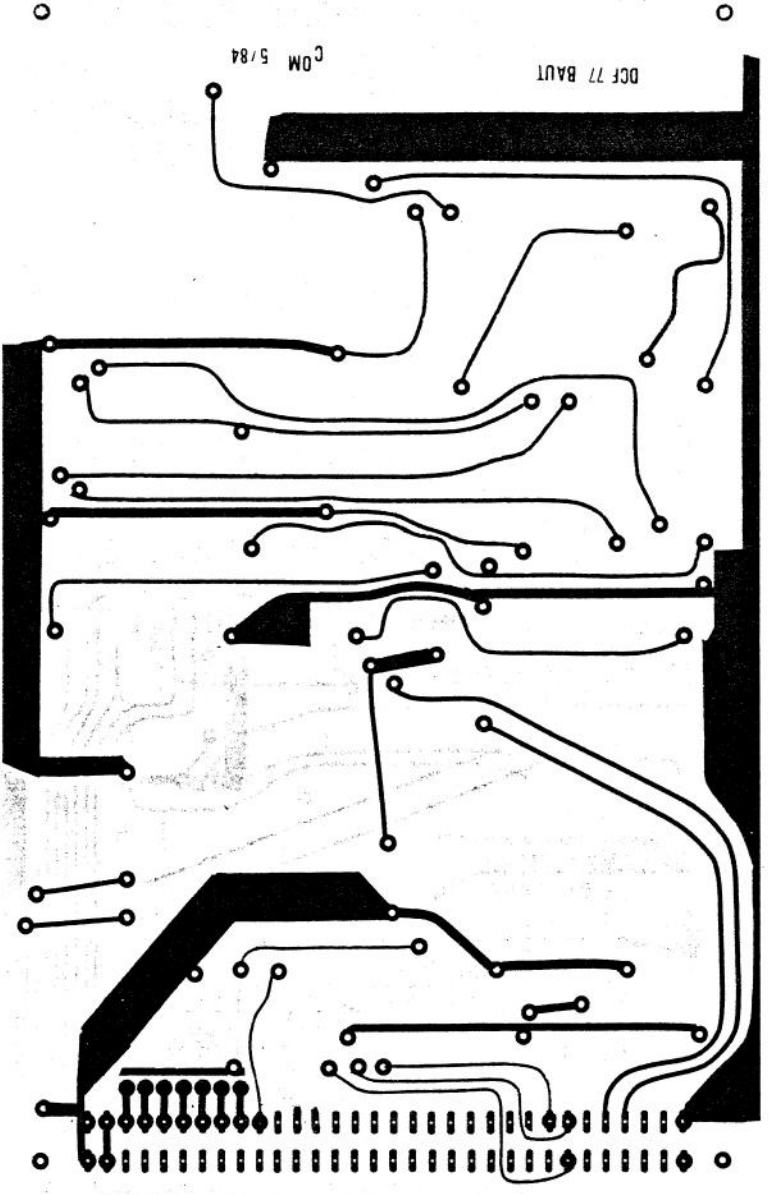


Abb. 5.7b Platinenlayout des Dekoders (System II), Bestückungsseite

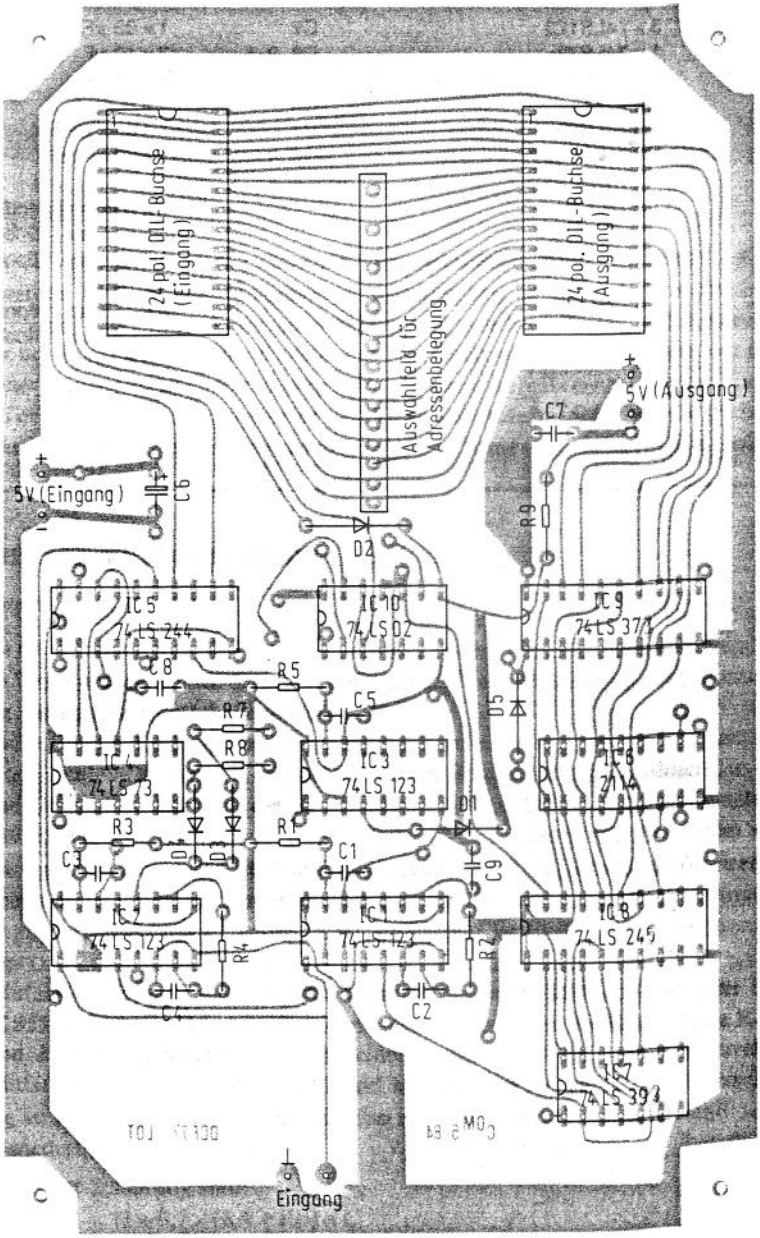
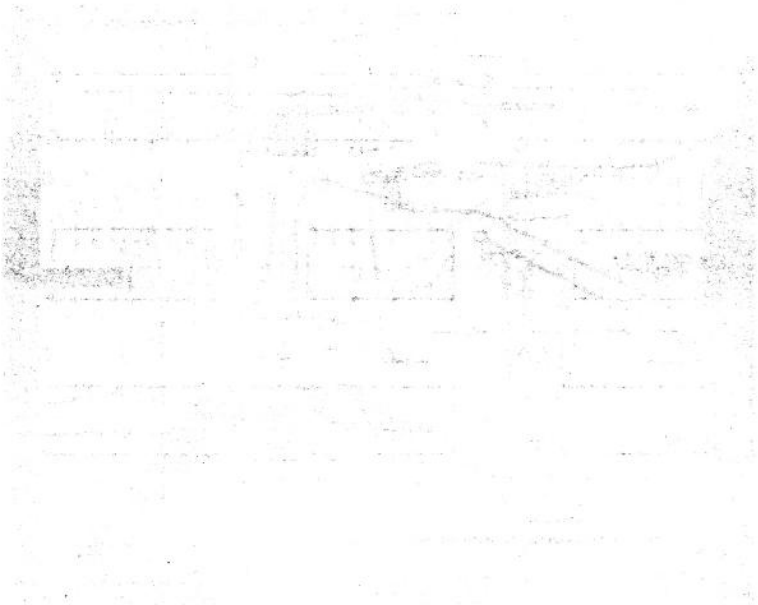


Abb. 5.8 Bestückungsplan (DCF-Dekoder)

Stückliste zum DCF-77-Dekoder

IC1 ... 3	3 x 74LS123	C1	1 x 33 µF
IC4	1 x 74LS73	C2, 3, 5	3 x 10 µF
IC5	1 x 74LS244	C4	1 x 0,3 µF
IC6	1 x 2114	C6	1 x 100 µF
IC7	1 x 74LS393	C7	1 x 0,22 µF
IC8	1 x 74LS245	C8, 9	2 x 22 nF
IC9	1 x 74LS373		
IC10	1 x 74LS02		2 x IC-Fass. 24polig
			3 x IC-Fass. 20polig
Di1 ... 5	5 x AA 119 o. ä. Ge-Diode		1 x IC-Fass. 18polig
			3 x IC-Fass. 16polig
R1, 4	2 x 100 kΩ		3 x IC-Fass. 14polig
R2, 5	2 x 47 kΩ		Steckstifte
R3	1 x 33 kΩ		
R6	1 x 10 kΩ		
R7, 8	2 x 22 kΩ		
R9	1 x 1 kΩ		



6 Tonerzeugung (Hardware)

6.1 Allgemeines

In Kapitel 8.2 wird erläutert, wie Töne mit Hilfe eines Ports rein softwaremäßig erzeugt und variiert werden können. Diese Methode ist sehr flexibel, da sämtliche Parameter (Impulsbreite, Frequenz usw.) über ein Programm eingestellt werden können. Auf der anderen Seite wird aber der Rechner dadurch vollständig in Anspruch genommen. Das hat zur Folge, daß der Rechner während der Tonerzeugung keine anderen Aufgaben übernehmen kann. Nachdem ein Tonprogramm vorzugsweise im FAST-Modus abläuft, fehlt auch die Bildschirmausgabe während dieser Zeit. Eine hardwareorientierte Tonerzeugung vermeidet solche Nachteile.

6.2 Das Schaltungsprinzip

6.2.1 Grundlagen

Bei einem Oszillator erfolgt die Frequenzänderung, indem ein frequenzbestimmendes Bauteil in seinem Wert verändert wird. Soll der Computer die Frequenz beeinflussen, so muß er dieses Bauteil in seinem Wert umschalten *(Abb. 6.1)*.

Über acht Datenbusleitungen wären somit zunächst acht verschiedene Frequenzen schaltbar. Werden Bauteile verwendet, deren Wert sich bei Parallelschaltung addiert, so kann man durch eine Gewichtung der über die Datenleitungen D0 ... D7 geschalteten Bauelemente im Verhältnis 1:2:4:8:16:32:64:128 erreichen, daß der Bauteilewert proportional zum vom Computer ausgegebenen Datenwert ist. Diese Bedingung wird z. B. für Kondensatoren erfüllt.

Falls bei der verwendeten Oszillatorschaltung die Frequenz oder die Schwingungsdauer ebenfalls proportional zum Bauteilewert ist, so besteht letztlich auch Proportionalität zwischen dem Datenwert und der Frequenz bzw. Schwingungsdauer.

Eine einfache und doch gut geeignete Oszillatorschaltung läßt sich mit dem bekannten Timer-IC 555 aufbauen *(Abb. 6.2)*. Der Kondensator wird über die Widerstände RA und RB aufgeladen. Erreicht die Spannung an Pin 6 einen bestimmten Wert, so wird C durch einen internen Entladetransistor (zwischen Pin 7 und Masse) über RB entladen. Unterschreitet die Spannung an Pin 6 einen bestimmten Wert, so sperrt der Entladetransistor und der gesamte Vorgang wiederholt sich.

Die Daten der Schaltung sind:

Aufladezeit	$t1 = 0{,}7 \times (RA + RB) \times C$
Entladezeit	$t2 = 0{,}7 \times RB \times C$
Periodendauer	$t = t1 + t2 = 0{,}7 \times (RA + 2 \times RB) \times C$
Frequenz	$f = 1/T = 1{,}44/(RA + 2 \times RB) \times C$
Tastverhältnis	$Tv = t2/t1 = RB/(RA + RB)$

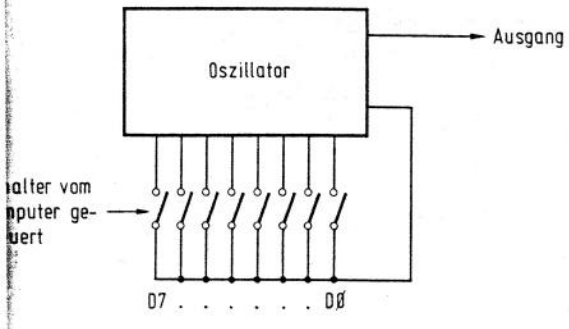


Abb. 6.1 Computergesteuerter Oszillator

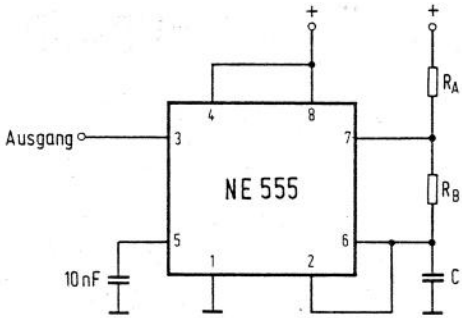


Abb. 6.2 Oszillator mit dem Timer NE 555

Um eine Frequenzvariation zu erreichen, können also RA, RB und C umgeschaltet werden.

Erwähnt sei außerdem, daß in letzter Zeit verstärkt spezielle Tonsynthese-ICs auf den Markt kommen, bei denen eine Vielzahl von Klangeffekten durch entsprechende Programmierung erreicht werden kann. Allerdings sind diese ICs zum Teil noch recht teuer und außerdem nicht überall erhältlich.

6.2.2 Kondensator-Umschaltung

Macht man in der Oszillatorschaltung aus Abb.6.2 den Kondensator schaltbar (*Abb.6.3*), so gilt:

$C_{ges} = C1 + C2 + C3 + \ldots$

Damit wird die Schwingungsdauer T proportional zu C_{ges} und folglich auch proportional zum 8-Bit-Datenwert N. Das bedeutet andererseits, daß die Frequenz umgekehrt proportional zu N ist.

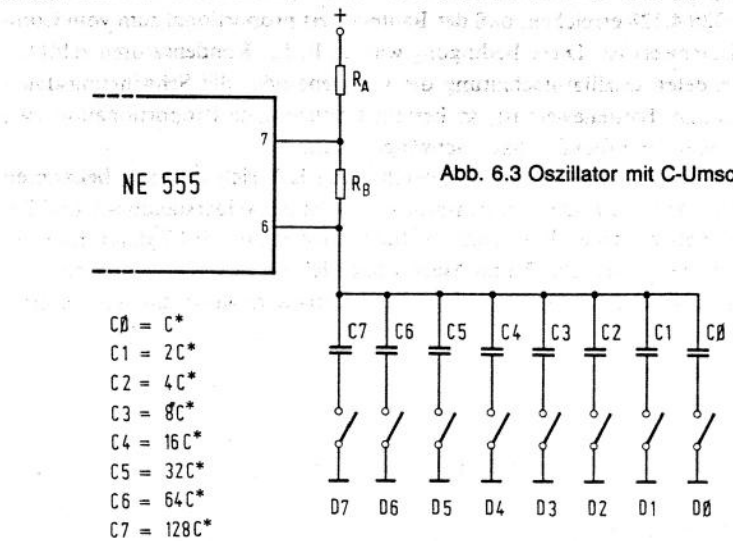


Abb. 6.3 Oszillator mit C-Umschaltung

Demnach sind die Frequenzabstände für kleine Werte von N groß (maximale Frequenzänderung 2 : 1), für große Werte von N dagegen klein (minimale Frequenzänderung 255:254 = 1,004:1). Vorteilhaft ist, daß bei einer Frequenzumschaltung das Tastverhältnis nicht verändert wird. Wählt man insbesondere RB sehr viel größer als RA, so ist das Tastverhältnis ungefähr 1:1.

Durch entsprechende Dimensionierung von RA und RB kann der überstrichene Frequenzbereich wie gewünscht gewählt werden. Nachteilig ist, daß Kondensatoren in der Regel nicht in ähnlich feiner Abstufung wie Widerstände erhältlich sind. Für das nachfolgende Programm zur Frequenzberechnung wurden daher Kondensatoren aus der E-6-Reihe gewählt. Die dadurch bedingten Ungenauigkeiten können normalerweise in Kauf genommen werden.

Das Programm berechnet die höchste und tiefste Frequenz sowie die zu den einzelnen Datenwerten gehörenden Frequenzwerte.

```
  1 REM FREQUENZ-BERECHNUNG
  2 REM TONERZEUGUNG MIT 555
  3 REM ANGABE VON FMIN/FMAX
  4 REM
  5 REM
 10 DIM A(8)
 20 DIM B(8)
 30 LET Z=1
 40 LET K=0
 50 LET A(1)=1.0
 60 LET A(2)=2.2
 70 LET A(3)=4.7
 80 LET A(4)=10
 90 LET A(5)=22
100 LET A(6)=47
110 LET A(7)=100
120 LET A(8)=220
200 PRINT "KONDWERTE IN NANO-FA
RAD:"
300 FOR N=1 TO 8
400 PRINT A(N);" ";
500 LET A(N)=A(N)*10**-9
600 NEXT N
700 PRINT
800 PRINT
900 PRINT "R(A) = ";
1000 INPUT RA
1100 PRINT RA;" K-OHM"
1200 LET RA=1000*RA
1300 PRINT "R(B) = ";
1400 INPUT RB
1500 PRINT RB;" K-OHM"
1600 LET RB=1000*RB
1605 LET C=0
1610 FOR N=1 TO 8
1615 LET C=C+A(N)
1616 NEXT N
1617 PRINT
1620 LET F=.001*INT (.5+1.44/((R
A+2*RB)*C))
1625 PRINT "F(MIN) = ";F;" KHZ"
1630 LET F=.001*INT (.5+1.44/((R
A+2*RB)*A(1)))
1635 PRINT
1640 PRINT "F(MAX) = ";F;" KHZ"
1698 FOR N=1 TO 150
1699 NEXT N
1700 CLS
1800 FOR G=0 TO 6
1801 FOR M=1 TO 40
1900 LET J=G*40+M
2000 FOR N=8 TO 1 STEP -1
2100 LET K=J/2
2200 LET B(N)=A(9-N)*(J-2*INT K)
2300 LET J=INT K
2400 NEXT N
2500 LET C=0
2600 FOR N=1 TO 8
2700 LET C=C+B(N)
2800 NEXT N
2900 LET F=.001*INT (.5+1.44/((R
A+2*RB)*C))
3000 LET X=M
3100 LET Y=0
3200 IF M<=20 THEN GOTO 3500
3300 LET X=M-20
3400 LET Y=17
3500 LET Z=4
3600 IF F<100 THEN LET Z=5
3700 IF F<10 THEN LET Z=6
3800 IF F<.1 THEN LET Z=7
3900 PRINT AT X,Y;G*40+M;AT X,Y+
Z;F; AT X,Y+12;"KHZ"
4100 NEXT M
4200 FOR F=0 TO 30
4300 NEXT F
4400 CLS
4500 NEXT G
```

Prog. 4

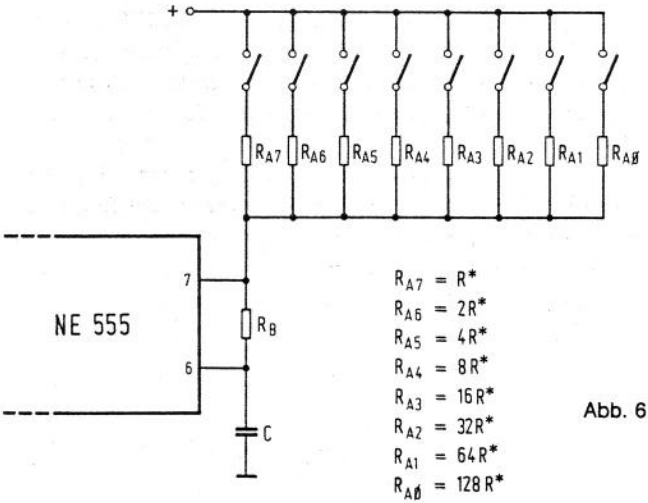


Abb. 6.4 Oszillator mit R-Umschaltung

6.2.3 Widerstands-Umschaltung

Wird der Widerstand RA (RB verhält sich sinngemäß) aus Abb. 6.2 schaltbar gemacht *(Abb. 6.4)*, so ergeben sich zunächst Probleme, da für den Gesamtwiderstand gilt:

$$1/RA_{ges} = 1/RA0 + 1/RA1 + 1/RA2 + \ldots$$

Additiv verhalten sich also nicht die Widerstände, sondern ihre Kehrwerte (die sogenannte Leitfähigkeit). Zum größten Widerstand gehört dabei die kleinste Leitfähigkeit. Folglich muß die Datenleitung D0 den höchsten Widerstandswert schalten, D7 den niedrigsten Widerstandswert (höchste Leitfähigkeit). Die Zuordnung der Bauteilewerte zu den Datenleitungen verläuft also umgekehrt wie bei der Kondensatorumschaltung.

Damit ist keine Proportionalität zwischen Datenwert N und der Periodendauer bzw. Frequenz gegeben. Nachteilig ist außerdem, daß das Tastverhältnis nicht konstant bleibt, sondern vom eingestellten Datenwert abhängt. Andererseits aber sind Widerstände in feinerer Abstufung erhältlich als Kondensatoren, so daß die erforderlichen Werte exakter getroffen werden können.

Bei dieser Schaltung gehört zu kleinen Werten von N eine tiefe Frequenz mit großen Frequenzabständen, zu hohen Werten von N eine hohe Frequenz mit kleinen Frequenzabständen. Das Schaltverhalten ist also gegenläufig zur C-Umschaltung. Das Frequenzverhältnis für zwei aufeinanderfolgende Töne läßt sich nun nicht mehr so einfach angeben wie bei der C-Umschaltung, da jetzt auch die Werte von RB und C mit eingehen.

Maximale und minimale Frequenz lassen sich wie folgt berechnen:

$$f_{max} = 1{,}44/(RA0 + 2 \times RB) \times C$$
$$f_{min} = 1{,}44/(RA_{ges} + 2 \times RB) \times C$$

Zum Umschalten der Widerstände eignen sich Relais aus bereits diskutierten Gründen nicht. Deshalb wird der bereits besprochene CMOS-Schalter 4066 verwendet. Dabei muß berücksichtigt werden, daß der EIN-Widerstand der einzelnen Schalter rund 90 Ω beträgt. Da-

mit dieser Wert nicht wesentlich in die Frequenz der Tongeneratoren eingeht, sollten die Widerstände RA und RB (bzw. die entsprechenden Gesamtwiderstände) groß gegenüber dem EIN-Widerstand der Schalter sein.

Die Steuereingänge der Schalter sind TTL-kompatibel und können direkt mit einem Ausgabe-Port verbunden werden.

6.3 Gesamtschaltung

6.3.1 Tongenerator mit C-Umschaltung

In *Abb.6.5* ist das Schaltbild des Tongenerators gezeigt. Er enthält drei identische, unabhängige Oszillatoren, so daß durch Mischung vielfältige Klangeindrücke erzeugt werden können. Selbstverständlich ist auch hier wieder ein Teilausbau möglich.

Die acht parallelen Kondensatoren werden über die CMOS-Schalter IC2 bzw. IC3 in Abhängigkeit von dem an der Eingangsbuchse DIL1 angelegten Datenwort nach Masse geschal-

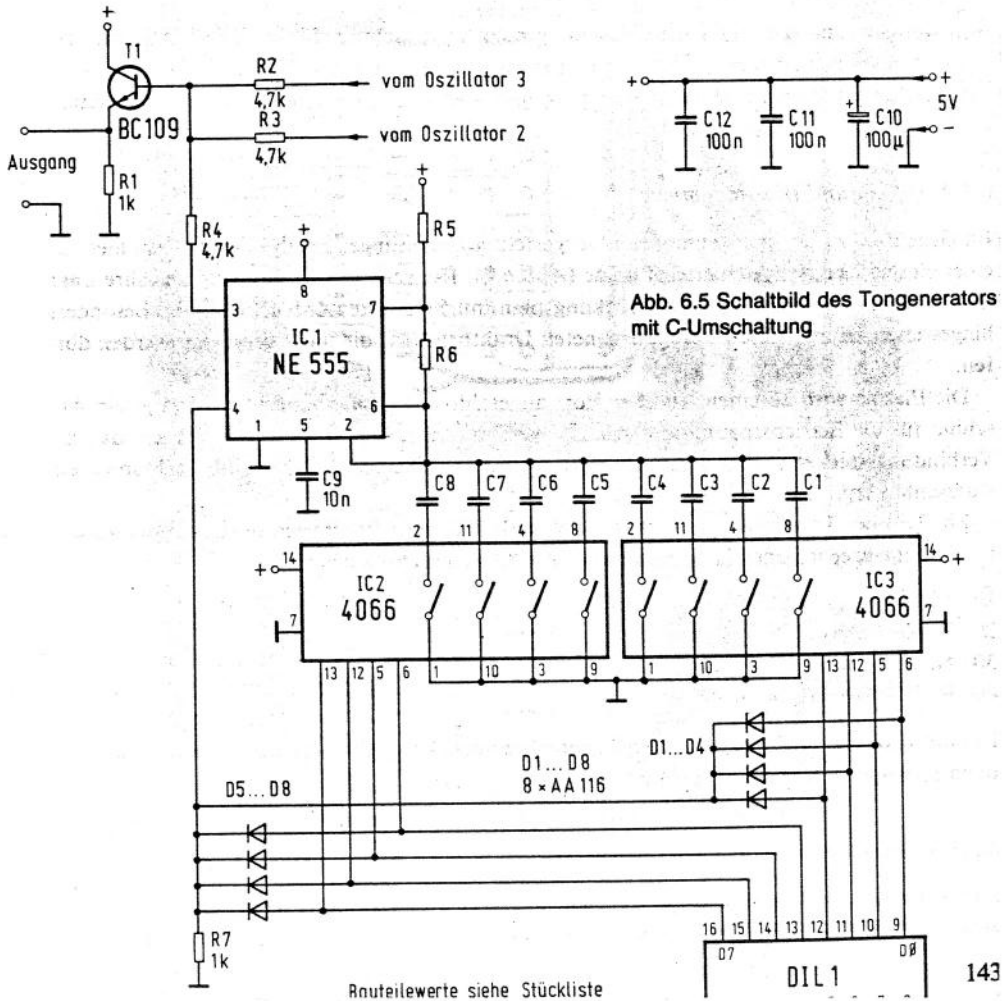


Abb. 6.5 Schaltbild des Tongenerators mit C-Umschaltung

tet. Dadurch ergeben sich $2^8 - 1 = 255$ verschiedene Werte für C_{ges} und damit 255 verschiedene Frequenzen.

Wird der Datenwert Null angelegt, so sind alle Kondensatoren abgeschaltet. Infolge der unvermeidlichen Schaltkapazitäten besteht aber immer noch ein „Kondensator", so daß die Schaltung auf einer hohen Frequenz schwingen würde. Darum wird bei dem Eingabewert Null der Oszillator über Pin 4 von IC1 abgeschaltet. Bewirkt wird das über ein Diodengatter D1 ... D8. Ist eine oder mehrere der Datenleitungen auf H-Pegel, dann erhält auch Pin 4 einen H-Pegel (ODER-Verknüpfung), so daß der Oszillator arbeitet. Erst wenn alle Datenleitungen 0 V führen, liegen auch an Pin 4 (Reset-Eingang des 555) 0 V an, wodurch er abgeschaltet wird.

Die Frequenzen lassen sich mit dem bereits vorgestellten BASIC-Programm berechnen, allerdings ergeben sich infolge von Bauteileltoleranzen gewisse Abweichungen. Ein Frequenzmesser ist also ein nützliches Meßgerät zum Abgleich und bei der Programmerstellung.

Der Ausgang des Oszillators führt über einen Entkopplungswiderstand auf die Basis des Ausgangstransistors T1. Seine Basis erhält gleichzeitig auch die Signale der beiden anderen auf der Platine befindlichen Oszillatoren, so daß am Ausgang das gemischte Signal abgenommen werden kann. Ist eine solche Mischung nicht erwünscht (z. B. um einzelne Signale getrennt weiterzuverarbeiten), so können sie an den im Bestückungsplan mit einem Stern gekennzeichneten Stellen auch einzeln ausgekoppelt werden. T1 sowie R1 ... R4 entfallen dann.

6.3.2 Aufbau und Inbetriebnahme

Im Gegensatz zu den meisten anderen vorgestellten Schaltungen erfolgt der Aufbau hier auf einer einseitig kupferkaschierten Platine (*Abb.6.6*). Die verwendeten Bauteile und ihre Lage auf der Platine gehen aus dem Bestückungsplan mit Stückliste (*Abb.6.7*) hervor. Besonders hingewiesen sei auf die dort eingezeichneten Drahtbrücken, die nicht vergessen werden dürfen.

Die Platine wird an einen Ausgabe-Port angeschlossen. Dabei muß noch separat ein Anschluß für die Betriebsspannung geschaltet werden. Auch hier ist darauf zu achten, daß die Verbindungsstecker zum Ausgabe-Port richtigherum eingeführt werden, da sich sonst ein Kurzschluß ergibt.

Ein Test des Tongenerators kann mit folgendem kleinen Programm durchgeführt werden (50-Ω-Lautsprecher oder 32-Ω-Postkapsel am Ausgang anschließen).

```
100 FOR N = 0 TO 255
200 POKE Adresse, N
300 NEXT N
400 POKE Adresse, 0
```

Es muß sich eine von hohen zu tiefen Tönen abgestufte Frequenzfolge ergeben, wobei die Frequenzsprünge zu Beginn am größten sind.

6.3.3 Tongenerator mit R-Umschaltung

Abb.6.8 zeigt das Schaltbild dieses Tongenerators. Er ist analog aufgebaut zur Schaltung aus Abb.6.5. Statt Kondensatoren werden die Widerstände R1 ... R8 geschaltet, um die ge-

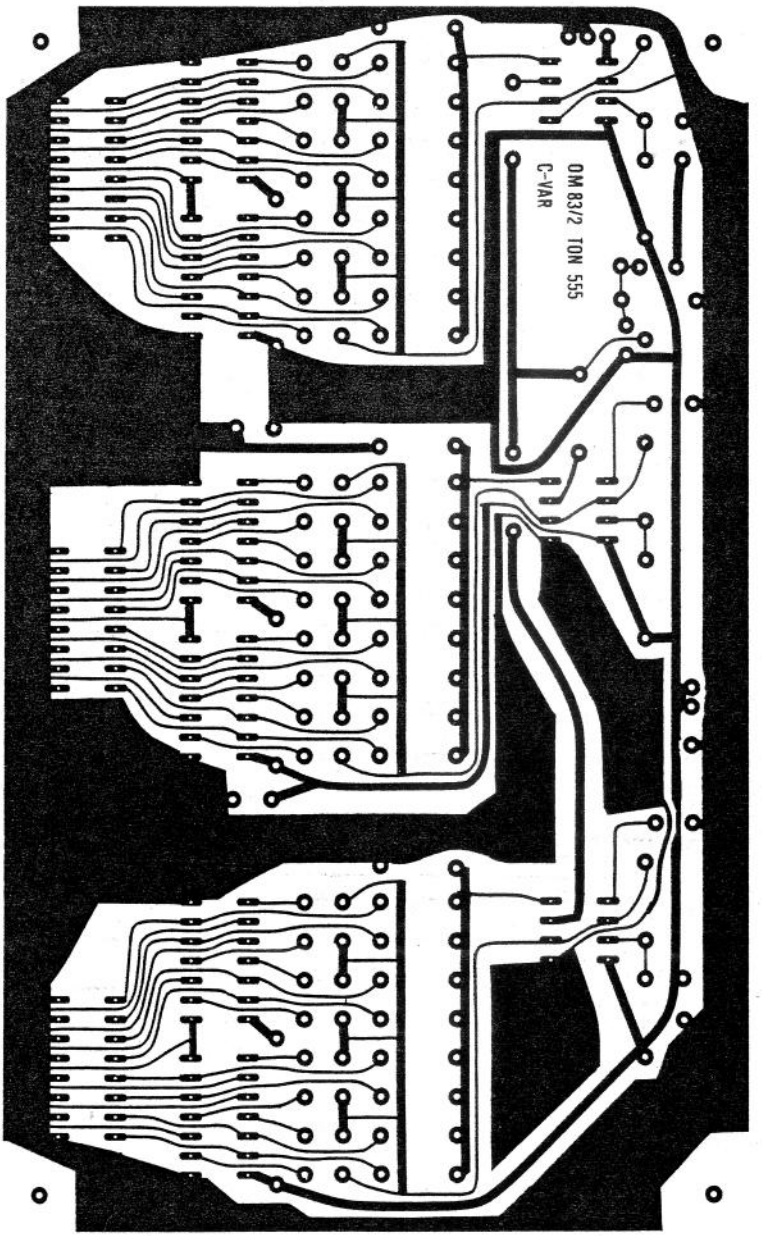


Abb. 6.6 Platinenlayout des Tongenerators (C-Umschaltung), Lötseite

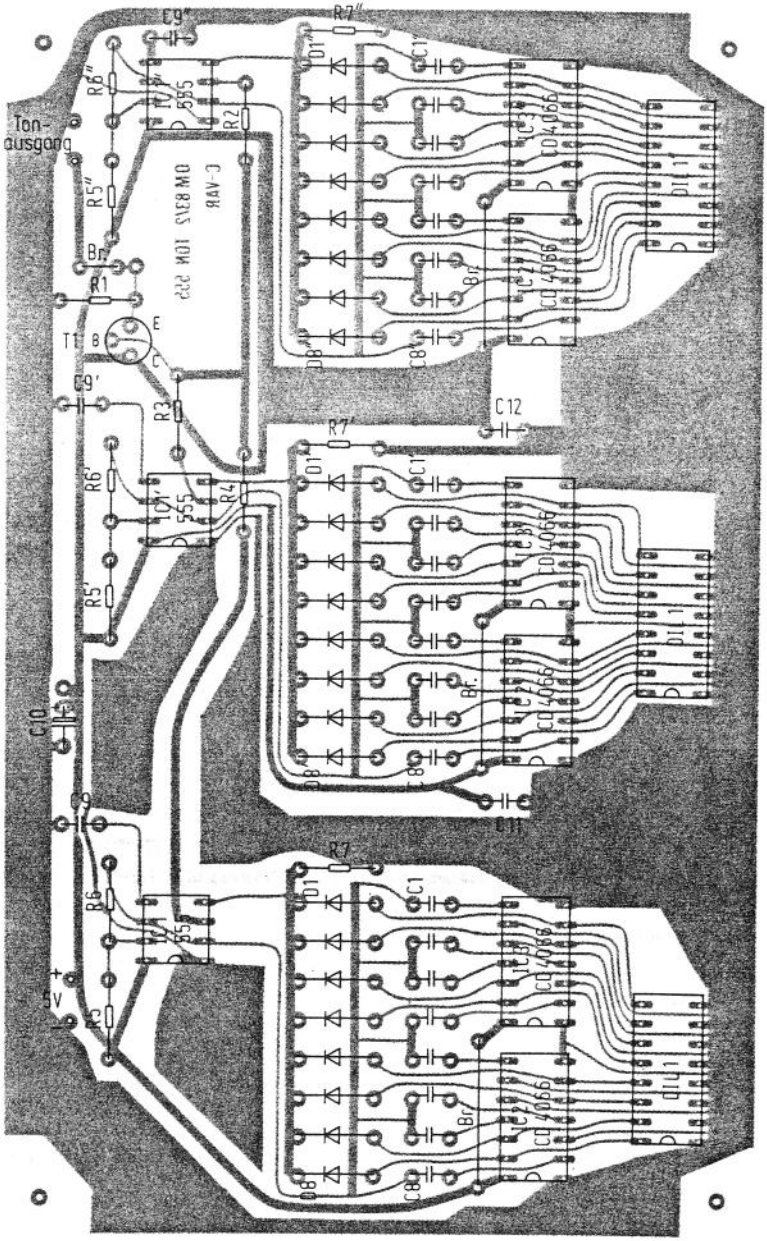


Abb. 6.7 Bestückungsplan (Tongenerator/C-Umschaltung)

Stückliste zum Tongenerator (C-Umschaltung)

T1	1 × BC 109 o. ä.		
R1	1 × 1 kΩ	C11, 12	2 × 0,1 µF
R2 ... 4	3 × 4,7 kΩ		

Lautsprecher 50 Ω oder Postkapsel 32 Ω

Pro Tongenerator werden folgende Bauteile benötigt:

IC1	1 × NE 555	1 × IC-Fass. 16polig
IC2, 3	2 × 4066	2 × IC-Fass. 14polig
D1 ... 8	8 × AA 116 o. ä. Ge-Diode	1 × DIL-Fass. 8polig
C9	1 × 10 nF	
R7	1 × 1 kΩ	
C1 ... 8	8 × Kondensator gemäß Berechnung	
R5, 6	2 × Widerstände gemäß Berechnung	

Beispiel:

C1 ... C8:	1 nF 2,2 nF 4,7 nF 10 nF 22 nF 47 nF 100 nF 220 nF
R5	1 × 1 kΩ
R6	1 × 100 kΩ

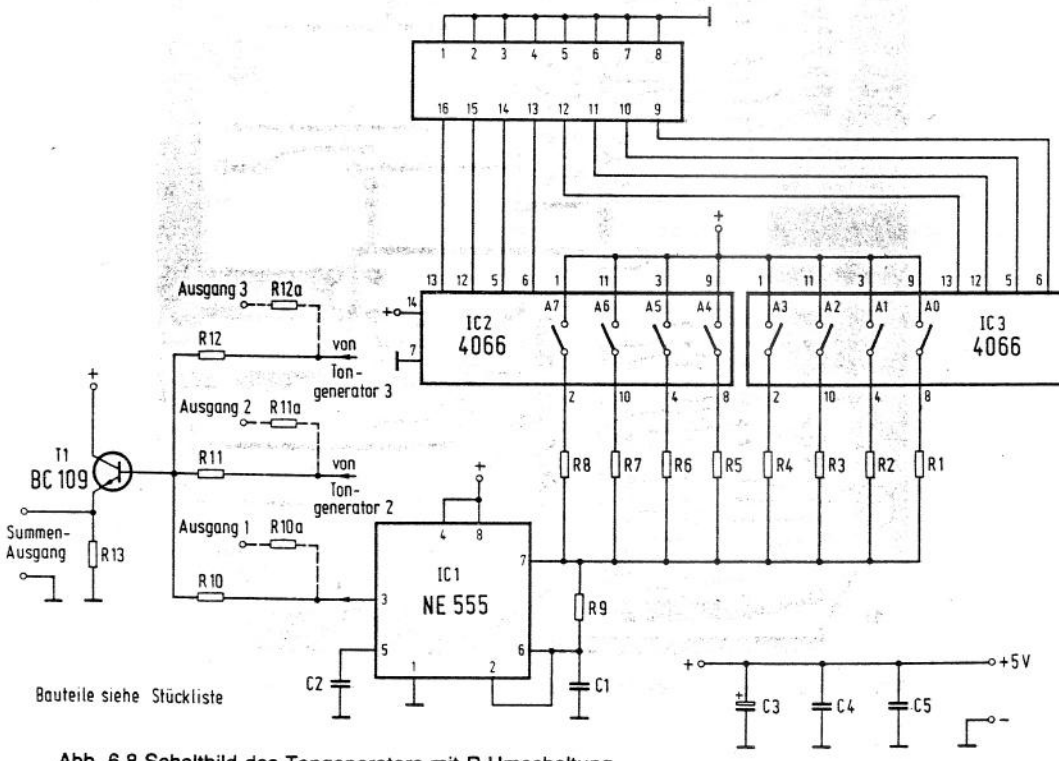


Abb. 6.8 Schaltbild des Tongenerators mit R-Umschaltung

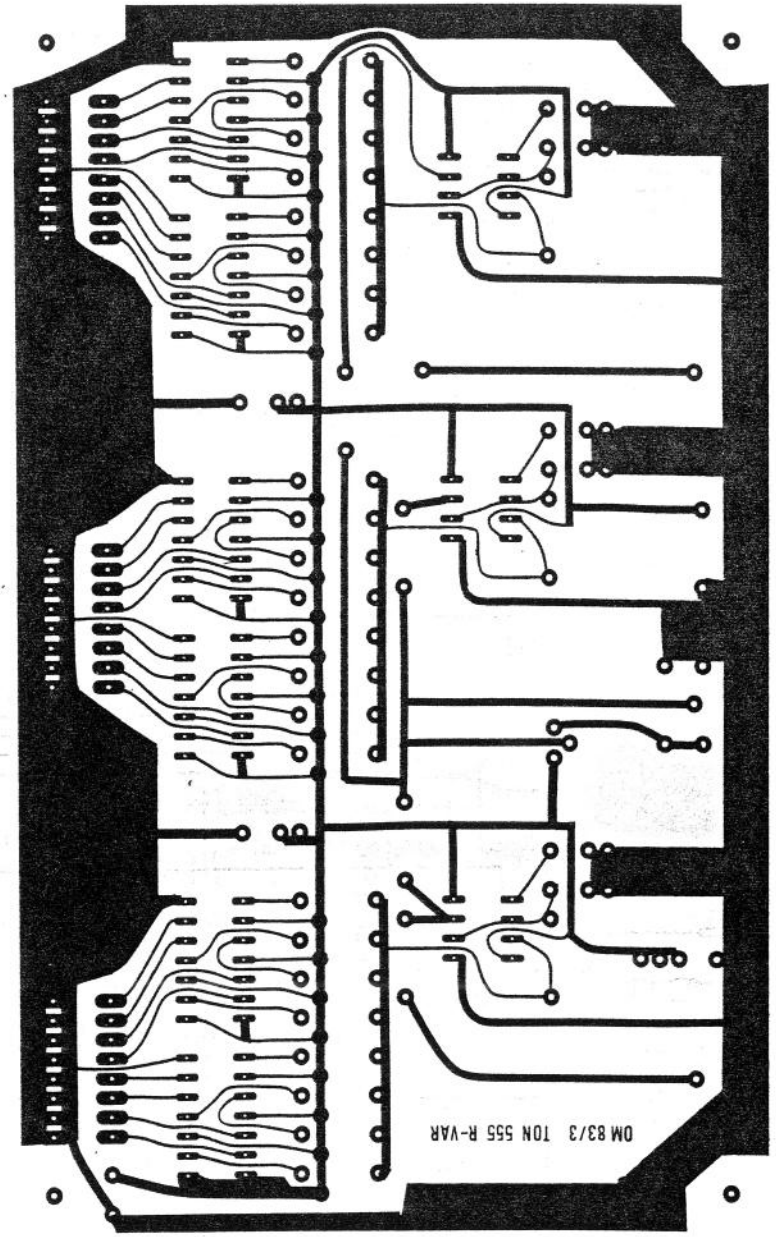


Abb. 6.9 Platinenlayout des Tongenerators (R-Umschaltung), Lötseite

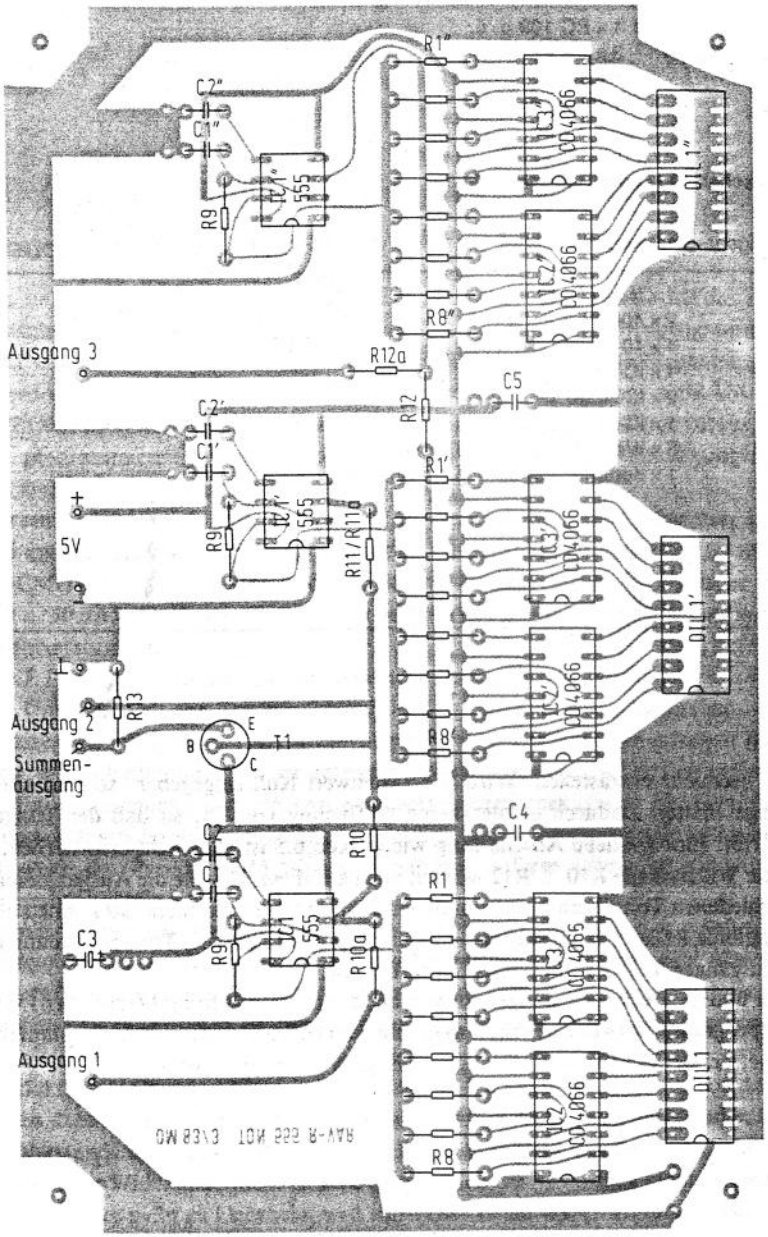


Abb. 6.10 Bestückungsplan (Tongenerator/R-Umschaltung)

Stückliste Tongenerator (R-Umschaltung)

T1	1 x BC 109 o. ä.
R10 (a) ... 12 (a)	3 x 4,7 kΩ
R13	1 x 1 kΩ
C3	1 x 470 µF
C4, 5	2 x 0,1 µF

Lautsprecher 50 Ω oder Postkapsel 32 Ω

Pro Tongenerator werden benötigt:

IC1	1 x NE 555
IC2, 3	2 x 4066
C2	1 x 10 nF
	1 x IC-Fass. 16polig
	2 x IC-Fass. 14polig
	1 x IC-Fass. 8polig
R1 ... 8	8 x Widerstände gemäß Berechnung
C1	1 x Kondensator gemäß Berechnung

Beispiel

R1 ... R8	470 kΩ 220 kΩ 100 kΩ 47 kΩ 22 kΩ 10 kΩ 4,7 kΩ 2,2 kΩ
R9	10 kΩ
C1	10 nF

wünschte Frequenz einzustellen. Wird der Datenwert Null eingegeben, so sind alle Widerstände abgeschaltet. Dadurch erfolgt keine Aufladung von C1, so daß der Tongenerator stumm bleibt. Eine spezielle Abschaltung wie in Abb.6.5 ist also nicht erforderlich.

Über die Widerstände R10 ... R12 werden auch auf dieser Platine die Ausgangssignale von drei verschiedenen Tongeneratoren summiert. Wünscht man das nicht, so können alternativ die Widerstände R10a ... R12a eingesetzt werden. Es stehen die drei Tonsignale dann getrennt an den Ausgängen 1 ... 3 zur weiteren Verarbeitung.

Aufbau und Test erfolgen sinngemäß wie bei der C-Umschaltung beschrieben. Wird dasselbe Testprogramm verwendet, erhält man eine von tiefen zu hohen Tönen verlaufende Frequenzfolge, bei der die Frequenzsprünge zu Beginn am größten sind.

7 Nichtflüchtige Datenspeicher

7.1 Pseudo-ROM

Grundplatine I und „language card" besitzen Steckplätze für statische RAMs des Typs 6116. Unter Umständen taucht der Wunsch auf, diese Speicher für Maschinenprogramme zu nützen. Dabei wird es irgendwann lästig, bei jedem Einschalten der Betriebsspannung die Programme neu vom Band laden zu müssen. Die an sich mögliche Lösung, die RAM-Bausteine durch EPROMs zu ersetzen (es gibt pinkompatible Typen), setzt allerdings voraus, daß man EPROMs programmieren kann. Darüber hinaus ist die Änderung einmal programmierter EPROMs zwar möglich, aber mit Umständen verbunden.

Aus diesem Grunde stellt ein „Pseudo-ROM" eine interessante Lösung dar. Es handelt sich dabei um ein ganz normales statisches RAM, allerdings mit eingebauter Batterie-Pufferung. Da CMOS-RAMs einen sehr niedrigen Stromverbrauch aufweisen (typische Werte liegen bei 10 ... 50 μA), erlauben auch Batterien oder Akkus geringer Kapazität eine ausreichend lange Stromversorgung.

In *Abb. 7.1* ist die elektrische Beschaltung eines solchen Pseudo-ROMs mit dem RAM 6116 zu sehen. Um den niedrigen Stromverbrauch in der Betriebsform „Datenerhaltung" zu sichern, müssen die Pegel an den RAM-Eingängen gewisse Bedingungen erfüllen. Daher sind alle Daten- und Adreßleitungen in der gezeigten Form zu beschalten. Die Leitungen $\overline{CS}$ (chip select), $\overline{OE}$ (output enable) und $\overline{WE}$ (write enable) werden über jeweils 10 kΩ mit $+U_B$ verbunden und dadurch desaktiviert, solange von außen keine anderen Signale anliegen. Dies ist der Fall, wenn das Pseudo-ROM aus der Schaltung entfernt wird oder die ansteuernden Schaltkreise hochohmig sind.

In der $\overline{WE}$-Leitung befindet sich ein Schalter. Wird er geöffnet, kann von außen nicht auf das Pseudo-ROM zugegriffen werden, es ist also schreibgeschützt. Man sollte diesen Schalter auch stets öffnen, bevor man die Betriebsspannung abschaltet, da über Ausschwingungsvorgänge auf den Signalleitungen im Ausschaltmoment stets die Gefahr völlig unkontrollierter Schreibaktivitäten besteht, wodurch die gespeicherten Daten zerstört werden.

Der Akku (besteht aus drei NiCd-Zellen mit 60 mAh Kapazität) wird während des Computer-Betriebs über einen Widerstand von 1 kΩ aus der 5-V-Versorgung aufgeladen. Der Wert dieses Widerstands bestimmt den Ladestrom. Dabei muß man einen Kompromiß schließen. Einerseits soll der Widerstand nicht zu groß sein, so daß die Aufladung nicht zu lange dauert, andererseits darf er nicht zu klein sein, weil sonst die Gefahr der Überladung besteht und der Akku zerstört wird. Am besten paßt man diesen Widerstandswert den jeweiligen eigenen Bedürfnissen an.

Wenn die Stromversorgung durch den Computer entfällt, liegt die Akku-Spannung über eine Diode am Speicher IC an. Die zweite Diode verhindert, daß sich der Akku über die ange-

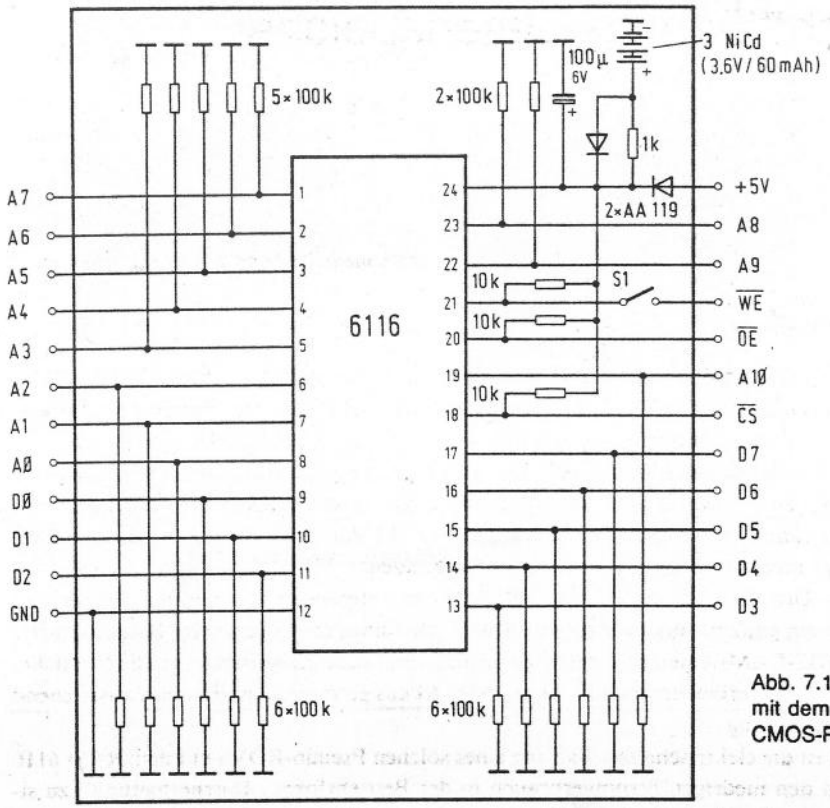


Abb. 7.1 Pseudo-ROM mit dem CMOS-RAM 6116

schlossene Schaltung entladen kann. Als Dioden müssen Ge-Typen verwendet werden, damit im Betrieb der Spannungsabfall möglichst gering bleibt.

Abb. 7.2 zeigt einen Musteraufbau des Pseudo-ROMs. Die gesamte Verdrahtung ist an einem DIL-Stecker angelötet und zu einer IC-Fassung geführt, in die das Speicher-IC eingesetzt wird. Bei einigermaßen geschicktem Aufbau lassen sich zwischen Stecker und Fassung Widerstände, Dioden und der Akku unterbringen. Den Schreibschutz-Schalter setzt man an der Seite an. Diese Aufbau-Methode hat den Vorteil, daß sich das Pseudo-ROM jederzeit gegen ein normales Speicher-IC austauschen läßt. Einziger Punkt der zu beachten ist: die Bauhöhe ist deutlich größer. Dies kann beim Einsatz im 19-Zoll-Gehäuse („language card") zu Schwierigkeiten führen. Man muß dann den nächstfolgenden Steckplatz im Gehäuse unbenützt lassen, um die notwendige Bauhöhe zu erhalten. Aus Platzgründen ist es auch sinnvoll, den Schreibschutz-Schalter an der Stirnseite des ICs anzuordnen. Infolge der relativ dichten Pakkung der Speicher-ICs in Querrichtung sind sonst nicht mehrere Pseudo-ROMs nebeneinander unterzubringen.

7.2 Betrieb mit dem Pseudo-ROM

Soll das Pseudo-ROM auf der Grundplatine I als Speicher für Maschinenprogramme dienen, so muß das IC in der Steckposition III sitzen. Grund: Der Computer benötigt für BASIC ei-

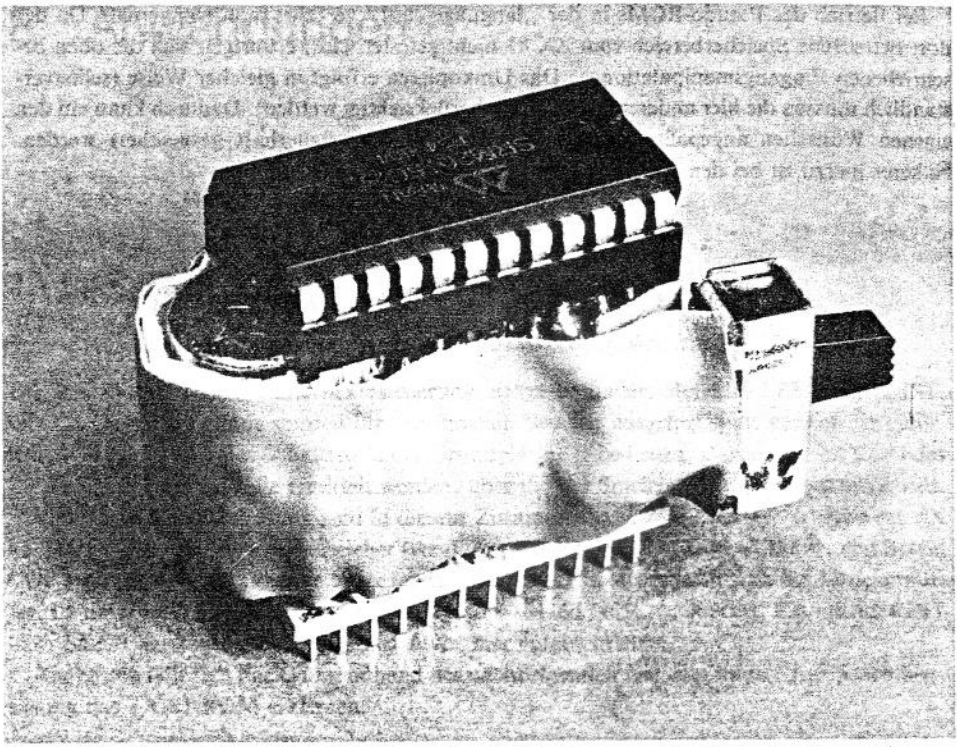


Abb. 7.2 Musteraufbau des Pseudo-ROMs

nen durchgehenden Speicherbereich, beim Einstecken in anderen Positionen wäre er unterbrochen. Vor dem Einschalten des Computers öffnet man den Schreibschutz-Schalter am Pseudo-ROM. Schaltet man jetzt das Gerät ein, findet der ZX 81 das Pseudo-ROM nicht, da er es nicht beschreiben kann. Es wird dadurch automatisch oberhalb RAMTOP angesiedelt. Nun wird der Schreibschutz-Schalter geschlossen. Jetzt kann das gewünschte Maschinenprogramm aus einem REM-Statement in das Pseudo-ROM umkopiert werden. Als Beispiel möge das nachfolgende BASIC-Programm dienen. Maschinenprogramme sind natürlich wesentlich schneller.

```
100 FOR N = 16514 TO ENDADRESSE
200 POKE (N+4990), PEEK N
300 NEXT N
```

Für ENDADRESSE wird die Adresse des letzten Maschinenprogramm-Befehls eingesetzt. N+4990 zeigt auf die Zieladresse, in diesem Beispiel beginnend bei 16514+4990=21504.

Nachdem die gewünschten Maschinenprogramme umkopiert sind, öffnet man den Schreibschutz-Schalter wieder. Das Pseudo-ROM ist jetzt schreibgeschützt und sein Programminhalt kann nicht zerstört werden. Es ist sogar möglich, das Pseudo-ROM aus der Fassung der Grundplatine herauszunehmen und beispielsweise in einem anderen Computer einzusetzen. Auf diese Weise wird die Software portabel.

Bei Betrieb des Pseudo-ROMs in der „language card" verfährt man sinnv
dort betroffene Speicherbereich vom ZX 81 nicht getestet wird, erübrigen sic
schriebenen Eingangsmanipulationen. Das Umkopieren erfolgt in gleicher W
ständlich müssen die hier anderen Adreßwerte berücksichtigt werden). Dadurc
eigenen Wünschen angepaßtes, geändertes Betriebssystem dauerhaft gespeichert werden.
Näheres hierzu ist bei den Anwendungen in Kapitel 8 nachzulesen.

Anmerkung: diese Lösung ist besonders geeignet um Software zu testen, die später dauerhaft in einem echten ROM / Eprom gebrannt werden soll ! Es ermöglicht quasi einen Einsatz unter "Realbedingungen".

8 Anwendungen

8.1 Language Card/Grundplatine II

8.1.1 Einbindung der Druckeransteuerung in das Betriebssystem des ZX 81

Der ZX 81 besitzt für die Druckersteuerung einige besondere Befehle: LPRINT, LLIST, COPY. Da diese Befehle speziell für den Sinclair-Drucker ausgelegt sind, können sie natürlich nicht die Centronics-Schnittstelle der Grundplatine II bedienen. Daher muß im ROM des ZX 81 die Programmstelle ermittelt werden, über die der Sinclair-Drucker angesteuert wird.

Von besonderer Bedeutung sind in diesem Zusammenhang zwei Speicherbereiche des ZX 81. Der eine ist der Bildschirmspeicher (vergl. Kapitel 1.4.3). Sein Inhalt wird über den Befehl COPY direkt auf den Drucker ausgegeben. Der andere Speicherbereich ist der Druckerpuffer. Er umfaßt 33 Speicherplätze von 16444d bis 16476d (403C ... 405Ch). Ein mit LPRINT bzw. LLIST auszugebender Text wird zuvor hier eingeschrieben.

Am besten läßt sich das Ganze anhand des nachfolgenden Ausschnitts aus dem Assembler-Listing des ZX-81-ROM erkennen.

```
                                          DRUCKERPUFFER VORBEREITEN
0851:FE 76      CP    76          Test auf NEWLINE
0853:28 1C      JR    Z,0871      Falls NEWLINE, Sprung nach 871h
0855:4F         LD    C,A
0856:3A 38 40   LD    A,(4038)    Akku laden mit nächster Druckadresse
0859:E6 7F      AND   7F          Unterdrücken von Bit 7
085B:FE 5C      CP    5C          Test, ob Druckerpuffer zu Ende
085D:6F         LD    L,A         HL wird mit Adresse des Druckerpuffers geladen
085E:26 40      LD    H,40
0860:CC 71 08   CALL  Z,0871      Falls Pufferende erreicht, Sprung nach 871h
0863:71         LD    (HL),C      Druckzeichen wird auf nächste freie Position
                                  im Druckerpuffer geschrieben
0864:2C         INC   L
0865:FD 75 38   LD    (IY+38),L
0868:C9         RET
                                          BILDSCHIRM-KOPIE VORBEREITEN
0869:16 16      LD    D,16        D laden mit Zeilenzahl
086B:2A 0C 40   LD    HL,(400C)   HL laden mit D-File (Adresse des Bildschirm-Speichers
086E:23         INC   HL
086F:18 05      JR    0876        Druckprogramm anspringen
                                          PUFFERKOPIE VORBEREITEN
0871:16 01      LD    D,01        D laden mit Zeilenzahl
0873:21 3C 40   LD    HL,403C     HL laden mit Adresse des Druckerpuffers
0876:C3 1F 2A   JP    2A1F        Sprung zur eigenen Ausgaberoutine
                                  (ursprünglich: Beginn der Sinclair-Druckerroutine)
```

DRUCKERPUFFER LÖSCHEN

```
08E2:21 5C 40     LD   HL,405C     HL laden mit Endadresse Puffer
08E5:36 76        LD   (HL),76     NEWLINE einschreiben (in letzte Position)
08E7:06 20        LD   B,20        B laden mit Anzahl Positionen im Puffer
08E9:2B           DEC  HL
08EA:36 00        LD   (HL),00     } alle Positionen im Ø beschreiben
08EC:10 FB        DJNZ 08E9        }
08EE:7D           LD   A,L
08EF:CB FF        SET  7,A
08F1:32 38 40     LD   (4038),A
08F4:C9           RET
```

Prog. 5

Zwischen 851h und 868h wird der Druckerpuffer vorbereitet (LLIST, LPRINT). Dieses Programm springt anschließend die Speicherstelle 871h an, wo die zum Ausdruck wichtigen Parameter an die Register D (Anzahl der auszugebenden Zeilen) und HL (Adresse, bei der der Ausdruck startet) übergeben werden. Anschließend wird das Druckerprogramm beginnend bei 876h aufgerufen.

Soll der Bildschirm kopiert werden, so trifft der Programmteil ab 869h die entsprechenden Vorbereitungen. Register D erhält den Wert 22d (Anzahl der Bildschirmzeilen) und das Registerpaar HL zeigt auf den Beginn des Bildschirmspeichers. Auch dieses Programm ruft das Druckerprogramm ab 876h auf.

Nach erfolgtem Ausdruck löscht das Unterprogramm 8E2h ...8F4h den Druckerpuffer, eine neue Ausgabe kann beginnen.

Damit ist der Ansatzpunkt für die Treiberroutine zur Bedienung der Centronics-Schnittstelle gefunden. Daher wird ab Adresse 876h ein Sprung zu dieser Treiberroutine eingegeben. Im Beispiel liegt diese Treiberroutine im RAM der „lanuage card" (JP 2A1Fh - 10783d). Nach dem Durchlaufen der Routine erfolgt der Rücksprung in das ZX-81-Betriebssystem bei Adresse 8E2h (2274d).

Da der ZX 81 nicht den ASCII-Code verwendet, muß vor der Ausgabe eines Zeichens auf den Drucker eine Umkodierung erfolgen. Am einfachsten geschieht dies in Form einer Tabelle, bei der die Adresse durch den ZX-Code (+ einer Konstanten), der Inhalt der jeweiligen Adresse durch den dazugehörigen ASCII-Wert dargestellt ist. Die Invers-Buchstaben werden in der hier gewählten Codierung in Kleinschrift wiedergegeben.

Ein besonderes Problem stellen die Grafik-Zeichen des ZX 81 dar, da sie in der Regel vom Zeichensatz des Druckers nicht abgedeckt werden. Bei Druckern, die das Definieren eigener Zeichen gestatten, ist es einfach: man definiert die Zeichen eben selbst. Das nachfolgende Beispiel ist auf den Drucker FX 80 von Epson abgestimmt. Um eigene Zeichen zu definieren, müssen bestimmte Kontroll-Codes zum Drucker geschickt werden. Der Einfachheit halber wurde kein Unterschied zwischen Normal- und Inversgrafik gemacht. Im Zweifelsfall sollte das Drucker-Handbuch zu Rate gezogen werden.

Bevor der Drucker benützt werden kann, müssen also Umkodiertabelle, Grafikdefinition samt Initialisierungsroutine sowie das Druckerprogramm in den RAM-Bereich der „language card" geladen werden. Dieses Programm- und Datenpaket, zusammen mit den Routinen zum Umkopieren des ROMs in die „language card" und Einbringen des Sprungbefehls in die Sinclair-Druckerroutine ist in einem REM-Statement untergebracht. Den Schluß bildet noch

der Programmabschnitt, mit dem die Programmteile aus dem REM-Statement an die Zielstellen gebracht werden.

Das gesamte Programm-Paket ist nachfolgend in dezimaler Form aufgelistet. Damit man sich nicht schon beim Erstellen des REM-Statements, das rund 650 Leerzeichen bereithalten muß, wunde Finger holt, ist in Kapitel 1.4.4 ein Hilfsprogramm aufgelistet, das diese Arbeit übernimmt.

UMKODIERTABELLE

```
16514  ---> 32
16515  ---> 33   ▘
16516  ---> 35   ▝
16517  ---> 37   ▀
16518  ---> 38   ▖
16519  ---> 39   ▌
16520  ---> 95   ▚
16521  ---> 91   ▛
16522  ---> 92   ▒
16523  ---> 93   ▞
16524  ---> 94   ▞
16525  ---> 34   "
16526  ---> 123  £
16527  ---> 36   $
16528  ---> 58   :
16529  ---> 63   ?
16530  ---> 40   (
16531  ---> 41   )
16532  ---> 62   >
16533  ---> 60   <
16534  ---> 61   =
16535  ---> 43   +
16536  ---> 45   -
16537  ---> 42   *
16538  ---> 47   /
16539  ---> 59   ;
16540  ---> 44   ,
16541  ---> 46   .
16542  ---> 48   0
16543  ---> 49   1
16544  ---> 50   2
16545  ---> 51   3
16546  ---> 52   4
16547  ---> 53   5
16548  ---> 54   6
16549  ---> 55   7
16550  ---> 56   8
16551  ---> 57   9
16552  ---> 65   A
16553  ---> 66   B
16554  ---> 67   C
16555  ---> 68   D
16556  ---> 69   E
16557  ---> 70   F
16558  ---> 71   G
16559  ---> 72   H
16560  ---> 73   I
16561  ---> 74   J
16562  ---> 75   K
16563  ---> 76   L
16564  ---> 77   M
16565  ---> 78   N
16566  ---> 79   O
16567  ---> 80   P
16568  ---> 81   Q
16569  ---> 82   R
16570  ---> 83   S
16571  ---> 84   T
16572  ---> 85   U
16573  ---> 86   V
16574  ---> 87   W
16575  ---> 88   X
16576  ---> 89   Y
16577  ---> 90   Z
16578  ---> 32   RND
16579  ---> 32   INKEY$
16580  ---> 32   PI
16581  ---> 32   ?
16582  ---> 32   ?
16583  ---> 32   ?
16584  ---> 32   ?
16585  ---> 32   ?
16586  ---> 32   ?
16587  ---> 32   ?
16588  ---> 32   ?
16589  ---> 32   ?
16590  ---> 32   ?
16591  ---> 32   ?
```

```
16592  ---> 32   ?
16593  ---> 32   ?
16594  ---> 32   ?
16595  ---> 32   ?
16596  ---> 32   ?
16597  ---> 32   ?
16598  ---> 32   ?
16599  ---> 32   ?
16600  ---> 32   ?
16601  ---> 32   ?
16602  ---> 32   ?
16603  ---> 32   ?
16604  ---> 32   ?
16605  ---> 32   ?
16606  ---> 32   ?
16607  ---> 32   ?
16608  ---> 32   ?
16609  ---> 32   ?
16610  ---> 32   ?
16611  ---> 32   ?
16612  ---> 32   ?
16613  ---> 32   ?
16614  ---> 32   ?
16615  ---> 32   ?
16616  ---> 32   ?
16617  ---> 32   ?
16618  ---> 32   ?
16619  ---> 32   ?
16620  ---> 32   ?
16621  ---> 32   ?
16622  ---> 32   ?
16623  ---> 32   ?
16624  ---> 32   ?
16625  ---> 32   ?
16626  ---> 32   ?
16627  ---> 32   ?
16628  ---> 32   ?
16629  ---> 32   ?
16630  ---> 32   ?
16631  ---> 32   ?
16632  ---> 32   ?
16633  ---> 32   ?
16634  ---> 32   ?
16635  ---> 32   ?
16636  ---> 32   ?
16637  ---> 32   ?
16638  ---> 32   ?
16639  ---> 32   ?
16640  ---> 32   ?
16641  ---> 32   ?
16642  ---> 124  ■
16643  ---> 33   ▪
16644  ---> 35   ▪
16645  ---> 37   ▬
16646  ---> 38   ▪
16647  ---> 39   ▮
16648  ---> 95   ▪
16649  ---> 91   ▛
16650  ---> 92   ▒
16651  ---> 93   ▫
16652  ---> 94   ▫
16653  ---> 162  "
16654  ---> 219  Ä
16655  ---> 164  $
16656  ---> 186  :
16657  ---> 191  ?
16658  ---> 168  (
16659  ---> 169  )
16660  ---> 190  >
16661  ---> 188  <
16662  ---> 189  =
16663  ---> 171  +
16664  ---> 173  -
16665  ---> 170  *
16666  ---> 175  /
16667  ---> 187  ;
16668  ---> 172  ,
16669  ---> 174  .
16670  ---> 176  0
16671  ---> 177  1
16672  ---> 178  2
16673  ---> 179  3
16674  ---> 180  4
16675  ---> 181  5
16676  ---> 182  6
16677  ---> 183  7
16678  ---> 184  8
16679  ---> 185  9
16680  ---> 97   a
16681  ---> 98   b
16682  ---> 99   c
16683  ---> 100  d
16684  ---> 101  e
16685  ---> 102  f
16686  ---> 103  g
16687  ---> 104  h
16688  ---> 105  i
16689  ---> 106  j
16690  ---> 107  k
16691  ---> 108  l
16692  ---> 109  m
16693  ---> 110  n
16694  ---> 111  o
16695  ---> 112  p
16696  ---> 113  q
16697  ---> 114  r
16698  ---> 115  s
16699  ---> 116  t
16700  ---> 117  u
16701  ---> 118  v
```

```
16702 ---> 119   w
16703 ---> 120   x
16704 ---> 121   y
16705 ---> 122   z
16706 ---> 34    "
16707 ---> 32    AT
16708 ---> 32    TAB
16709 ---> 32    ?
16710 ---> 32    CODE
16711 ---> 32    VAL
16712 ---> 32    LEN
16713 ---> 32    SIN
16714 ---> 32    COS
16715 ---> 32    TAN
16716 ---> 32    ASN
16717 ---> 32    ACS
16718 ---> 32    ATN
16719 ---> 32    LN
16720 ---> 32    EXP
16721 ---> 32    INT
16722 ---> 32    SQR
16723 ---> 32    SGN
16724 ---> 32    ABS
16725 ---> 32    PEEK
16726 ---> 32    USR
16727 ---> 32    STR$
16728 ---> 32    CHR$
16729 ---> 32    NOT
16730 ---> 101   **
16731 ---> 32    OR
16732 ---> 32    AND
16733 ---> 125   <=
16734 ---> 126   >=
16735 ---> 96    <>
16736 ---> 32    THEN
16737 ---> 32    TO
16738 ---> 32    STEP
16739 ---> 32    LPRINT
16740 ---> 32    LLIST
16741 ---> 32    STOP
16742 ---> 32    SLOW
16743 ---> 32    FAST
16744 ---> 32    NEW
16745 ---> 32    SCROLL
16746 ---> 32    CONT
16747 ---> 32    DIM
16748 ---> 32    REM
16749 ---> 32    FOR
16750 ---> 32    GOTO
16751 ---> 32    GOSUB
16752 ---> 32    INPUT
16753 ---> 32    LOAD
16754 ---> 32    LIST
16755 ---> 32    LET
16756 ---> 32    PAUSE
16757 ---> 32    NEXT
16758 ---> 32    POKE
16759 ---> 32    PRINT
16760 ---> 32    PLOT
16761 ---> 32    RUN
16762 ---> 32    SAVE
16763 ---> 32    RAND
16764 ---> 32    IF
16765 ---> 32    CLS
16766 ---> 32    UNPLOT
16767 ---> 32    CLEAR
16768 ---> 32    RETURN
16769 ---> 32    COPY
```

GRAFIK -> EPSON

```
16770 ---> 27
16771 ---> 58
16772 ---> 0
16773 ---> 0
16774 ---> 0
16775 ---> 27
16776 ---> 37
16777 ---> 1
16778 ---> 0
16779 ---> 27
16780 ---> 38
16781 ---> 0
16782 ---> 33
16783 ---> 33
16784 ---> 139
16785 ---> 240
16786 ---> 0
16787 ---> 240
16788 ---> 0
16789 ---> 240
16790 ---> 0
16791 ---> 0
16792 ---> 0
16793 ---> 0
16794 ---> 0
16795 ---> 0
16796 ---> 27
16797 ---> 38
16798 ---> 0
16799 ---> 35
16800 ---> 35
```

```
16801  ---> 139
16802  ---> 0
16803  ---> 0
16804  ---> 0
16805  ---> 0
16806  ---> 0
16807  ---> 0
16808  ---> 240
16809  ---> 0
16810  ---> 240
16811  ---> 0
16812  ---> 240
16813  ---> 27
16814  ---> 38
16815  ---> 0
16816  ---> 37
16817  ---> 37
16818  ---> 139
16819  ---> 240
16820  ---> 0
16821  ---> 240
16822  ---> 0
16823  ---> 240
16824  ---> 0
16825  ---> 240
16826  ---> 0
16827  ---> 240
16828  ---> 0
16829  ---> 240
16830  ---> 27
16831  ---> 38
16832  ---> 0
16833  ---> 38
16834  ---> 38
16835  ---> 139
16836  ---> 15
16837  ---> 0
16838  ---> 15
16839  ---> 0
16840  ---> 15
16841  ---> 0
16842  ---> 0
16843  ---> 0
16844  ---> 0
16845  ---> 0
16846  ---> 0
16847  ---> 27
16848  ---> 38
16849  ---> 0
16850  ---> 39
16851  ---> 39
16852  ---> 139
16853  ---> 255
16854  ---> 0
16855  ---> 255
16856  ---> 0
16857  ---> 255
16858  ---> 0
16859  ---> 0
16860  ---> 0
16861  ---> 0
16862  ---> 0
16863  ---> 0
16864  ---> 27
16865  ---> 38
16866  ---> 0
16867  ---> 95
16868  ---> 95
16869  ---> 139
16870  ---> 15
16871  ---> 0
16872  ---> 15
16873  ---> 0
16874  ---> 15
16875  ---> 0
16876  ---> 240
16877  ---> 0
16878  ---> 240
16879  ---> 0
16880  ---> 240
16881  ---> 27
16882  ---> 38
16883  ---> 0
16884  ---> 91
16885  ---> 91
16886  ---> 139
16887  ---> 255
16888  ---> 0
16889  ---> 255
16890  ---> 0
16891  ---> 255
16892  ---> 0
16893  ---> 240
16894  ---> 0
16895  ---> 240
16896  ---> 0
16897  ---> 240
16898  ---> 27
16899  ---> 38
16900  ---> 0
16901  ---> 92
16902  ---> 92
16903  ---> 139
16904  ---> 170
16905  ---> 0
16906  ---> 85
16907  ---> 0
16908  ---> 170
16909  ---> 0
16910  ---> 85
```

```
16911  ---> 0
16912  ---> 170
16913  ---> 0
16914  ---> 85
16915  ---> 27
16916  ---> 38
16917  ---> 0
16918  ---> 93
16919  ---> 93
16920  ---> 139
16921  ---> 10
16922  ---> 0
16923  ---> 5
16924  ---> 0
16925  ---> 10
16926  ---> 0
16927  ---> 5
16928  ---> 0
16929  ---> 10
16930  ---> 0
16931  ---> 5
16932  ---> 27
16933  ---> 38
16934  ---> 0
16935  ---> 94
16936  ---> 94
16937  ---> 139
16938  ---> 160
16939  ---> 0
16940  ---> 80
16941  ---> 0
16942  ---> 160
16943  ---> 0
16944  ---> 80
16945  ---> 0
16946  ---> 160
16947  ---> 0
16948  ---> 80
16949  ---> 27
16950  ---> 38
16951  ---> 0
16952  ---> 123
16953  ---> 123
16954  ---> 139
16955  ---> 0
16956  ---> 18
16957  ---> 0
16958  ---> 126
16959  ---> 128
16960  ---> 18
16961  ---> 128
16962  ---> 2
16963  ---> 128
16964  ---> 66
16965  ---> 0
16966  ---> 27
16967  ---> 38
16968  ---> 0
16969  ---> 124
16970  ---> 124
16971  ---> 139
16972  ---> 255
16973  ---> 0
16974  ---> 255
16975  ---> 0
16976  ---> 255
16977  ---> 0
16978  ---> 255
16979  ---> 0
16980  ---> 255
16981  ---> 0
16982  ---> 255
16983  ---> 27
16984  ---> 38
16985  ---> 0
16986  ---> 125
16987  ---> 125
16988  ---> 139
16989  ---> 0
16990  ---> 20
16991  ---> 32
16992  ---> 84
16993  ---> 128
16994  ---> 20
16995  ---> 0
16996  ---> 20
16997  ---> 0
16998  ---> 20
16999  ---> 0
17000  ---> 27
17001  ---> 38
17002  ---> 0
17003  ---> 126
17004  ---> 126
17005  ---> 139
17006  ---> 0
17007  ---> 20
17008  ---> 0
17009  ---> 20
17010  ---> 0
17011  ---> 20
17012  ---> 128
17013  ---> 84
17014  ---> 32
17015  ---> 20
17016  ---> 0
17017  ---> 27
17018  ---> 38
17019  ---> 0
17020  ---> 96
```

```
17021  ---> 96
17022  ---> 139
17023  ---> 0
17024  ---> 40
17025  ---> 0
17026  ---> 40
17027  ---> 0
17028  ---> 254
17029  ---> 0
17030  ---> 40
17031  ---> 0
17032  ---> 40
17033  ---> 0
17034  ---> 118
17035  ---> 0
17036  ---> 0
```

■ LADEN EPSON MIT GRAFIK ■

```
17037  ---> 33
17038  ---> 0
17039  ---> 41
17040  ---> 1
17041  ---> 10
17042  ---> 1
17043  ---> 17
17044  ---> 3
17045  ---> 60
17046  ---> 237
17047  ---> 160
17048  ---> 224
17049  ---> 27
17050  ---> 26
17051  ---> 203
17052  ---> 127
17053  ---> 32
17054  ---> 251
17055  ---> 24
17056  ---> 245
```

■ MASCHINENPROGRAMM : DRUCK ■

```
17057  ---> 197
17058  ---> 245
17059  ---> 229
17060  ---> 213
17061  ---> 1
17062  ---> 0
17063  ---> 40
17064  ---> 126
17065  ---> 254
17066  ---> 118
17067  ---> 40
17068  ---> 15
17069  ---> 78
17070  ---> 10
17071  ---> 50
17072  ---> 3
17073  ---> 60
17074  ---> 58
17075  ---> 3
17076  ---> 60
17077  ---> 203
17078  ---> 127
17079  ---> 32
17080  ---> 249
17081  ---> 35
17082  ---> 24
17083  ---> 236
17084  ---> 62
17085  ---> 13
17086  ---> 50
17087  ---> 3
17088  ---> 60
17089  ---> 58
17090  ---> 3
17091  ---> 60
17092  ---> 203
17093  ---> 127
17094  ---> 32
17095  ---> 249
17096  ---> 62
17097  ---> 10
17098  ---> 21
17099  ---> 32
17100  ---> 226
17101  ---> 50
17102  ---> 3
17103  ---> 60
17104  ---> 58
17105  ---> 3
17106  ---> 60
```

```
17107 ---> 203
17108 ---> 127
17109 ---> 32
17110 ---> 249
17111 ---> 209
17112 ---> 225
17113 ---> 241
17114 ---> 193
17115 ---> 195
17116 ---> 226
17117 ---> 8
```

```
17127 ---> 237
17128 ---> 176
17129 ---> 50
17130 ---> 8
17131 ---> 60
17132 ---> 33
17133 ---> 118
17134 ---> 8
17135 ---> 62
17136 ---> 195
17137 ---> 119
17138 ---> 35
17139 ---> 62
17140 ---> 31
17141 ---> 119
17142 ---> 35
17143 ---> 62
17144 ---> 42
17145 ---> 119
17146 ---> 33
17147 ---> 130
17148 ---> 64
17149 ---> 17
17150 ---> 0
17151 ---> 40
17152 ---> 1
17153 ---> 92
17154 ---> 2
17155 ---> 237
17156 ---> 176
17157 ---> 201
```

```
■ KOPIEREN DATEN REM -> RAM ■
```

```
17118 ---> 33
17119 ---> 0
17120 ---> 0
17121 ---> 17
17122 ---> 0
17123 ---> 0
17124 ---> 1
17125 ---> 0
17126 ---> 32
```

Prog. 6

Zum besseren Verständnis werden die wichtigsten Teile aus dem Programmpaket in disassemblierter und kommentierter Form aufgelistet. Dabei ist zu beachten, daß sich die angegebenen Adressen auf die ursprüngliche Programmlage im REM-Statement beziehen.

Laden des Druckers mit Grafik-Zeichen (17037 ... 17056):

```
428D:21 00 29    LD   HL,2900   HL mit Startadresse für Grafik-Kodierung laden
4290:01 0A 01    LD   BC,010A   BC mit Anzahl der Zeichen laden
4293:11 03 3C    LD   DE,3C03   DE mit Zieladresse (Centronics OUT 2) laden
4296:ED A0       LDI             Transferbefehl
4298:E0          RET  PO         Falls letztes Zeichen: Rücksprung
4299:1B          DEC  DE         DE wieder auf Zieladresse (Centronics OUT 2) einste
429A:1A          LD   A,(DE)     Akku mit Druckerrückmeldung laden
429B:CB 7F       BIT  7,A        Bit 7 testen, ob Drucker bereit für neues Zeichen
429D:20 FB       JR   NZ,429A    Warten, bis Drucker bereit
429F:18 F5       JR   4296       Sprung zum Transfer des nächsten Zeichens
```

Prog. 7

Druckerprogramm (17057 ... 17117):

```
42A1:C5          PUSH BC
42A2:F5          PUSH AF
42A3:E5          PUSH HL
42A4:D5          PUSH DE
42A5:01 00 28    LD   BC,2800
42A8:7E          LD   A,(HL)
42A9:FE 76       CP   76
42AB:28 0F       JR   Z,42BC
42AD:4E          LD   C,(HL)
42AE:0A          LD   A,(BC)
42AF:32 03 3C    LD   (3C03),A
42B2:3A 03 3C    LD   A,(3C03)
42B5:CB 7F       BIT  7,A
42B7:20 F9       JR   NZ,42B2
42B9:23          INC  HL
42BA:18 EC       JR   42A8
42BC:3E 0D       LD   A,0D
42BE:32 03 3C    LD   (3C03),A
42C1:3A 03 3C    LD   A,(3C03)
42C4:CB 7F       BIT  7,A
42C6:20 F9       JR   NZ,42C1
42C8:3E 0A       LD   A,0A
42CA:15          DEC  D
42CB:20 E2       JR   NZ,42AF
42CD:32 03 3C    LD   (3C03),A
42D0:3A 03 3C    LD   A,(3C03)
42D3:CB 7F       BIT  7,A
42D5:20 F9       JR   NZ,42D0
42D7:D1          POP  DE
42D8:E1          POP  HL
42D9:F1          POP  AF
42DA:C1          POP  BC
42DB:C3 E2 08    JP   08E2
```

Register retten

BC mit Startadresse für Umkodiertabelle laden

Auszugebendes Zeichen in Akku laden

Falls NEWLINE, Sprung relativ + 15d

C laden mit Code des auszugebenden Zeichens

Aus Kodiertabelle Ausgabecode für Drucker ermitteln

Zeichen zum Drucker schicken (Centronics OUT 2)

Akku mit Druckerrückmeldung laden

Warten, bis Drucker bereit für neues Zeichen

Zeiger auf nächstes Zeichen

Sprung zu neuer Zeichenausgabe

Bei NEWLINE: Ausgabe von CR (carriage return - Wagenrücklauf) und LF (line feed - neue Zeile) an den Drucker

Wiedergewinnen der Register

Rücksprung zum Löschen des Druckerpuffers

Prog. 8

Initialisieren der „language card" (17118 ... 17128):

```
42DE:21 00 00    LD   HL,0000
42E1:11 00 00    LD   DE,0000
42E4:01 00 20    LD   BC,2000
42E7:ED B0       LDIR
```

HL mit Adresse der Datenquelle laden (ROM)

DF mit Adresse des Datenziels laden (RAM)

BC mit Anzahl der Datenbytes laden

Blocktransfer-Befehl, kopiert ROM in RAM

Prog. 9

Laden des Sprungvektors nach 876h (17129 ... 17145):

```
42E9:32 08 3C     LD    (3C08),A   Umschalten auf RAM statt ROM
42EC:21 76 08     LD    HL,0876    HL mit Adresse für Sprungvektor laden
42EF:3E C3        LD    A,C3
42F1:77           LD    (HL),A
42F2:23           INC   HL
42F3:3E 1F        LD    A,1F       Sprungbefehl zur neuen Druckerroutine
42F5:77           LD    (HL),A     ab 876 h ins RAM laden
42F6:23           INC   HL
42F7:3E 2A        LD    A,2A
42F9:77           LD    (HL),A
```

Prog. 10

Umkopieren von Code-Tabelle, Druckerinitialisierung und Druckerprogramm nach 10240 ff (17146 ... 17157):

```
42FA:21 82 40     LD    HL,4082    HL mit Startadresse der Umkodiertabelle laden
                                   (REM-Statement)
42FD:11 00 28     LD    DE,2800    DE mit Zieladresse laden (Maschinenprogramm
                                   speicher auf „language card")
4300:01 5C 02     LD    BC,025C    BC mit Anzahl der Datenbytes laden
4303:ED B0        LDIR             Blocktransferbefehl, kopiert Umkodiertabelle um
4305:C9           RET              Rücksprung aus Unterprogramm
```

Prog. 11

Die drei letztgenannten Programmsegmente werden nicht aus dem REM-Statement hochkopiert, da sie später nicht mehr benötigt werden.

Nach dem Eingeben wird das Initialisierungsprogramm vollends fertiggestellt:

```
100 REM (enthält das obige Programmpaket)
200 LET A = USR (17118)
300 LET A = USR (10763)
```

Zeile 200 startet die Umkopierung. Zeile 300 sorgt dafür, daß die Sonderzeichen auf dem Drucker vorhanden sind. Wird später im Betrieb der Drucker zwischenzeitlich abgeschaltet, kann die Sonderzeichen-Initialisierung jederzeit mit

```
LET A =USR (10763)
```

wieder aufgerufen werden.

Bevor nun aber das Programm mit RUN gestartet wird, unbedingt den Maschinenprogramm-Teil sorgfältig kontrollieren und dann das Ganze erst einmal abspeichern. Falls doch noch ein Fehler enthalten war und der Rechner sich beim ersten Probelauf „aufhängt", war wenigstens die Tipparbeit nicht umsonst, man muß nur den Fehler suchen.

Ist die Drucker-Initialisierung erfolgreich abgeschlossen, kann man ans Werk gehen. Die Befehle LLIST, LPRINT und COPY funktionieren wie erwartet.

8.1.2 Tastatur-Umkodierung

Drückt man beim ZX 81 beispielsweise in der obersten Reihe die links außen liegende Taste, so erscheint auf dem Bildschirm die Ziffer „1". Die Zuordnung von Taste und Zeichen bzw. Symbol ist durch eine Codiertabelle festgelegt, die im ROM des ZX 81 zwischen den Adressen

7Eh (126d) und 110h (272d) liegt. Dabei weist der aus der Tastendekodierung gewonnene Zahlenwert (vergl. Kapitel 1.4.2) auf die entsprechende Speicherstelle im ROM, während der auf dieser Speicherstelle eingeschriebene Wert das jeweilige Symbol im ZX-Code darstellt.

```
126 :  63 --> Z      175 : 217--> OR      224 :  120--> ?
127 :  61 --> X      176 : 224--> STEP    225 :  120--> ?
128 :  40 --> C      177 : 219--> <=      226 :  120--> ?
129 :  59 --> V      178 : 221--> <>      227 :  120--> ?
130 :  38 --> A      179 : 117--> ?       228 :  194--> TAB
131 :  56 --> S      180 : 218--> AND     229 :  211--> PEEK
132 :  41 --> D      181 : 222--> THEN    230 :  196--> CODE
133 :  43 --> F      182 : 223--> TO      231 :  214--> CHR$
134 :  44 --> G      183 : 114--> ?       232 :  213--> STR$
135 :  54 --> Q      184 : 119--> ?       233 :  120--> ?
136 :  60 --> W      185 : 116--> ?       234 :  212--> USR
137 :  42 --> E      186 : 115--> ?       235 :  198--> LEN
138 :  55 --> R      187 : 112--> ?       236 :  197--> VAL
139 :  57 --> T      188 : 113--> ?       237 :  208--> SQR
140 :  29 --> 1      189 :  11 --> "      238 :  120--> ?
141 :  30 --> 2      190 :  17 --> )      239 :  120--> ?
142 :  31 --> 3      191 :  16 --> (      240 :  66 --> PI
143 :  32 --> 4      192 :  13 --> $      241 :  215--> NOT
144 :  33 --> 5      193 : 220--> >=      242 :  65 --> INKEY$
145 :  28 --> 0      194 : 121--> ?       243 :  8  --> ▒
146 :  37 --> 9      195 :  20 --> =      244 :  10 --> ▒
147 :  36 --> 8      196 :  21 --> +      245 :  9  --> ▒
148 :  35 --> 7      197 :  22 --> -      246 :  138--> ▒
149 :  34 --> 6      198 : 216--> **      247 :  137--> ▒
150 :  53 --> P      199 :  12 --> £      248 :  129--> ▘
151 :  52 --> O      200 :  26 --> ,      249 :  130--> ▝
152 :  46 --> I      201 :  18 --> >      250 :  7  --> ▛
153 :  58 --> U      202 :  19 --> <      251 :  132--> ▀
154 :  62 --> Y      203 :  23 --> *      252 :  6  --> ▞
155 : 118--> ?       204 : 205--> LN      253 :  1  --> ▘
156 :  49 --> L      205 : 206--> EXP     254 :  2  --> ▝
157 :  48 --> K      206 : 193--> AT      255 :  135--> ▛
158 :  47 --> J      207 : 120--> ?       256 :  4  --> ▀
159 :  45 --> H      208 : 202--> ASN     257 :  5  --> ▌
160 :  0  -->        209 : 203--> ACS     258 :  119--> ?
161 :  27 --> .      210 : 204--> ATN     259 :  120--> ?
162 :  50 --> M      211 : 209--> SGN     260 :  133--> ▌
163 :  51 --> N      212 : 210--> ABS     261 :  3  --> ▬
164 :  39 --> B      213 : 199--> SIN     262 :  131--> ▬
165 :  14 --> :      214 : 200--> COS     263 :  139--> "
166 :  25 --> ;      215 : 201--> TAN     264 :  145--> )
167 :  15 --> ?      216 : 207--> INT     265 :  144--> (
168 :  24 --> /      217 :  64 --> RND    266 :  141--> $
169 : 227--> STOP    218 : 120--> ?       267 :  134--> ▞
170 : 225--> LPRINT  219 : 120--> ?       268 :  120--> ?
171 : 228--> SLOW    220 : 120--> ?       269 :  146--> ;
172 : 229--> FAST    221 : 120--> ?       270 :  149--> +
173 : 226--> LLIST   222 : 120--> ?       271 :  150--> -
174 : 192--> "       223 : 120--> ?       272 :  136--> ▒
```

Prog. 12

In dieser Tabelle sind die inversen Symbole kursiv ausgedruckt. Einige Symbole werden aus druckertechnischen Gründen als Fragezeichen ausgedruckt. Das „echte" Fragezeichen (Code 15) steht auf Speicherstelle 167. Speicherstellen, die den Wert 120 enthalten, sind unbenützt.

Ein Beispiel soll die Umkodierung erläutern:
Speicherstelle 126d (7Eh) enthält den Wert 63d, das ist laut ZX-81-Handbuch der Code des Buchstaben Z. Will man nun etwa eine deutsche Tastaturbelegung erreichen (Vertauschung von Z und Y), so schreibt man in der „language card" folgendermaßen ein:

```
POKE 126, 62
POKE 154, 63
```

Damit haben bei der Tastaturbedienung X und Y ihren Platz getauscht. Natürlich hat sich die Tastenbeschriftung dadurch nicht geändert, hier muß man erforderlichenfalls überkleben.

Zu beachten ist, daß der oberhalb der Taste angeschriebene Befehl (an der Y-Taste also „RETURN") in gleicher Weise umkodiert wird wie der dazugehörige Buchstabe. Die rot geschriebenen Befehle sowie die unterhalb der Taste angebrachten Funktionen können separat umkodiert werden.

8.1.3 Aufruf selbstdefinierter Befehle

Bei der Abarbeitung eines Programms bzw. vor der Ausführung eines Befehls überprüft der ZX 81, um welchen Befehl es sich handelt. Hat er dies herausgefunden, springt er im ROM auf die Adresse, wo die Maschinenroutine zur Ausführung des Befehls beginnt. Diese Anfangsadressen sind (neben anderen Informationen) in einer Tabelle abgelegt, die zwischen C48h (3144d) und CB9h (3257d) liegt. In der nachfolgenden Auflistung sind die Sprungziele für die jeweiligen Befehle markiert. Die Sprungadresse erhält man, wenn man die aufeinanderfolgenden Werte W1 und W2 in folgender Weise verknüpft:

Startadresse (Befehl) = W1 + 256 × W2

```
3147 --> 6
3148 --> 0
3149 --> 129  } GOTO
3150 --> 14   }
3151 --> 6
3152 --> 222
3153 --> 5
3154 --> 171  } IF
3155 --> 13   }
3156 --> 6
3157 --> 0
3158 --> 181  } GOSUB
3159 --> 14   }
3160 --> 0
3161 --> 220  } STOP
3162 --> 12   }
3163 --> 0
3164 --> 216  } RETURN
3165 --> 14   }
3166 --> 4
3167 --> 20
3168 --> 6
3169 --> 223
3170 --> 6
3171 --> 5
3172 --> 185  } FOR
3173 --> 13   }
3174 --> 4
3175 --> 0
3176 --> 46   } NEXT
3177 --> 14   }
3178 --> 5
3179 --> 207  } PRINT
3180 --> 10   }
3181 --> 1
3182 --> 0
3183 --> 233  } INPUT
3184 --> 14   }
3185 --> 5
3186 --> 9    } DIM
3187 --> 20   }
3188 --> 5
3189 --> 106  } REM
3190 --> 13   }
3191 --> 0
3192 --> 195  } NEW
3193 --> 3    }
3194 --> 3
3195 --> 175  } RUN
3196 --> 14   }
3197 --> 3
```

```
3198 --> 48  } LIST
3199 --> 7   }
3200 --> 6
3201 --> 26
3202 --> 6
3203 --> 0
3204 --> 146 } POKE
3205 --> 14  }
3206 --> 3
3207 --> 108 } RAND
3208 --> 14  }
3209 --> 5
3210 --> 64  } LOAD
3211 --> 3   }
3212 --> 5
3213 --> 246 } SAVE
3214 --> 2   }
3215 --> 0
3216 --> 124 } CONT
3217 --> 14  }
3218 --> 0

3219 --> 154 } CLEAR
3220 --> 20  }
3221 --> 0
3222 --> 42  } CLS
3223 --> 10  }
3224 --> 6
3225 --> 26
3226 --> 6
3227 --> 0
3228 --> 175 } PLOT
3229 --> 11  }
3230 --> 6
3231 --> 26
3232 --> 6
3233 --> 0
3234 --> 175 } UNPLOT
3235 --> 11  }
3236 --> 0
3237 --> 14  } SCROLL
3238 --> 12  }
3239 --> 6

3240 --> 0
3241 --> 50  } PAUSE
3242 --> 15  }
3243 --> 0
3244 --> 43  } SLOW
3245 --> 15  }
3246 --> 0
3247 --> 35  } FAST
3248 --> 15  }
3249 --> 0
3250 --> 105 } COPY
3251 --> 8   }
3252 --> 5
3253 --> 203 } LPRINT
3254 --> 10  }
3255 --> 3
3256 --> 44  } LLIST
3257 --> 7   }
```

Prog. 13

Auch hier wieder ein Beispiel zur Erläuterung: Adresse 3241d enthält 50d, 3242d den Wert 15d. Erkennt der ZX 81 in einem Programm bzw. im Direktmodus den Befehl PAUSE, so springt er ein Unterprogramm an, das auf Adresse (50 + 256 × 15) = 3890d (0F32h) beginnt.

Will man den Befehl PAUSE beispielsweise zur Druckerinitialisierung zweckentfremden, so müssen die folgenden Befehle eingegeben werden:

POKE 3241, 11
POKE 3242, 42

Wird nun der PAUSE-Befehl identifiziert, so verzweigt das Programm zur Subroutine auf 10763 (11 + 256×42 = 10763). Dies ist der Anfang des Unterprogramms, mit dem der Drukker initialisiert wird.

In entsprechender Weise kann man auch andere selbsterstellte Routinen aufrufen. Ein Beispiel hierfür ist das nachfolgend vorgestellte Unterprogramm zur Bildschirminvertierung. Das REM-Statement enthält zuerst 21 beliebige Zeichen, an deren Stelle die angegebenen Werte gepoket werden. Speichert man das Programm nach der Eingabe mit

GOTO 700

ab, so ist es bei erneutem Laden selbstanlaufend. Dabei kopiert es sich in den Speicherbereich der „language card" ab 8192d. Da das Programm nur relative Sprünge verwendet, ist es auch in einem anderen Speicherbereich lauffähig. Um es anstelle einer originalen ZX-81-Funktion zu verwenden, muß in der erläuterten Weise ein Zeiger aus der oben aufgeführten Tabelle auf 8192d „verbogen" werden.

Zum besseren Verständnis ist das Programm auch in disassemblierter Form aufgelistet. Seine Funktion beruht darauf, daß bei einem Zeichen im Bildschirmspeicher das führende Bit

invertiert wird (XOR (HL)). Dadurch erhält man zu einem eingespeicherten Zeichen die Inversdarstellung.

```
 100 REM ▞/E£RND■i?7/ POKE 7( LE
T TAN  ▒
 200 FOR N=16514 TO 16535
 300 POKE (N-8322),PEEK N
 400 NEXT N
 500 NEW
 600 STOP
 700 SAVE "INVERS"
 800 GOTO 200
```

```
16514 ---> 6
16515 ---> 24
16516 ---> 42
16517 ---> 12
16518 ---> 64
16519 ---> 126
16520 ---> 254
16521 ---> 118
16522 ---> 40
16523 ---> 7
16524 ---> 62
16525 ---> 128
16526 ---> 174
16527 ---> 119
16528 ---> 35
16529 ---> 24
16530 ---> 244
16531 ---> 35
16532 ---> 16
16533 ---> 241
16534 ---> 201
16535 ---> 0
```

```
4082:06 18      LD   B,18
4084:2A 0C 40   LD   HL,(400C)
4087:7E         LD   A,(HL)
4088:FE 76      CP   76
408A:28 07      JR   Z,4093
408C:3E 80      LD   A,80
408E:AE         XOR  (HL)
408F:77         LD   (HL),A
4090:23         INC  HL
4091:18 F4      JR   4087
4093:23         INC  HL
4094:10 F1      DJNZ 4087
4096:C9         RET
```

Prog. 14

In gleicher Weise ist es möglich, Routinen des ZX 81 zu erweitern oder zu verbessern. Beispielsweise läßt sich eine eigene SCROLL- oder auch LOAD-Routine in den BASIC-Interpreter einbinden. Der Phantasie sind keine Grenzen gesetzt.

8.2 Tonerzeugung

8.2.1 Tonerzeugung über Software-Schalter (S/R-Flipflop)

Grundplatine I enthält einen, die „language card" sogar zwei frei verfügbare Software-Schalter mit S/R-Flipflops. Diese Schalter können, falls nötig unter Zwischenschaltung eines Transistors, einen (hochohmigen) Lautsprecher betreiben. Mit Hilfe eines geeigneten Programms

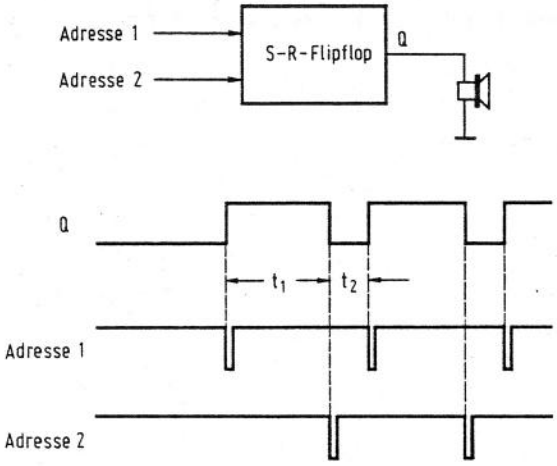


Abb. 8.1 Software-Tonerzeugung mit dem S/R-Flipflop

ist der ZX 81 in der Lage, Töne zu erzeugen. Dadurch lassen sich akustische Marken in einem Programmablauf einbauen, ja sogar eine Miniorgel wird möglich.

Das Grundprinzip dieser Tonerzeugung ist einfach. Der Software-Schalter wird durch das Tonprogramm in schneller Folge hin- und hergeschaltet. Da es sich um ein S/R-Flipflop handelt, läßt sich das Tastverhältnis beliebig einstellen *(Abb. 8.1)*.

Mit dem folgenden kleinen Programm kann ein Ton erzeugt werden:

```
100 POKE Adresse 1,0
200 POKE Adresse 2,0
300 GOTO 100
```

Befindet sich der ZX 81 im SLOW-Modus, so klingt der Ton ein wenig jaulend. Der Grund liegt darin, daß der Computer 50mal pro Sekunde seine Programmabarbeitung unterbricht, um ein Fernsehbild darzustellen. Wird auf FAST-Modus umgeschaltet, so ist das Jaulen weg, gleichzeitig wird auch der Ton höher. Eine definierte Tonerzeugung ist also nur im FAST-Modus möglich.

Man erkennt unschwer, daß die Tonfrequenz durch die Zeiten t1 und t2 in *Abb. 8.1* bestimmt wird. Für die Grundfrequenz gilt:

$f = 1(t1 + t2)$.

Von Grundfrequenz spricht man, weil es sich hier um eine stark oberwellenhaltige Schwingung handelt.

Die Frequenz wird also durch die Zeit bestimmt, die zum Abarbeiten der Programmbefehle benötigt wird. Man kann das leicht testen, indem man zwischen die POKE-Befehle weitere (beliebige) Befehle einschiebt, z. B.:

```
150 LET A =0
250 LET A =0
```

Die Frequenz wird dadurch erniedrigt.

Im SLOW-Modus wird zusätzlich Zeit für die Bildschirm-Bedienung benötigt, so daß die Tonhöhe stets geringer als im FAST-Modus ist.

Will man den Ton erhöhen, so sieht man sehr schnell, daß das nicht geht. Nach dem Löschen der Zeilen 150 und 250 hat man bereits das schnellstmögliche BASIC-Programm zur Tonerzeugung. Eine weitere Tonerhöhung ist nur durch die Verwendung eines Maschinenprogramms möglich, BASIC-Programme sind zu langsam.

In *Abb.8.2* ist anhand eines Flußdiagramms gezeigt, wie ein solches Maschinenprogramm aufgebaut ist. Die auf der linken Seite angegebenen Adressen ergeben sich, wenn das Programm wie üblich in einem REM-Statement untergebracht wird.

Das Maschinenprogramm ist im Prinzip ähnlich aufgebaut wie das BASIC-Programm. Den POKE-Befehlen von dort entsprechen die Ladebefehle in 16520 bzw. 16527 (LD (NN),A) zum Umschalten der S/R-Flipflops. Da das Maschinenprogramm zu schnell ist, müssen zwischen die Schaltbefehle noch Programmschleifen zur Verzögerung eingebaut werden, sonst liegt der erzeugte Ton oberhalb des Hörbereichs. In diesen Verzögerungsschleifen wird der Inhalt des B-Registers bis auf Null dekrementiert (heruntergezählt). Der entsprechende Anfangswert des B-Registers (Speicherstelle 16517d bzw. 165244d) bestimmt die hierfür nötige Zeit und damit die Frequenz. Nach jeder Schwingungsperiode wird der Akku-Inhalt dekrementiert, sein Anfangswert (Speicherstelle 16515d) bestimmt also die Tonlänge (genauer: die Anzahl der Schwingungszüge, die Tonlänge ist also frequenzabhängig).

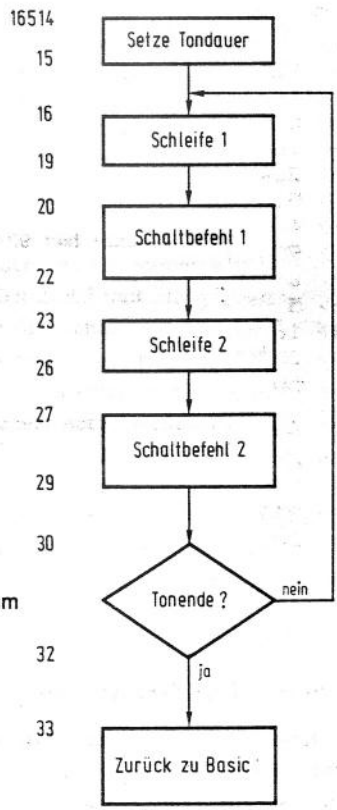


Abb. 8.2 Flußdiagramm zum Tonprogramm

Eine Berechnung der Frequenz und Tonlänge ist im Prinzip möglich, indem man die Zeiten, die zur Abarbeitung der einzelnen Maschinenbefehle benötigt werden, aufaddiert. Da der ZX 81 mit einer Taktfrequenz von 3,23 MHz läuft, dauert z. B. die Abarbeitung des DEC-Befehls 1,24 μs. Weil aber bedingte Sprungbefehle (JRNZ, DJNZ) verschieden lange benötigen, je nachdem ob die Sprungbedingung erfüllt ist oder nicht, ist dieses Verfahren nicht ganz einfach, experimentieren führt meist schneller zum Ziel.

Nachstehend ist das Programm in disassemblierter sowie dezimaler Form (zum direkten Eintippen) aufgeführt.

```
4082:3E FF        LD    A,FF
4084:06 05        LD    B,05
4086:10 FE        DJNZ  4086
4088:32 04 5C     LD    (5C04),A   SET-Adresse
408B:06 05        LD    B,05
408D:10 FE        DJNZ  408D
408F:32 05 5C     LD    (5C05),A   RESET-Adresse
4092:3D           DEC   A
4093:20 EF        JR    NZ,4084
4095:C9           RET
```

Prog. 15

```
16514 --> 62
16515 --> 255
16516 --> 6
16517 --> 5
16518 --> 16
16519 --> 254
16520 --> 50
16521 --> 4      } SET-Adresse, hier: 92·256+4 = 23556 d
16522 --> 92     }
16523 --> 6
16524 --> 5
16525 --> 16
16526 --> 254
16527 --> 50
16528 --> 5      } RESET-Adresse, hier: 92·256+5 = 23557 d
16529 --> 92     }
16530 --> 61
16531 --> 32
16532 --> 239
16533 --> 201
```

Prog. 16

8.2.2 Tonmarken in BASIC-Programmen

Akustische Marken im Ablauf eines Programmes besitzen einen hohen Aufmerksamkeitswert. So kann etwa bei einer fehlerhaften Eingabe (wenn beispielsweise eine Zahl einen bestimmten Maximalwert überschreitet) ein Warnton ausgegeben werden. Dabei ist es beson-

ders günstig, wenn das hierzu benützte Maschinenprogramm überhaupt nicht in Erscheinung tritt, sondern „versteckt" wird.

Dies läßt sich erreichen, indem man die Systemvariable RAMTOP herabsetzt, der Computer „vergißt" dadurch gewissermaßen einen Teil seines Speichers. In diesem „vergessenen" Speicherbereich wird anschließend das Maschinenprogramm untergebracht.

Wenn bei der Grundplatine I die Speichererweiterung voll bestückt ist, besitzt RAMTOP den Wert 23552. Durch

POKE 16388, 204 und
POKE 16389, 91

Anschließend wird NEW eingegeben, danach das Programm (im REM-Statement der Zeile 100 enthalten) vom Band geladen. Nun fügt man noch folgende Programmfolge an:

```
200 FOR N =16514 TO 16533
300 POKE (N + 6986), PEEK N
400 NEXT N
```

Startet man dieses Programm, so wird das Maschinenprogramm in den „vergessenen" Speicherbereich kopiert. Es ist dort ohne Probleme lauffähig, da nur relative Sprünge verwendet werden, so daß keine Sprungadressen geändert werden müssen.

Nach erneutem Eingeben von NEW (vorher kann man natürlich das Tonprogramm mit Umkopierteil abspeichern) steht der Computer für ein neues BASIC-Programm zur Verfügung, der Speicher ist ja gewissermaßen „leer". Im Rahmen dieses Programms kann die Tonmarke mit

```
LET A = USR 23500
```

aufgerufen werden.

Soll die Tonmarke im SLOW-Modus verwendet werden, empfiehlt es sich, die Werte für die Tonhöhe zu ändern (Inhalt der Speicherstellen 23503 und 23510: jeweils 50).

Zu beachten ist, daß bei einem Abspeichern des nun erstellten BASIC-Programms das Ton-Programm nicht mit abgespeichert wird, da oberhalb vom RAMTOP liegende Speicherbereiche hierbei nicht berücksichtigt werden. Das Ton-Programm muß also jeweils vor dem BASIC-Programm in der beschriebenen Weise geladen werden. Wird dies nicht beachtet und verwendet das BASIC-Programm den Maschinenprogrammaufruf (LET A =USR 23500), stürzt das Programm beim Lauf ab.

Bei Verwendung der Grundplatine II mit „language card" wird das Ton-Programm in entsprechender Weise in den zusätzlichen RAM-Speicher (oberhalb 8191) geladen. Falls hier das Pseudo-ROM eingesetzt wird, läßt es sich in diesem Speicher „permanent" unterbringen und muß nicht jedesmal vom Band geladen werden.

8.2.3 Tonerzeugung über JK-Flipflop

Das S/R-Flipflop auf der Grundplatine I bzw. der „language card" erlaubt die Erzeugung von Impulsfolgen mit unsymmetrischem Tastverhältnis. Andererseits muß das dazugehörige Programm zwei Zeitschleifen enthalten, jeweils für die Ein- bzw. Ausschaltdauer getrennt.

Falls nur ein symmetrisches Tastverhältnis benötigt wird, wird das Maschinenprogramm unnötig aufwendig. Dies läßt sich ändern, wenn statt des S/R-Flipflops ein JK-Flipflop der „language card" (Adresse 15374 bzw. 15375) verwendet wird. Es wirkt als „toggle switch", d. h. jeder Schreibbefehl (auf ein und derselben Adresse) schaltet den Zustand des Flipflops um.

Ein Ton-Programm kann analog wie oben dargestellt aufgebaut werden. Es ist nur eine Zeitschleife zu verwenden, wobei stets dieselbe Adresse umgeschaltet wird.

8.2.4 Tonerzeugung über OUT-Port bzw. Centronics-Schnittstelle

Durch Ein- oder Ausschalten einzelner Bits eines OUT-Ports (Grundplatine I oder Zusatzkarte) bzw. einer Centronics-Schnittstelle (Grundplatine II) läßt sich ebenfalls eine tonfrequente Wechselspannung erzeugen. Besonders elegant gelingt dies durch fortlaufendes Dekrementieren bzw. Inkrementieren eines CPU-Registers und Ausgabe des Register-Inhalts über einen Port.

Aus *Abb.8.3* ist zuerkennen, daß an der Bit-Position 0 des Ports die höchste Frequenz abgenommen werden kann, während Bit 1, 2, 3 ... jeweils Impulsfolgen der halben, viertel, achtel ... Frequenz führen. Damit eignet sich diese Methode insbesondere, um auf dem ZX 81 ein Musikinstrument zu realisieren, da aufeinanderfolgende Oktaven im Frequenzverhältnis 1:2 stehen.

Ein 50-Ω-Lautsprecher bzw. eine 32-Ω-Postkapsel kann über einen Vorwiderstand von 220 ... 470 Ω (Widerstandswert je nach gewünschter Lautstärke) direkt zwischen die entsprechende Bitposition des Ports und Masse geschaltet werden.

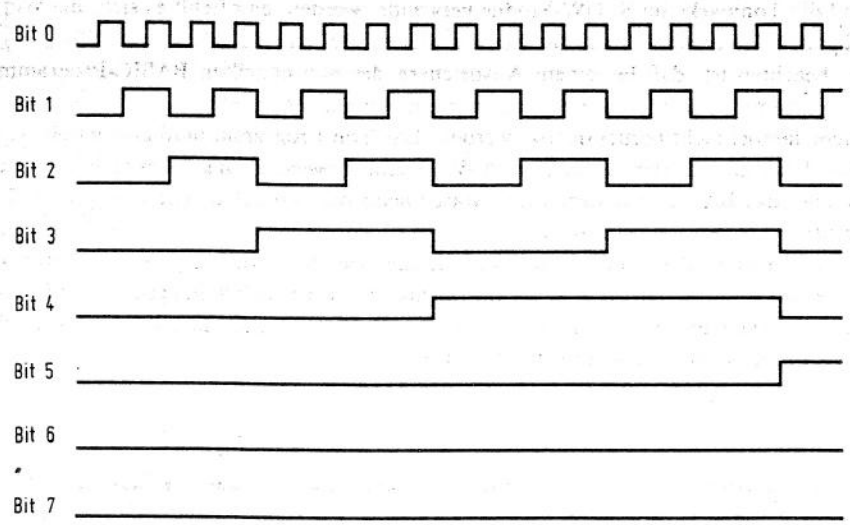


Abb. 8.3 Impulsdiagramm zur Software-Tonerzeugung (OUT-Port)

In BASIC läßt sich die Funktion mit dem folgenden kleinen Programm testen:

```
100 FOR N = 0 to 255
200 POKE Adresse, N
300 NEXT N
400 GOTO 100
```

Auch hier bietet ein Maschinenprogramm wieder Vorteile. Im nächsten Kapitel ist es zur Realisierung einer Mini-Orgel mit dem ZX 81 eingesetzt.

8.2.5 Miniorgel

Will man auf der Tastatur des ZX 81 ein elektronisches Musikinstrument realisieren, so darf der Ton nur erklingen, solange die entsprechende Taste gedrückt wird. Darum muß im Maschinenprogramm zur Tonerzeugung eine Tastaturabfrage vorhanden sein. Zum besseren Verständnis dieses Programms sei nochmals auf Kapitel 1.4.2 hingewiesen, wo die Funktion der Tastaturabfrage ausführlich erläutert wird.

Wird der ZX 81 als Miniorgel verwendet, sollen auf der untersten Tastaturreihe die Stammtöne, in der darüberliegenden Reihe die entsprechenden Halbtöne gegriffen werden können (analog einer Klaviertastatur). Das bedeutet, daß für die Tastaturabfrage die Adreßleitungen A15, A14, A9 und A8 getestet werden müssen. Hierzu gehört das Ausgabe-Bitmuster 0011 1100, d. h. 60d bzw. 3Ch. Falls eine der gewünschten Tasten gedrückt wird, ist der Zahlenwert des eingelesenen Bitmusters ungleich 127. Diese Tatsache nützt das nachfolgend aufgeführte Maschinenprogramm aus. Ausgabeport ist hier der Parallelport der Grundplatine I (Adresse 23558). Bei Verwendung eines anderen Ports ist diese Adresse in entsprechender Weise abzuändern (Speicherstelle 16522d und 16523d). Eingegeben wird das Programm in ein REM-Statement ab Adresse 16514.

```
4082:00          NOP
4083:00          NOP
4084:00          NOP
4085:00          NOP
4086:00          NOP
4087:00          NOP
4088:00          NOP
4089:00          NOP
408A:21 06 5C    LD    HL,5C06
408D:06 00       LD    B,00
408F:3A 86 40    LD    A,(4086)
4092:4F          LD    C,A
4093:3E 3C       LD    A,3C
4095:DB FE       IN    A,(FE)
4097:D6 7F       SUB   7F
4099:C8          RET   Z
409A:0D          DEC   C
409B:20 FD       JR    NZ,409A
409D:70          LD    (HL),B
409E:05          DEC   B
409F:18 EE       JR    408F
```

Prog. 17

```
16514 --> 0
16515 --> 0
16516 --> 0
16517 --> 0
16518 --> 0
16519 --> 0
16520 --> 0
16521 --> 0
16522 --> 33
16523 --> 6
16524 --> 92
16525 --> 6
16526 --> 0
16527 --> 58
16528 --> 134
16529 --> 64
16530 --> 79
16531 --> 62
16532 --> 60
16533 --> 219
16534 --> 254
16535 --> 214
16536 --> 127
16537 --> 200
16538 --> 13
16539 --> 32
16540 --> 253
16541 --> 112
16542 --> 5
16543 --> 24
16544 --> 238
```

Prog. 18

In Adresse 16518d (4086h) wird der für die Tonhöhe maßgebliche Zahlenwert vor Aufruf des Maschinenprogramms eingegeben. Das eigentliche Programm beginnt bei Adresse 16522d (408Ah), Aufruf mit

LET A = USR 16522

In den Programmzeilen 16531d (4093h) ... 16537d (4099h) erfolgt die Überprüfung, ob eine der gewünschten Tasten gedrückt ist. Falls dies nicht der Fall ist, bekommt Akku minus 127 den Wert Null, so daß der Rücksprung aus dem Maschinenprogramm erfolgt (RET Z).

Für die Tonerzeugung sind zwei Schleifen wichtig: 16527d (408Fh) ... 16530d (4092h) laden in das Register C den für die Tonhöhe wichtigen Zahlenwert aus der Speicherstelle 16518d (4086h). Nachdem der Inhalt von Register C auf Null heruntergezählt ist, wird der Inhalt des B-Registers (zu Programmbeginn: Null) über den Port ausgegeben. Anschließend wird das B-Register dekrementiert. Dann wiederholt sich der gesamte Vorgang.

Als Hilfestellung und Anregung für eigene Experimente sind in *Abb. 8.4* einige Zahlenwerte (Inhalt der Speicherstelle 16518d) und die dazugehörigen Frequenzwerte (Tonabnahme an Bit 2 des Ports) angegeben. Bei einer Tonabnahme an den anderen Bitleitungen erhält man, wie bereits erläutert, die entsprechendenen Vielfachen dieser Frequenzen.

Besonders hervorgehoben sind in Abb.8.4 die zu den Tönen der zwischen c' und c'' liegenden Oktave (näherungsweise) gehörenden Frequenzen. Dabei ist zu beachten, daß die Taktfrequenz des ZX 81 nur durch ein Keramikfilter stabilisiert ist, so daß es auch Abweichungen infolge von Exemplarstreuungen geben kann.

Um den ZX 81 als elektronische Miniorgel zu benützen, muß das Maschinenprogramm in ein kleines BASIC-Programm eingebunden werden.

```
 100 REM 000000000000000000000000
0000000000000000000000000000000000
0000000000000000000000000000000000
 200 IF INKEY$<>"" THEN GOTO 200
 300 LET A=CODE INKEY$
 400 IF A=63 THEN POKE 16518,187
 500 IF A=61 THEN POKE 16518,166
 600 IF A=40 THEN POKE 16518,149
 700 IF A=59 THEN POKE 16518,139
 800 IF A=39 THEN POKE 16518,123
 900 IF A=51 THEN POKE 16518,111
1000 IF A=50 THEN POKE 16518,98
1100 IF A=27 THEN POKE 16518,91
1200 IF A=56 THEN POKE 16518,176
1300 IF A=41 THEN POKE 16518,156
1400 IF A=44 THEN POKE 16518,131
1500 IF A=45 THEN POKE 16518,116
1600 IF A=47 THEN POKE 16518,104
1700 LET A=USR 16522
1800 GOTO 200
```

Prog. 19

Zahlenwert für Schleife	Frequenz (Hz)	Ton
255	195	
250	199	
200	247	
190	259	
187	264	c′
180	274	
176	280	cis′
170	290	
166	296	d′
160	307	
156	315	dis′
150	327	
149	329	e′

Zahlenwert für Schleife	Frequenz (Hz)	Ton
140	350	
139	352	f′
131	373	fis′
130	376	
123	397	g′
120	406	
116	420	gis′
111	438	a′
110	442	
104	467	ais′
100	485	
98	494	h′
91	530	c″

Abb. 8.4 Frequenztabelle zur Tonerzeugung mit dem OUT-Port (Tonabnahme an Bit 2, ZX 81 im FAST-Modus)

Das Maschinenprogramm ist wieder wie üblich in einem REM-Statement (Zeile 100) untergebracht. In Zeile 200 wartet der Computer solange, bis eine Taste gedrückt wird.

Der Variablen A wird der Code-Wert der entsprechenden Taste zugewiesen (Zeile 300) und in Abhängigkeit von diesem Wert die der gewünschten Frequenz entsprechende Zahl in die Speicherstelle 16518d eingeschrieben.

In *Abb.8.5* ist die „Orgeltastatur" gezeigt. Für einen ersten Probelauf eignet sich das nachfolgende, sicher allgemein bekannte Liedchen. Ein Strich unter dem Buchstaben bedeutet, daß der Ton etwas länger gehalten wird:

Z X C V B B N N N N B N N N N B V V V V C C X X X X Z

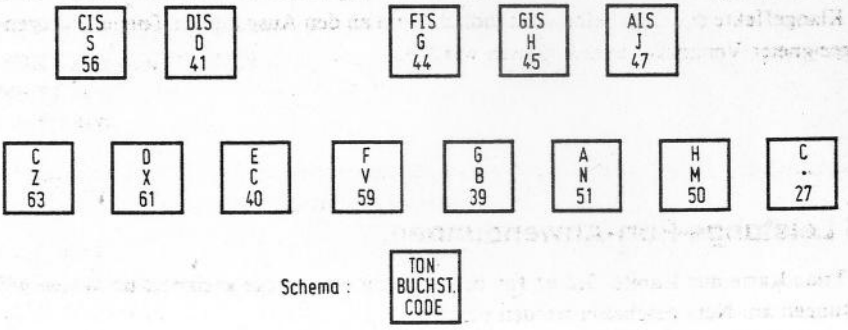


Abb. 8.5 „Orgeltastatur" des ZX 81

Wie in Kapitel 6 bereits näher ausgeführt, wird bei der Hardware-Tonerzeugung die Frequenz durch das auf die betreffende Adresse geschriebene Datenwort bestimmt. Im Regelfall wird man mit einem BASIC-Programm auskommen, da keine besonderen Anforderungen an die Geschwindigkeit gestellt werden.

Eine Miniorgel läßt sich in anloger Weise zur Software-Lösung erstellen. Dabei kann die Orgel ohne weiteres mehrstimmig ausgeführt werden, indem mehrere Tongeneratoren benützt werden. Nachfolgend ist ein Programmbeispiel aufgeführt, das Anregung für eigene Experimente geben soll:

```
100 IF INKEY$ <> " " THEN GOTO 300
200 POKE Adresse, 0
300 LET A = CODE INKEY$
400 IF A = XX THEN POKE Adresse, YY
.
.
.
.
.
ZZZ GOTO 100
```

Dabei bedeutet XX die gewünschte Taste, YY den Datenwert für die gewünschte Taste (ausprobieren, der Tonbereich des Oszillators muß natürlich hardwaremäßig so ausgelegt werden, daß alle gewünschten Töne erzeugt werden können). ZZZ ist die Zeilennummer der letzten Programmzeile.

Auch Klangeffekte lassen sich ausprobieren. Das folgende Programm ist auf den Oszillator mit C-Umschaltung abgestimmt (R5 = 470 Ω, R6 = 15 kΩ, Kondensatoren wie angegeben), Programmablauf im FAST-Modus:

```
100 FOR N = 10 TO 15
200 POKE Adresse, N
300 NEXT N
400 GOTO 100
```

Klang Grillenzirpen schon einmal schöner? Durch Experimentieren lassen sich die vielfältigsten Klangeffekte erzielen. Selbstverständlich kann an den Ausgang der Tongeneratoren auch ein geeigneter Verstärker angeschlossen werden.

8.3 Leistungs-Port-Anwendungen

Die Triac-Karte aus Kapitel 3.2 ist für beliebige Einsatzzwecke geeignet, bei denen größere Leistungen am Netz geschaltet werden müssen.

In Maschinenprogrammen wird man mit Vorteil die Befehle zum Setzen und Löschen von Einzelbits verwenden. So läßt sich beispielsweise die gewünschte Leitung b (b = 0 ... 7) mit

folgendem Befehl einschalten:

SET b, (HL)

Die Speicherstelle, unter der die Triac-Karte erscheint, wird über das Registerpaar HL adre siert.

Ausschalten kann man in analoger Weise mit dem Befehl:

RES b, (HL)

Möchte man Schaltaufgaben in BASIC programmieren, so kann man sich die Gesetzm ßigkeiten der Dezimal-Binär-Wandlung zunutze machen. Die einer Binärzahl zugeordne Dezimalzahl findet man als

$$ZAHL = D0 \times 1 + D1 \times 2 + D2 \times 4 + D3 \times 8 + D4 \times 16 + D5 \times 32 + D6 \times 64 + D7 \times 128.$$

D0 ... D7 symbolisieren die entsprechenden Datenleitungen und besitzen die Werte 0 oder Leitung D5 läßt sich beispielsweise einschalten mit dem Befehl:

POKE Adresse, 32

Sollen mehrere Leitungen gleichzeitig bedient werden, so geschieht dies in entsprechend Weise:

POKE Adresse, 49

schaltet D0, D4 und D5 ein $(1 + 16 + 32 = 49)$.

Soll im letztgenannten Beispiel anschließend D4 ausgeschaltet werden, so besorgt dies d Befehl:

POKE Adresse, 33
$(1 + 32 = 33)$.

Auch hier führt Experimentieren im Regelfall schnell zum Ziel. Als unterhaltsame Anwe dungen seien noch zwei Beispiele für „Lichtspiele" (Ansteuerung von Glühlampen/Schei werfern über je eine Datenleitung) angegeben.

Eine „Zufalls-Lichtorgel" läßt sich verwirklichen, indem man das ROM (oder sonst eine Speicherbereich) ausliest und die Daten über den Leistungs-Port ausgibt.

```
100 FOR N=0 TO 8191
200 POKE Adresse, PEEK N
300 NEXT N
400 GOTO 100
```

Will man ein Lauflicht produzieren (wenn etwa acht Glühbirnen nebeneinander angeordn sind), so läßt sich folgendes Programm verwenden

```
100 LET N=1
200 POKE Adresse, N
300 LET N=2*N
400 IF N<129 THEN GOTO 200
500 GOTO 100
```

8.4 Steuerung einer Modelleisenbahn (IN/OUT-Ports)

Das Steuern einer Modelleisenbahn-Anlage mit Hilfe eines Computers ist ein sehr komplexes und unter Umständen auch arbeitsintensives Betätigungsfeld. Daher können an dieser Stelle nur einige Anregungen gegeben werden, wie ein solcher Einsatz geplant und ausgeführt werden kann. Die nachfolgenden Beispiele sind auf Märklin-H0-Bahnen (Wechselstromsystem) abgestimmt. Bei Verwendung anderer Fabrikate sind unter Umständen Abänderungen notwendig.

Für den Computereinsatz ergeben sich mehrere Schwerpunkte:
a) Darstellung von Schienen- und Belegungsplan auf dem Fernsehbildschirm
b) Steuern der Zugbewegungen auf einzelnen Fahrbereichen über OUT-Ports
c) Schalten von Weichen und Signalen
d) Erfassen der Zugbewegungen über Kontaktgleisstücke und IN-Ports.

Im folgenden soll auf die Punkte b), c) und d) näher eingegangen werden. Geeignete Hardware-Schaltungen sind die in den Kapiteln 3.1, 3.3 und 3.4 beschriebenen Ports.

Normalerweise ist eine Modelleisenbahn-Anlage in eine Reihe von Fahrbereichen unterteilt, bei denen jeweils separat die Betriebsspannung ein- und ausgeschaltet werden kann. Meist werden diese Fahrbereiche von Signalen bedient. Um in einem solchen Fahrbereich die Fahrspannung vom Computer zu schalten, schließt man an die einzelnen Bit-Leitungen eines OUT-Ports jeweils ein Relais an. Gegebenenfalls muß noch eine Transistorstufe eingefügt werden, die den nötigen Treiberstrom zur Relaisansteuerung aufbringt. Über den Arbeitskontakt des Relais wird die Fahrspannung ein- bzw. ausgeschaltet.

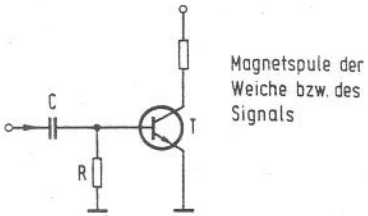
Sollten Magnetartikel, also Weichen und Signale, geschaltet werden, so dürfen sie nur kurzzeitig betätigt werden. Dies läßt sich softwaremäßig realisieren, indem die betreffende Port-Leitung kurze Zeit nach dem Anschalten wieder abgeschaltet wird (Zeitschleife). Da dadurch aber der Computer bei einer größeren Anlage sehr stark beschäftigt wird, empfiehlt es sich, die entsprechende Impulsgestaltung hardwaremäßig zu erreichen. Dies ist durch ein Differenzieren der Schaltimpulse über ein RC-Glied leicht zu erreichen (*Abb.8.6*). Der Wert des Kondensators wird durch Versuche optimal an die verwendeten Signale oder Weichen angepaßt.

Das Steuerprogramm kann in BASIC oder Maschinensprache erstellt werden, je nach Anforderungen an die Arbeitsgeschwindigkeit des Programms. Dabei gelten die in Kapitel 8.3 gemachten Ausführungen sinngemäß.

Um die Zugbewegungen zu erfassen, lassen sich Kontaktgleisstücke verwenden, die über IN-Ports abgefragt werden. Diese Gleisstücke schließen beim Überfahren einen Kontakt nach Masse kurz. Wird der Kontakt über einen Widerstand von 1kΩ an +5 V gelegt, so läßt sich ein H/L-Signal gewinnen. Problematisch ist dabei allerdings, daß dieses Signal nur für einen sehr kurzen Augenblick aktiv ist. Wird sein Zustand während dieser kurzen Zeitspanne nicht abgefragt, so erfolgt fälschlicherweise keine Registrierung der Zugbewegung. Abhilfe wäre möglich, wenn jedes Überfahren eines Kontaktes einen Interrupt auslösen könnte. Leider ist beim ZX 81 dieser Weg versperrt, da Interrupt-Techniken vom Betriebssystem verwendet werden. Daher bietet sich eine Speicherung der Kontaktinformation an, wie sie mit der S/R-IN-Karte (Kapitel 3.4) möglich ist.

Abb.8.7 erläutert anhand eines Ausschnitts aus einer Modellanlage, wie dabei vorzugehen ist. Das Kontaktgleisstück schließt je nach Fahrtrichtung zwei verschiedene Schalter, a) bzw.

Abb. 8.6 Ansteuerung von Magnetartikeln



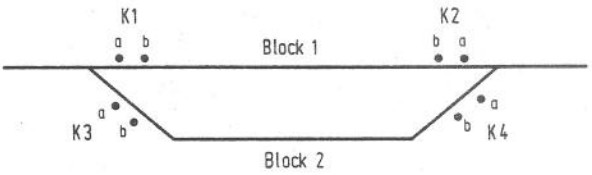


Abb. 8.7 Erfassung von Zugbewegungen in einer Modelleisenbahn-Anlage

b). Angenommen, ein Zug fährt von links kommend in den Block 1 ein. Dann setzt K1a das S/R-Flipflop für Block 1. Verläßt der Zug Block 1 nach rechts, so wird über K2b das Flipflop zurückgesetzt. Umgekehrt muß ein von rechts nach links fahrender Zug über K2a das Flipflop setzen und über K1b rücksetzen. K1a und K2a bzw. K1b und K2b müssen also über eine ODER-Funktion verknüpft werden, das ist hier durch einfaches Parallelschalten der jeweiligen Kontakte möglich. Der Schaltzustand des S/R-Flipflop und damit die Verlegung des Blocks kann also über die Abfrage der entsprechenden Bit-Position der IN-Karte registriert werden.

Wird auf dem Fernsehschirm der Gleisverlauf der Anlage dargestellt, so lassen sich belegte und unbelegte Blöcke sowie Weichen- und Signalstellungen anzeigen. Allerdings ist man wegen der nicht sehr hoch auflösenden Grafik des ZX 81 auf kleinere Anlagen beschränkt. Durch ein entsprechendes Programm ist auch ein vollautomatischer Betrieb möglich, der hierfür notwendige Programmieraufwand ist aber nicht unerheblich.

8.5 Einsatz der Paddle-Karte

8.5.1 Steuerknüppel für Spiele

In Kapitel 4.2 ist bereits das Grundsätzliche zur Bedienung der Paddle-Karte gesagt. Der Zahlenwert, den die Karte in Abhängigkeit von der Potentiometerstellung liefert, kann zur Positionsbestimmung einer Figur oder Marke auf dem Bildschirm herangezogen werden. Da dieser Wert zwischen 0 und 255 liegen kann, muß eine Normierung auf den Wertebereich der PLOT- bzw. UNPLOT-Funktion erfolgen (Y-Richtung 0 ... 43, X-Richtung 0 ... 63). Das nachfolgende Programm zeichnet, von zwei Potentiometern bestimmt, eine Kurve auf den Bildschirm.

```
100 POKE Adresse 1, 0
200 LET X = INT ((PEEK Adresse 2)/4 - 1)
300 LET Y = INT ((PEEK Adresse 3)/6)
400 PLOT X,Y
500 GOTO 100
```

Adresse 2 bzw. 3 gehören zu den beiden Paddle-Kanälen, die mit den Hardware-Zählern ausgestattet sind.

Für Spielanwendungen sind BASIC-Programme in der Regel zu langsam. In einem Maschinenprogramm übernimmt die Rolle des POKE-Befehls zum Starten der Monoflops der Befehl

LD (nn), A.

Dabei gibt nn die Adresse an. Statt PEEK wird zum Lesen der Zählerstände der Befehl

LD A, (nn)

benützt. Der Einsprung in die PLOT-Routine erfolgt auf Adresse 0BB2h (2994d) des Monitor-ROMs. Zuvor muß das CPU-Register C mit der y-Koordinate, das Register B mit der x-Koordinate des zu plotenden Punktes geladen sein.

Zu beachten ist allerdings, daß der Wertebereich für die jeweilige Koordinate nicht überschritten wird. Bevor man nun ein aufwendiges Maschinenprogramm zur Division entwickelt, kann man die Tatsache anwenden, daß eine binäre Division durch zwei auf einfache Weise zu realisieren ist. Hierzu ist es lediglich notwendig, die Binärzahl um eine Stelle nach rechts zu verschieben. Ein Beispiel soll dies erläutern:

1010 1100 = 172
0101 0110 = 86
0010 1011 = 43
0001 0101 = 21

Im letzten Fall bleibt ein Rest, da 43 eine ungerade Zahl ist. Man erkennt es bei der Binärzahl daran, daß ihre letzte Stelle den Wert 1 hat. Wenn dieser Fehler nicht weiter stört, kann die Anpassung an die x-Koordinate durch zweimalige Ausführung des Schiebebefehls

SRL B

erreicht werden (Division durch 4). Für die y-Koordinate wird dieser Befehl dreimal ausgeführt (Division durch 8).

8.5.2 Temperaturmessung mit NTC-Widerstand

Schließt man an die Paddle-Karte statt des Potentiometers einen NTC-Widerstand an, läßt sich die Karte zur Temperaturmessung verwenden. Dabei muß darauf geachtet werden, daß die aus dem NTC-Widerstand und dem Kondensator (am Flipflop) gebildete Zeitkonstante im erlaubten Bereich liegt. Im Zweifelsfall wird ein anderer Kondensatorwert eingesetzt.

Das Programm kann analog wie in Kapitel 8.5.1 ausgeführt werden. In aller Regel ist ein BASIC-Programm völlig ausreichend. Bevor allerdings Temperaturwerte angegeben werden können, müssen die von der Paddle-Karte gelesenen Werte umgesetzt werden. Da NTC-Widerstände keinen linearen Verlauf besitzen, kann dies nicht durch Rechnung geschehen, sondern muß anhand einer Eichkurve ermittelt werden.

Hierzu legt man im Programm eine Array TEMP (256) an. Dann wird mit Hilfe eines Thermometers ermittelt, welche Temperatur zu einem bestimmten, von der Paddle-Karte gelieferten Wert gehört. Diese Werte werden in das Array eingegeben. Beispiel:

Paddle-Wert N	*Temperatur (°C)*	*TEMP (N)*
.	.	.
.	.	.
100	35,5	35.5
101	36,2	36.2
102	37,9	37.9
.	.	.
.	.	.
.	.	.

Der Ausdruck der Temperatur erfolgt dann im Programmverlauf durch:

ZZZ PRINT TEMP (T)

Der Variablen T muß zuvor der Paddle-Wert zugewiesen worden sein. ZZZ ist die Zeilennummer.

8.5.3 Helligkeitsmessung mit LDR-Widerstand

Analog zur Temperaturmessung läßt sich auch eine Helligkeitsmessung mit Hilfe eines LDR-Widerstands ausführen. Allerdings besitzen LDR-Widerstände einen Variationsbereich von etlichen MΩ bis einigen hundert Ω, so daß sich nicht der gesamte Bereich ausnützen läßt. Programmgestaltung und Eichung werden wie oben erläutert durchgeführt.

8.5.4. Grafiktablett

Will man Figuren oder beliebig geformte Kurvenzüge in eine grafische Darstellung auf dem Bildschirm übernehmen, so ist das mit Hilfe des PLOT-Befehls eine mühsame Angelegenheit. Ein Grafiktablett unter Verwendung der Paddle-Karte vereinfacht diese Arbeit wesentlich.

Abb.8.8 erläutert das Prinzip. Bei Grafikdarstellungen kann ein Punkt A des Bildschirms durch seine Koordinaten x und y bestimmt werden. Denkt man sich in der linken unteren

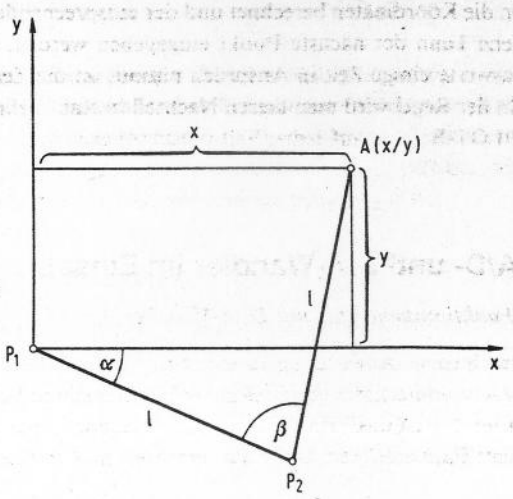


Abb. 8.8 Prinzip des Grafiktabletts

Ecke eine Stange der Länge l drehbar befestigt, an deren Ende eine weitere Stange derselben Länge ebenfalls drehbar angebracht ist, so kann man mit dieser Anordnung jeden Punkt des Bildschirms erreichen. Dabei besitzen (bei geeigneter Anwendung) die Winkel α und β eindeutige Werte. Aus der Zeichnung erkennt man die Zuordnung der Winkel zu den Koordinaten:

$x = l \times (\cos \alpha - \cos (\beta - \alpha))$
$y = l \times (\sin \alpha \times \sin (\beta - \alpha))$

Dabei werden Winkel α von der x-Achse aus im Uhrzeigersinn positiv, entgegen dem Uhrzeigersinn negativ gerechnet.

Baut man sich ein mechanisches Modell zu dieser Skizze aus Holzleisten (Länge l für die Größe der Zeichnungen geeignet wählen) und ordnet man in den Punkten P1 und P2 Potentiometer an, so besteht ein Zusammenhang zwischen dem Widerstandswert und dem Winkel. Für den Winkel β sind Werte zwischen 0° und 180° erlaubt, für α solche zwischen +90° und −90°. Im ersten Fall können also die eingelesenen Paddle-Werte 0 ... 255 direkt dem Winkel 0 ... 180° zugeordnet werden, im zweiten Fall muß vom ermittelten Paddle-Wert noch 128 subtrahiert werden. Über ein geeignetes BASIC-Programm werden zu jedem Paar von Paddle-Werten die zugehörigen Koordinaten berechnet und ein Punkt gezeichnet. Als Anregung möge das folgende Beispiel dienen:

```
100 IF INKEY$ = " " THEN GOTO 100
200 POKE Adresse 1,0
300 ALPHA = 0.0123 * (PEEK (Adresse 2) — 128)
400 BETA = 0.0123 * PEEK Adresse 3
500 X = 63 * (COS ALPHA - COS (BETA - ALPHA))
600 Y = 63 * (SIN ALPHA + SIN (BETA - ALPHA))
700 IF X > 63 THEN X = 63
800 IF Y > 43 THEN Y = 43
900 PLOT X,Y
```

Nach Eingabe von RUN führt man A auf den Punkt, der in die Grafik übernommen werden soll. Das Programm wartet in Zeile 100 auf einen Tastendruck. Nach Betätigen einer Taste werden die Koordinaten berechnet und der entsprechende Punkt auf dem Bildschirm gezeichnet. Jetzt kann der nächste Punkt eingegeben werden. Da die Berechnung der Sinus- und Cosinuswerte einige Zeit in Anspruch nimmt, ist die Zeichengeschwindigkeit nur relativ gering. In der Regel wird man diesen Nachteil in Kauf nehmen können, da eine Kurveneingabe über PLOT-Befehle auf jeden Fall umständlicher ist.

8.6 A/D- und D/A-Wandler im Einsatz

8.6.1 Funktionsgenerator mit D/A-Wandler

Für verschiedene Anwendungsfälle benötigt man Generatoren mit einstellbarer Kurvenform. Die D/A-Wandlerkarte gestattet es, solche Signale auf einfache Weise zu erzeugen.

In *Abb. 8.9* ist das Prinzip dargestellt. Nachdem man die gewünschte Kurvenform auf einem Blatt Papier aufgezeichnet hat, ermittelt man in regelmäßigen Zeitabständen Δt die mo-

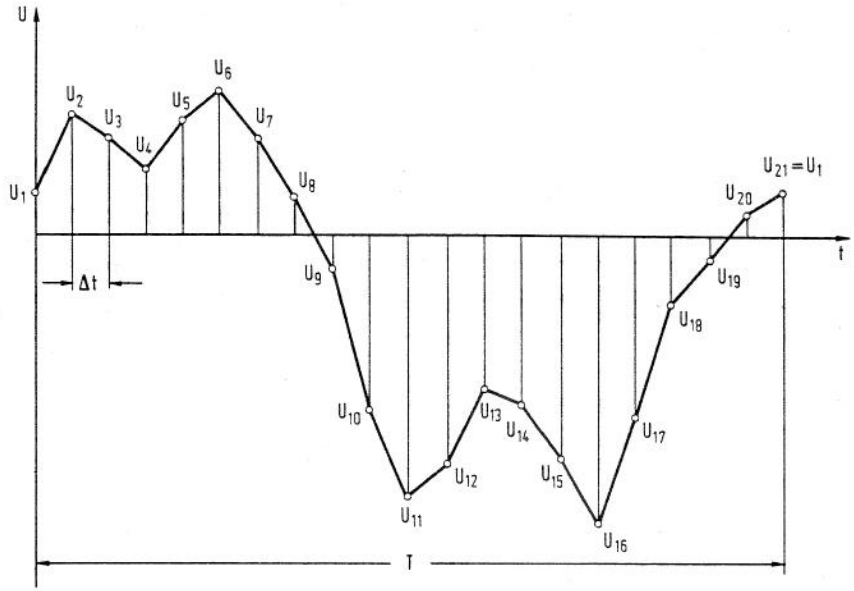


Abb. 8.9 Schwingungserzeugung mit dem D/A-Wandler

mentanen Spannungswerte U1, U2, ... Diese Spannungswerte werden in binär codierter Form im Speicher zwischen den Adressen START und ENDE abgelegt. Für das Schützen dieses Speicherbereichs gegen ungewolltes Überschreiben gelten die üblichen Regeln (REM-Statement oder oberhalb RAMTOP).

In BASIC gestaltet sich das Generatorprogramm recht einfach:

```
100 FOR N = START TO ENDE
200 POKE Adresse, PEEK N
300 NEXT N
400 GOTO 100
```

„Adresse" ist die Adresse, unter der der gewünschte D/A-Kanal angesprochen wird.

In dieser einfachen Form lassen sich keine sehr hohen Frequenzen erreichen, außerdem ist die Frequenz nicht einstellbar. Eine Frequenzvariation (hin zu niedrigeren Frequenzen) kann durch eine zwischen Zeile 200 und 300 eingefügte Warteschleife erreicht werden. Will man höhere Frequenzen erzeugen, so muß man zu Maschinenprogrammen greifen.

8.6.2 Computersteuerung von elektronisch stabilisierten Netzgeräten

Elektronisch stabilisierte Netzgeräte mit variabler Ausgangsspannung lassen sich unter Einsatz der D/A-Wandlerkarte auf Computersteuerung umrüsten. Dazu muß die Referenzspannung des Netzgeräts vom Ausgang des D/A-Wandlers geliefert werden.

Abb.8.10 zeigt diese Anwendung am Beispiel eines einfachen Netzgeräts mit Längstransistor. Die in dieser Schaltung sonst üblicherweise von einer Zenerdiode gelieferte Referenz-

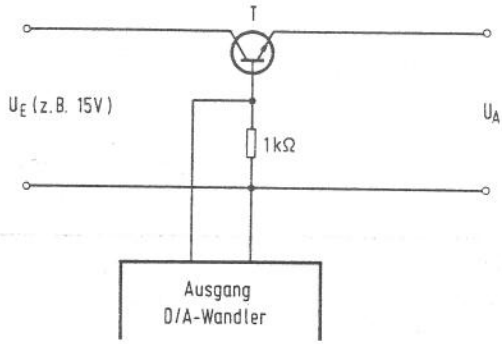


Abb. 8.10 Steuerung der Ausgangsspannung eines Netzgeräts durch den ZX 81

spannung wird durch die Ausgangsspannung des Wandlers ersetzt. Das Programmieren der Ausgangsspannung ist nicht weiter schwer, mit

POKE Adresse, N

wird der gewünschte Wert eingestellt. Da die Wandlerkarte aus Kapitel 4.3 für eine symmetrische Ausgangsspannung ($-5V \ldots +5V$) ausgelegt ist, kann N nur Werte zwischen 128 und 255 einnehmen. Zu beachten ist, daß die Ausgangsspannung um den Wert der Basis-Emitter-Spannung (rund 0,7 V) kleiner ist als die Ausgangsspannung des D/A-Wandlers. Um höhere Ausgangsspannungen zu erzeugen, kann auf der D/A-Karte der Verstärkungsfaktor des Impedanzwandlers verändert werden ($V = 1 + (R3 + R4)/(R5 + R6)$, vergl. Abb.4.8).

8.6.3 Ausmessen von elektronischen Bauteilen

Die Daten von elektronischen Bauelementen, z. B. Transistoren und Dioden können einschlägigen Tabellenwerken entnommen werden. Allerdings besitzen die Bauteile gewisse Toleranzen. Für manche Anwendungen, etwa in symmetrisch aufgebauten Verstärkerstufen, benötigt man Bauteile mit weitgehend übereinstimmenden Daten. Hierzu muß man eine Reihe von Bauteilen ausmessen und diejenigen Exemplare aussuchen, deren Daten am besten übereinstimmen. Dabei kann der Computer zusammen mit der A/D- und der D/A-Wandlerkarte nutzbringend eingesetzt werden.

Abb.8.11 erläutert den Einsatz bei der Aufnahme einer Diodenkennlinie, während in *Abb.8.12* die Kennlinie eines Transistors ausgemessen wird.

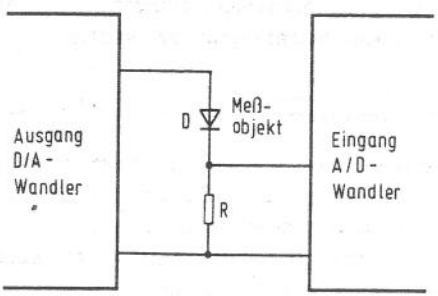


Abb. 8.11 Messen der U_F/I_F-Kennlinie einer Diode

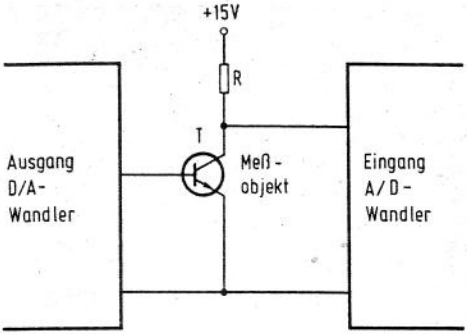


Abb. 8.12 Messen der U_{BE}/I_C-Kennlinie eines Transistors

Das Vorgehen ist in beiden Fällen gleich. Über den D/A-Wandler wird vom Rechner eine Spannung ausgegeben, die unter Programmsteuerung variiert wird. Zu jeder eingestellten Spannung mißt man den sich ergebenden Strom. Die Strommessung erfolgt dabei als Spannungsmessung über einen Arbeitswiderstand nach $I = U/R$. Dabei sollte R einen möglichst geringen Wert besitzen, damit der an ihm auftretende Spannungsabfall das Meßergebnis nicht verfälscht. Für einfache Anwendungen genügt das folgende Programm:

```
100 DIM A (128)
200 FOR N = 0 TO 128
300 POKE Adresse 1, (N+127)
400 POKE Adresse 2,0
500 LET A(N) = PEEK Adresse 3
600 NEXT N
```

Unter Adresse 1 erscheint die D/A-Karte, während Adresse 2 zum Starten der A/D-Wandler dient. Mit Adresse 3 wird der gewandelte Wert eingelesen. Nach Lauf des Programms enthält das Array A(N) die gemessenen Wertepaare. Sie können als Referenz abgespeichert werden und mit den Werten anderer ausgemessener Exemplare verglichen werden. Bei der Messung der Transistorkennlinie muß noch umgerechnet werden, da der Spannungsabfall über dem Widerstand die Differenz der gemessenen Spannung zur Versorgungsspannung (15V) darstellt. Ein direkter Anschluß des A/D-Wandlers an den Widerstand R würde einen Kurzschluß verursachen, da die einzelnen Wandler-Kanäle nicht erdfrei sind, denn sie besitzen eine gemeinsame Masseleitung.

8.7 Programm zur Dekodierung von DCF 77

Die DCF-77-Karte speichert die während einer Minute übertragenen Daten in einem Schieberegister. Um diese Informationen auszulesen und auszuwerten ist ein Maschinenprogramm nötig. Das hier vorgestellte Programm schreibt darüber hinaus das so ermittelte Datum in die rechte obere Ecke des Bildschirms ein *(Abb. 8.13)*.

Das Maschinenprogramm ist in ein REM-Statement eingebunden. Startadresse ist wie üblich 16514, die Länge beträgt 420 Byte (einschließlich Tagestabellen und Zwischenspeicher für die ausgelesenen Daten). Soll das Programm in einem anderen Speicherbereich laufen, so müssen einige absolute Adressen angepaßt werden, sonst stürzt das Programm ab.

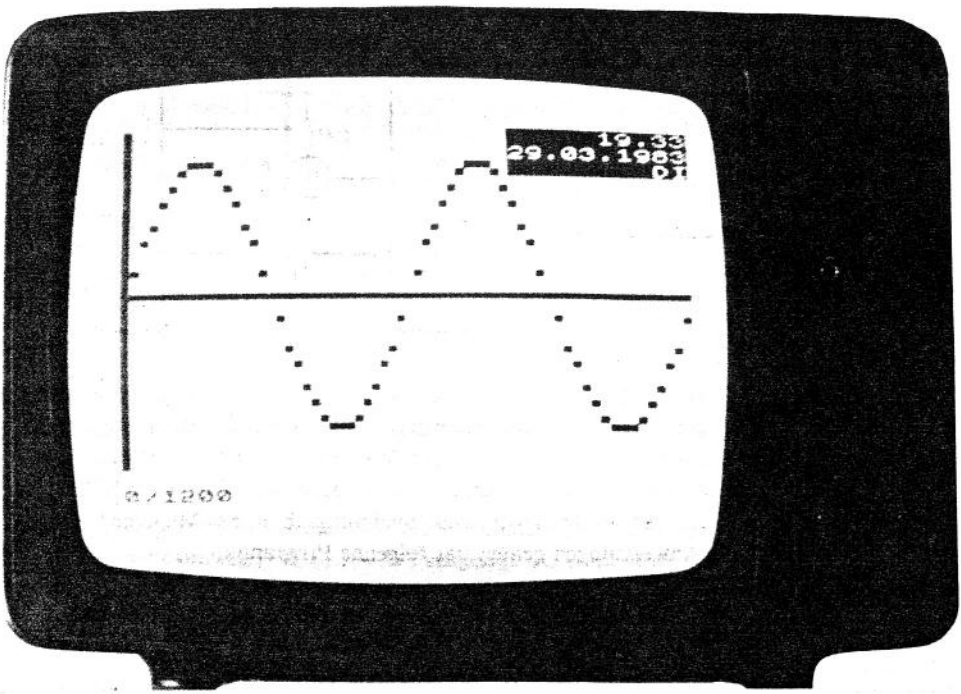


Abb. 8.13 Schirmbild (Sinusfunktion) mit Datum- und Zeitanzeige (DCF 77)

```
v 100 REM ▗": +VAL ▗": ; VAL ▗": 1VAL
 ▗": 5VAL ▗": 8VAL ▗": CVAL ▗": EVAL
 ▗": HVAL ▗": LVAL ▗": MVAL ▗": QVAL
           5"?) U  ?CHR$ +CLAT ?Y
""aACS ?4 CLS J NEW ▗m?1?<4 GOS
UB ?CHR$ "4" 1?CHR$ "4" 1Y"?,PI? "
- /INT    TAN  E£RND,")* ;
 Y?7( UNPLOT   STR$ 20"PI"?7"
"?7"Y.?7"?7"?7SGN  ;STR$ "P
I20"?7"?7Y.?7",PI"?7"?7Y.?
7Y1?7Y9?7",PI"?7"?SGN 7  ;Y
?7( UNPLOT "PI" *PI" ? FOR
FOR ,?7<,?TAN E£RND,")* ;Y ?<7?C
HR$ 54 PLOT  ( LOAD TAN
1111111111111111111
1111111112222222222222222222
2222222223333333333??"?"?"?
3333333333modimidofrsaso
 200 REM
 300 REM UHR DRUCKEN:  16514
 400 REM UHR LOESCHEN: 16784
 500 REM
```

Prog. 20

In das REM-Statement wird das Maschinenprogramm eingegeben

```
16514 ---> 6
16515 ---> 11
16516 ---> 14
16517 ---> 21
16518 ---> 197
16519 ---> 6
16520 ---> 10
16521 ---> 14
16522 ---> 25
16523 ---> 197
16524 ---> 6
16525 ---> 9
16526 ---> 14
16527 ---> 29
16528 ---> 197
16529 ---> 6
16530 ---> 8
16531 ---> 14
16532 ---> 33
16533 ---> 197
16534 ---> 6
16535 ---> 7
16536 ---> 14
16537 ---> 36
16538 ---> 197
16539 ---> 6
16540 ---> 6
16541 ---> 14
16542 ---> 40
16543 ---> 197
16544 ---> 6
16545 ---> 5
16546 ---> 14
16547 ---> 42
16548 ---> 197
16549 ---> 6
16550 ---> 4
16551 ---> 14
16552 ---> 45
16553 ---> 197
16554 ---> 6
16555 ---> 3
16556 ---> 14
16557 ---> 49
16558 ---> 197
16559 ---> 6
16560 ---> 2
16561 ---> 14
16562 ---> 50
16563 ---> 197
16564 ---> 6
16565 ---> 1
16566 ---> 14
16567 ---> 54
16568 ---> 197
16569 ---> 0
16570 ---> 0
16571 ---> 0
16572 ---> 0
16573 ---> 0
16574 ---> 0
16575 ---> 0
16576 ---> 0
16577 ---> 0
16578 ---> 0
16579 ---> 33
16580 ---> 3
16581 ---> 92
16582 ---> 17
16583 ---> 58
16584 ---> 0
16585 ---> 0
16586 ---> 123
16587 ---> 214
16588 ---> 21
16589 ---> 40
16590 ---> 49
16591 ---> 193
16592 ---> 115
16593 ---> 62
16594 ---> 192
16595 ---> 166
16596 ---> 203
16597 ---> 119
16598 ---> 32
16599 ---> 251
16600 ---> 47
16601 ---> 230
16602 ---> 128
16603 ---> 178
16604 ---> 7
16605 ---> 87
16606 ---> 29
16607 ---> 121
16608 ---> 147
16609 ---> 32
16610 ---> 237
16611 ---> 120
16612 ---> 214
16613 ---> 7
16614 ---> 32
16615 ---> 1
16616 ---> 29
16617 ---> 120
16618 ---> 214
16619 ---> 9
16620 ---> 32
16621 ---> 1
16622 ---> 29
16623 ---> 62
16624 ---> 3
16625 ---> 128
16626 ---> 79
16627 ---> 6
16628 ---> 66
16629 ---> 122
16630 ---> 2
16631 ---> 22
16632 ---> 0
16633 ---> 24
16634 ---> 207
16635 ---> 0
16636 ---> 0
16637 ---> 0
16638 ---> 0
16639 ---> 0
16640 ---> 42
16641 ---> 12
16642 ---> 64
16643 ---> 6
16644 ---> 5
16645 ---> 17
16646 ---> 23
16647 ---> 0
16648 ---> 25
16649 ---> 0
16650 ---> 0
16651 ---> 0
16652 ---> 0
16653 ---> 0
16654 ---> 0
16655 ---> 0
16656 ---> 62
16657 ---> 128
16658 ---> 119
16659 ---> 35
16660 ---> 16
16661 ---> 252
16662 ---> 0
16663 ---> 0
16664 ---> 213
16665 ---> 30
16666 ---> 156
16667 ---> 1
16668 ---> 11
16669 ---> 66
16670 ---> 10
16671 ---> 131
16672 ---> 119
```

Hardware-Erweiterungen für den ZX 81

Der ZX 81 ist mehr als ein Programmier-Übungsgerät. Wer das noch nicht gewußt hat, erkennt das sofort, wenn er das Buch aufschlägt; wer das geahnt hat, erkennt sofort die Fülle der Möglichkeiten, den ZX 81 für neue, vielseitige Aufgaben zurechtzutrimmen.
Vorgestellt werden verschiedene Hardware-Zusätze. Mit diesen Baugruppen läßt sich der Computer für Meß-, Steuer- und Regelaufgaben sowie zur Tonerzeugung einsetzen. Eine Centronier-Schnittstelle zum Anschluß eines handelsüblichen Druckers sowie eine "language card", mit der sich Änderungen im Betriebssystem durchführen lassen, werden ebenfalls beschrieben. In allen Fällen sind detaillierte Bauanleitungen mit Platinenvorlagen angegeben. So wird gleichzeitig die Funktion der einzelnen Schaltungen erklärt. Das ist ein wesentlicher Gesichtspunkt, denn er legt die Verknüpfung von Hard- und Software-Aspekten offen zu Tage.
Der kleine ZX 81, nach diesem Buch sinnvoll aufgerüstet, leistet plötzlich fast das Gleiche, wie ein wesentlich kostspieligerer PC-Computer.