

Anhang 1 *fehlt!!!*

**Mein Homecomputer
selbstgebaut**

5.4.88 *fe*

34.-

V185
2/1935

Joseph Glagla/Dieter Feiler

Mein Homecomputer

selbstgebaut

zum Lernen, Spielen, Messen, Steuern, Regeln . . .

Otto Maier Verlag Ravensburg

(2984)

Wichtiger Hinweis: Alle Schaltungen und Verfahren werden ohne Rücksicht auf die Patentlage mitgeteilt. Sie sind ausschließlich für Lern- und Experimentierzwecke bestimmt und dürfen gewerblich nicht genutzt werden. Alle Angaben wurden sorgfältig überprüft, trotzdem sind Fehler nicht auszuschließen. Verfasser und Verlag weisen ausdrücklich darauf hin, daß sie keine Garantie oder juristische Verantwortung bzw. Haftung übernehmen für Folgen, die sich aus etwaigen Irrtümern ergeben.

Ma 1460/93



Originalausgabe
 © 1984 by Otto Maier Verlag Ravensburg
 Umschlaggestaltung: Graphisches Atelier
 Otto Maier Verlag unter Verwendung eines
 Autorenfotos
 Satz: acomp, Wemding
 Druck und Verarbeitung: Mohndruck
 Graphische Betriebe GmbH, Gütersloh
 Printed in Germany

88 87 86 85 84 5 4 3 2 1

ISBN 3-473-44005-1

Inhalt

Vorwort	9	Eingangs- und Ausgangslastfaktoren (fan in, fan out)	64
Kapitel 1			
Handwerkliche Grundlagen	15	Eingangsspannungen	65
Kapitel 2		Kapitel 3	
Grundlagen der Digitaltechnik	40	Die Baustufen unseres Mikrocomputers	66
Analogtechnik – Digitaltechnik	40		
Logische Verknüpfungen	41	Kapitel 4	
Die UND-Verknüpfung (Konjunktion)	43	Das Netzteil (Baugruppe 1)	69
Die ODER-Verknüpfung (Disjunktion)	45	Schaltungstechnik	69
Die Negation (NICHT, NOT)	47	Hinweise zum Aufbau des Netzteiles	75
Gatterkombinationen	48	Inbetriebnahme und Prüfung des Netzteiles	79
Das NAND-Gatter	48		
Das NOR-Gatter	49	Kapitel 5	
Inverter aus NAND- oder NOR-Gattern	50	Die Busplatte (Baugruppe 2)	81
UND aus NAND, ODER aus NOR	50	Schaltungstechnik	81
NAND und NOR als Universalgatter	51	Hinweise zum Aufbau	83
Das exklusive ODER (EXOR, Antivalenzschaltung)	52	Die Prüfung der Busplatte	88
Der Schmitt-Trigger	53	Ersparnismöglichkeit	89
Das RS-Flip-Flop	54	Ein nützliches Hilfsmittel – der Busadapter	90
Das getaktete Flip-Flop	56		
Das Register	59	Kapitel 6	
Wichtige technische Eigenschaften der TTL-Familien		Die Dateneingabe und -anzeige (Baugruppe 3)	93
74XX und 74LSXX	61	Das Dualsystem	93
Die Betriebsspannung	61	Die hexadezimale (sedezimale) Schreibweise	95
Das Zusammenschalten von Bausteinen	61	Die Unterscheidung der Zahlensysteme	96
Der Open-Collector-Ausgang	62	Die Rechenregeln	97
Tri-State-Logik	62	Die Addition	97
		Die Subtraktion	97

Die Multiplikation	99	Prüfen des Flip-Flops FF2 für die SCHRITT-Schaltung	138
Die Division	99	Prüfen des Oszillators	138
Schaltungstechnik	100	Prüfen von $\overline{\text{INTREQ}}$ und RESET	139
Die Erzeugung der Nullen und Einsen	100	Prüfen der Datenleitungen	139
Die Anzeige der Nullen und Einsen	100	Prüfen der Adreßleitungen	139
Hinweise zum Aufbau	105	Funktionsprüfung mit dem Prozessor	139
Prüfung des Dateneingabe- und -anzeigebausteins	109	Erste Programmierübungen mit dem Computer	140
Übung von Eingaben	110		

Kapitel 7

Die CPU-Platte (Baugruppe 4)	111	Kapitel 8	
Die Struktur des 2650 A	111	Die Adreßeingabe und -anzeige (Baugruppe 5)	141
Das Rechenwerk (ALU)	111		
Das Programm-Status-Wort (PSW)	114	Kapitel 9	
Die Arbeitsregister R0, R1, R2, R3 und R1', R2', R3'	117	Der Speicher (Baugruppe 6)	148
Das Befehlsadreßregister (IAR)	118	Die Organisation der Speicheradressen	148
Der Stack	119	Die Zusammenschaltung von Speicherbausteinen	150
Das Operandenadreßregister (OAR)	119	Speichertypen	154
Die Ablaufsteuerung	120	Die Struktur von Speicherbausteinen	155
Die Anschlüsse des 2650	120	Festwertspeicher	156
Die Steuerleitungen	121	Schreib-/Lesespeicher (RAM)	159
Schaltungstechnik	125	Das EAROM	160
Der CLOCK-Oszillator	125	Die Schaltung der Speicherplatte	161
Die Decodierung der Steuersignale ADMEM und ADPER	127	Hinweise zum Aufbau	164
Die Erzeugung von $\overline{\text{WRITE}}$	128	Prüfen des Speichers	168
Das RESET-Signal	128	Die Arbeit mit dem Speicher	170
Das PAUSE-Signal	129	Den Speicher laden	170
Das $\overline{\text{INTREQ}}$ -Signal	129	Den Speicherinhalt lesen	171
Das OPACK-Signal	129	Ein Programm abarbeiten lassen	171
Die Einzelschrittsteuerung	129		
Die Signale ADREN und DBUSEN	132	Kapitel 10	
Hinweise zum Aufbau	133	Die Porteinheit (Baugruppe 7)	173
Prüfung der CPU	137	Schaltungstechnik	173
Prüfen der Steuersignale	137	Ein Ausgangsport mit Auffangregister	178
Prüfen des Tastenentprell-Flip-Flops (N5, N6)	138	Ein Universaleingangsausgangsport	179
		Weitere Portadressen	181
		Hinweise zum Aufbau	182
		Prüfung und Inbetriebnahme der Porteinheit	187

Kapitel 11

Der Analog-Digital-Wandler (Baugruppe 8)	190	Die Schaltung der Tastatur	220
Schaltungstechnik	190	Aufbau der Tastatur- und Anzeigeeinheit	225
Digital-Analog-Umwandlung mit einem R-/2R-Widerstandsnetzwerk	190	Aufbau der Anzeigeplatte	225
Analog-Digital-Wandlung mit einem R-/2R-Widerstandsnetzwerk, einem Dualzähler und einem Spannungskomparator	193	Aufbau der Grundplatine	227
Der CLOCK-Oszillator für den Zähler	196	Die Bedienung der Tastatur	234
Die Erzeugung des Umwandlungsimpulses $\overline{\text{CC}}$ und des Speicherübernahme-Impulses E	197	Beispiel für die Eingabe eines Programms	237
Die Portschaltung für den AD-Wandler	199		
Der Vorverstärker	200	Kapitel 13	
Die Gesamtschaltung des AD-Wandlers	203	Das Kassetteninterface	240
Hinweise zum Aufbau	205	Die Arbeitsweise des Kassetteninterfaces	240
Inbetriebnahme und Abgleich des AD-Wandlers	209	Aufbau und Inbetriebnahme des Kassetteninterfaces	243
Vorbereitungen	209		
Die benötigten Bausteine	210	Kapitel 14	
Der Offsetabgleich von IC 7	210	Programmbeispiele	250
Der Offsetabgleich von IC 6	210	Allgemeine Hinweise zur Programmierung und Anwendung der Programme	250
Die Einstellung des Grundmeßbereichs	210	Liste der nutzlosen Monitorroutinen	253
Der Abgleich der übrigen Meßbereiche	212	Vom Monitor benutzte Speicherstellen	254
Die Anwendung des AD-Wandlers	212	440-Hz-Programm zum Abgleich des CLOCK-Oszillators	255
		Portadressierung	256
		Einfache Addition	256
		Einfache Subtraktion	256
		Einfache Multiplikation	257
		Einfache Division	257
		Verkehrssampel 1	258
		Blinklicht	258
		Würfel	259
		Metronom	260
		Lauflicht	260
		Voltmeter mit dualer Anzeige	261
		Denkzeitbegrenzer	261
		Verkehrssampel 2	262
		VU-Meter	263
		Laufschrift	264
		Würfelspiel	265
		Reaktionstest	266
		Stoppuhr	268
		Digitaluhr	270
		HEX-DEZ-HEX-Wandlung	274

HEX-Rechnen	276
DEZ-Rechnen	277
Lottozahlen	279
Morse-Übungsprogramm	281
Voltmeter 2 (3stellig dezimal)	283
Anhang 1	
Platinenvorlagen	285
Anhang 2	
Detaillierte Beschreibung des Befehlssatzes des Mikroprozessors 2650	307
Literatur	357
Sachverzeichnis	358

Vorwort

Lieber Computerfreund, dieses Buch soll Ihnen Einsichten in die Arbeitsweise eines Computers vermitteln und Ihnen nebenbei auch noch zu einem eigenen Homecomputer verhelfen.

Kleincomputer für den privaten Bereich bietet der Markt – gemessen an ihrer Leistungsfähigkeit – schon für wenig Geld. Mit ihnen kann man sich gut in die **Software**, das Bedienungs- und Programmierwissen, einarbeiten. Das ist nicht wenig. Ein Computer, egal ob klein oder groß, ist nämlich nur so viel wert wie die Programmierfähigkeit seines Benutzers. Über das materielle Innenleben, die **Hardware**, also alles, was man anfassen kann, ist in den Bedienungsanleitungen nicht viel zu erfahren.

Trotz der Beschränkung auf die Software kann man mit einem Computer sinnvoll umgehen. So geht es uns doch in vielen Dingen: Wer z. B. eine Fahrschule besucht und anschließend die Führerscheinprüfung besteht, kann sein Fahrzeug sinnvoll nutzen, auch wenn er in seinem Auto weder die Benzinpumpe noch den Vergaser finden würde. Schließlich bildet der Fahrlehrer seine Schüler ja nicht zu Automechanikern aus, sondern führt sie an das Bedienwissen heran. Irgendwann kommen aber doch die Fragen, warum und wieso das alles funktioniert, beim Auto wie beim Computer. Und je mehr man von der

Hardware, also den technischen Hilfsmitteln, versteht, desto besser kann man mit dem Computer umgehen.

Der sicherste Weg, hinter die eigentlich gar nicht so geheimnisvollen Geheimnisse der Technik zu kommen, ist, sie im ursprünglichen Sinne des Wortes zu „begreifen“. Wenn Sie die folgende Bauanleitung ins Werk setzen, besitzen Sie am Ende nicht nur einen leistungsfähigen Homecomputer, der Ihnen allerlei Arbeiten abnehmen kann, sondern Sie kennen sich auch mit ihm aus. Sie lernen nicht nur die Software, sondern auch die Hardware kennen, und zwar von Grund auf:

Allein vermag ein Mikroprozessor nichts. Umgeben Sie ihn jedoch mit einigen Schaltern und Leuchtdioden, so können Sie ihn zu den ersten Arbeiten veranlassen. Sie werden seine „Zutaten“ erweitern; er wird zu einem Mikrocomputer heranwachsen und immer mehr Aufgaben übernehmen, und zugleich wird Ihr Wissen über die typischen Funktionen eines Mikrocomputers mitwachsen. Damit diese überschaubar bleiben, werden sämtliche Funktionseinheiten für sich auf eigenen Leiterplatten aufgebaut. Und so wie ein Baum immer ein Ganzes ist und er doch Jahr um Jahr weiterwachsen kann, so ist auch Ihr Computer auf jeder Stufe ein Ganzes, das immer weiter ausgebaut werden kann.

Zwar unterscheiden sich die Mikroprozessoren der verschiedenen Hersteller teilweise deutlich voneinander, doch das Prinzip, nach dem sie arbeiten, ist allen gemeinsam. Sie werden Ihr Wissen, das Sie sich beim Nachbau des Geräts erworben haben, leicht auf andere Mikroprozessoren übertragen können.

Wenn Sie sich schon die Mühe machen, Ihren Computer selbst zu bauen, so wollen Sie selbstverständlich den besten Mikroprozessor einsetzen. Nur leider gibt es „den besten“ nicht – jedenfalls nicht allgemein. Jeder Mikroprozessor hat seine besonderen Vorzüge und Eigenheiten. Bei unserem Computer, der ja nicht zuletzt auch ein Lerncomputer sein soll, kommt es darauf an, daß man ihn „echt“, also ohne programmier-technische Kunstgriffe, beliebig langsam, ja nahezu zeitunabhängig arbeiten lassen kann, denn nur dann ist man imstande, sich in Ruhe den Datenfluß und das Zusammenspiel der Steuerungssignale, den Ablauf eines Maschinenzyklus usw. anzusehen.

Für diesen Zweck kommt nur ein „vollstatischer“ (siehe Seite 159) Mikroprozessor in Frage. Warum?

Die Funktionsabläufe in Mikroprozessoren werden von einem Taktgenerator koordiniert. Bei einem vollstatischen Mikroprozessor ist es nicht wichtig, ob zwischen zwei Taktimpulsen die kürzestmögliche Zeit von einigen Nanosekunden (Milliardstel Sekunden) vergeht oder ob es Minuten, Stunden oder Tage sind, wenn nur die Betriebsspannung zwischendurch nicht abgeschaltet wird. Man hat daher im Bedarfsfall beliebig viel Zeit, sich mit den Vorgängen von einem Taktimpuls bis zum nächsten zu beschäftigen.

„Dynamische“ (siehe Seite 159) Mi-

kroprozessoren benötigen in sehr kurzen Abständen eine Auffrischung ihres Gedächtnisses. Die längste Zeit, die von einem Auffrischungsimpuls bis zum nächsten vergehen darf, liegt durchschnittlich in der Größenordnung von 2 Mikrosekunden (2 Millionstel Sekunden). Werden die internen Speicher eines dynamischen Mikroprozessors innerhalb dieser Zeit nicht aufgefrischt, gehen sie verloren. Die Auffrischung ist an den Taktimpuls gekoppelt. Man hat also zum Beobachten der Vorgänge von einem Taktimpuls bis zum nächsten höchstens 2 μ s Zeit, und das ist sogar für eine Fliege zu wenig, obwohl sie doch viel schneller sehen kann als ein Mensch. Dem Bedürfnis nach „Beschaulichkeit“ können daher dynamische Mikroprozessoren nur bedingt entgegenkommen.

Freilich lassen sie sich programmier-technisch quasi anhalten, aber auch dann zeigen sie nicht alles. Wer es genau wissen will, benötigt zur Beobachtung der Steuerungssignale einen beachtlichen Meßgeräteaufwand, z. B. ein (Mehrkanal-)Oszilloskop, und dieses nur, um sichtbar zu machen, was bei einem vollstatischen Mikroprozessor mit einigen Leuchtdioden bequem zu beobachten ist.

Der dargestellte Unterschied zwischen den Mikroprozessoren bezieht sich nur auf das Zeitverhalten ihrer internen Speicher und hat im übrigen mit der Arbeitsweise der Mikroprozessoren nichts zu tun. Im Normalbetrieb sollen Mikroprozessoren sowieso schnell laufen, je schneller, desto besser.

Zum Lernen empfiehlt sich daher ein vollstatischer Mikroprozessor – aber welcher? Beim Durchsuchen der Datenbücher merkt man bald, daß die Auswahl minimal ist, eigentlich gar

nicht mehr gegeben ist, wenn man außerdem auch noch einen möglichst vielseitig einsetzbaren Mikroprozessor verlangt. Die meisten Wünsche erfüllt der 2650 A von VALVO/Signetics. Er ist kein Frischling mehr, eher fällt einem das Wort „Großvatergeneration“ ein. Er kam 1976 auf den Markt. Doch sein Alter spricht nicht gegen seine Vielseitigkeit. Typische Anwendungsgebiete sind industrielle Steuerungen, Meßgeräte, Haushaltsgeräte, Waagen, Registrierkassen, Fernsehspiele, mittlere Datentechnik, Textautomaten, Kommunikationstechnik usw.

Fortschrittsehrgeizige brauchen sich nicht schrecken zu lassen, denn die Vorzüge des 2650 gelten auch heute noch: Er besitzt einen klar strukturierten, leistungsfähigen Befehlssatz, der alle Grundbefehle enthält, die so oder sehr ähnlich bei den anderen Mikroprozessoren ebenfalls vorkommen. Die Arbeit mit ihm führt zu einem Basiswissen, an dem der Zahn der Zeit nicht so schnell nagen kann. Was ihm als Nachteil angekreidet werden kann, ist sein vergleichsweise langsames Arbeitstempo – aber gerade das zeichnet ihn ja für unseren Zweck aus.

Außerdem ist er einfach zu beschalten und überschüttet Sie nicht mit Problemen, die den Lernenden mindestens zu Beginn nur behindern können. Dazu ein Beispiel:

Mikroprozessoren werden vielfach in 40poligen Gehäusen untergebracht. Oft reichen die 40 Anschlüsse nicht aus. Wie hilft sich nun der gewiefte Elektroniker? Er benutzt viele Anschlüsse doppelt, abwechselnd für den einen oder anderen Zweck. Und wenn sich der Wechsel dann auch noch „dynamisch“ schnell vollzieht, versteht nur noch derjenige die Vor-

gänge, der sie eigentlich schon kennt. Beim 2650 haben sich die Entwicklungsingenieure zurückgehalten: sie haben nämlich nur zwei Anschlüsse doppelt belegt – und das auch nur für zwei Fälle, die man durch Befehle bewußt herbeiführt. Ansonsten sind alle Anschlüsse für jeweils eine einzige Funktion vorgesehen.

Außerdem ist der 2650 ein sehr „gutmütiger“ Mikroprozessor; er ist relativ unempfindlich gegen Störimpulse und Zeitprobleme. Wie bereits gesagt, werden die Funktionsabläufe durch den Taktgenerator koordiniert. Bei den sehr kurzen Taktzeiten kann es zu sogenannten *Timing-Problemen* (time, engl. Zeit) kommen. Diese können dem Selbstbauer erhebliches Kopfzerbrechen bereiten: Ungeschickt verlegte oder zu lange Drähte können u. U. einen Mikroprozessor in seiner Funktion erheblich beeinträchtigen. In dieser Hinsicht werden Sie nichts Übles zu befürchten haben, auch dann nicht, wenn Ihnen einmal ein paar Drähte einige cm zu lang geraten sind.

Und damit sind wir bei der Nachbausicherheit. Die Tatsache, daß der beschriebene Mikrocomputer von Schülern der 7. und 8. Jahrgangsklasse gebaut wurde, die erst über sehr geringe Erfahrungen bei elektronischen Bastelarbeiten verfügten, darf Sie ermutigen. Ihre Zuversicht, daß Sie das auch können, ist daher wohl berechtigt.

Es ist freilich schon ein gutes Stück Arbeit, einen Mikrocomputer selbst zu bauen; das soll hier nicht verschwiegen werden. Im Vergleich zu anderen Vorhaben ist diese Arbeit jedoch leicht. Nur an Geduld und an einer gewissen Pingeligkeit darf es nicht fehlen. Dafür werden keine eigentlich elektronischen Probleme

auf Sie zukommen, die Sie nur mit einem breiten Grundlagenwissen beherrschen können. Allerdings werden Sie auch nicht ohne jegliche Grundkenntnisse auskommen: Die Begriffe „Spannung“, „Strom“ und „Widerstand“, kurz: das Ohmsche Gesetz sollte Ihnen vertraut sein; auch die Grundfunktionen der Bauelemente Widerstand, Kondensator, Diode, Leuchtdiode, Transistor sowie des Schalters sollten Sie kennen. Da sich aber nahezu alle Funktionen in integrierten Schaltkreisen (IC von Integrated Circuit) abspielen, können Sie sich mit Recht darauf verlassen, daß der Hersteller die elektronischen Probleme für Sie gelöst hat. Die notwendigen Kenntnisse über die verwendeten ICs können Sie sich beim Bau Ihres Computers nebenher erarbeiten.

Außerdem gibt es kaum mechanische Arbeiten, die von Ihnen die Geschicklichkeit eines Feinmechanikers erfordern. Daher ist es nicht übertrieben zu behaupten, daß der Bau eines guten Radios viel schwieriger ist. Anforderungen an Ihre handwerklichen Fähigkeiten, wie sie z. B. bei der Herstellung eines Skalenantriebs für ein Rundfunkgerät auftreten, gibt es bei dem beschriebenen Computer nicht. Auch mit der ersten Inbetriebnahme werden Sie es wesentlich leichter haben als mit dem Abgleich eines Radios, denn der erfordert nicht nur einen gewissen Meßgeräteaufwand, sondern auch Erfahrung. Das liegt daran, daß es in der Computerei im wesentlichen nur zwei Spannungspiegel gibt: Entweder ist an einem Punkt die Spannung „hoch“ (H, von engl. High), also nahe der Betriebsspannung, oder sie ist niedrig (L, von engl. Low), nahe dem Bezugspotential („Masse“), die vielen möglichen Zwi-

schwerte treten gar nicht auf. In einem Rundfunkgerät dagegen sind gerade sie interessant, aber auch nicht leicht zu beherrschen.

Zum Nachbau werden Ihnen zwei Wege vorgeschlagen: der Aufbau auf Platinen („gedruckten Schaltungen“) oder das Verdrahten auf Experimentierplatten.

Die Platinen bieten eine höhere Nachbausicherheit, weil durch die vorgegebenen Leiterbahnzüge die Möglichkeiten, Fehler einzubauen, weitgehend ausgeschlossen sind. Außerdem ist der Arbeitsaufwand vergleichsweise gering. Dafür sind aber die Möglichkeiten, die Einzelfunktionen durch Zwischentests kennenzulernen, etwas eingeengt.

Der Aufbau auf Experimentierplatten erfordert deutlich mehr Arbeit; auch sind Verdrahtungsirrtümer nicht auszuschließen. Trotzdem lohnt sich gerade der Aufbau: Jede Einzelfunktion ist in einem Zwischentest sofort zu überprüfen.

Die verwendeten Bauelemente dürfen, mit Ausnahme des Mikroprozessors MAB 2650 A und der besonders hellen Siebensegmentanzeigen, in jedem Elektronikladen zu bekommen sein. Die *Firma Helga Feiler, Hohenloher Ring 5, 2081 Bönningstedt, Tel. 040/5 56 61 10*, hat sich freundlicherweise bereit erklärt, alle speziellen Teile auch in Einzelstücken im Versand zu liefern. Dort erhalten Sie auch das programmierte EPROM, das Sie zur Inbetriebnahme der Tastatur und der Anzeige benötigen und das Ihnen mit vielen, immer wiederkehrenden Unterprogrammen Ihre spätere Programmierpraxis erleichtert. Daher dürften auch materiell dem Nachbau keine unüberwindlichen Schwierigkeiten im Wege stehen.

Im Anschluß an die Erläuterung der Hardware und der Bauanleitung enthält das Buch eine Anzahl von Programmbeispielen, mit denen Sie gleich arbeiten können. Programmieren lernt man nicht zuletzt durch das Analysieren und systematische Verändern vorhandener Programme. Daher sind in den Programmbeispielen viele Stellen markiert, an denen Sie beginnen können, ein Programm zu ändern und Ihren Vorstellungen anzupassen.

Die lange Befehlsliste (Anhang) soll Sie nicht erschrecken. Sie brauchen nicht gleich alle Befehle auswendig zu lernen. Suchen Sie sich für den Anfang einige wenige Befehle heraus, mit denen Sie immer wieder experimentieren. Auch ein professioneller Programmierer hat seine „Lieblingsbefehle“ und nutzt durchaus nicht immer den gesamten Befehlsvorrat.

Einen kompletten Programmierlehrgang kann Ihnen dieses Buch wegen des Raumbedarfs noch nicht bieten; er ist jedoch in Vorbereitung und wird einen eigenen Band füllen.

Mit der im Anhang 2 ausführlich dokumentierten Befehlsliste dürfen Sie hemmungslos experimentieren. Sie tun dem Prozessor auch durch die seltsamsten Befehlsverknüpfungen nicht weh. Nur notieren Sie sich alles, was Sie tun, denn sonst kann es Ihnen passieren, daß ein Programm plötzlich läuft – nur wissen Sie nicht, warum.

Die Verfasser wünschen Ihnen viel Erfolg und noch mehr Freude bei Ihrer Computerei!

Last, not least danken sie Herrn Kind aus dem Hause VALVO/Hamburg, der dieses Projekt von Anfang an mit seinem guten Rat begleitete.

Kapitel 1

Handwerkliche Grundlagen

Der Computer ist so aufgebaut, daß sich jede Funktionseinheit auf einer eigenen Leiterplatte befindet und dadurch auch als solche sichtbar wird. Deshalb sind auf jeder Leiterplatte auch vergleichsweise wenige Bauelemente untergebracht. Es ist daher nicht unbedingt notwendig, die vorgeschlagenen, gedruckten Schaltungen zu verwenden. Wer schon einige Erfahrung im Verdrahten auf Experimentierplatten hat, kann sich den Computer auch darauf zusammelöten. Daher sind in diesem Kapitel auch Hinweise zum Aufbau auf Experimentierplatten zu finden. Wer jedoch noch wenig Erfahrung im Nachbau elektronischer Schaltungen hat, sollte sich lieber an die gedruckten Schaltungen halten, weil durch sie viele Fehlerquellen von vornherein entfallen, und so kann sich der Mehraufwand durchaus bezahlt machen.

Montage der Bauelemente

Die Anschlüsse der Bauelemente werden von der Isolierseite aus durch die Bohrungen gesteckt und auf der Kupferseite mit den Leiterbahnen verlötet (Bild 1.1). Alle Bauelemente sollen dicht auf der Leiterplatte aufliegen; das erhöht die mechanische Festigkeit; sie ist deswegen von gro-

ßer Bedeutung, weil der Computer kein Gehäuse bekommt. Die Bauelemente wie Schalter und Taster müssen unbedingt fest aufliegen, damit bei der Handhabung keine Kraft auf die dünnen Kupferbahnen übertragen wird. Das würden sie nicht lange überstehen.

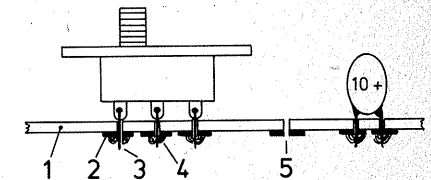


Bild 1.1 Die Anschlußdrähte der Bauelemente werden von der Isolierseite der Platte (oben) aus durchgesteckt und an der Kupferseite angelötet. Die überstehenden Drahtenden werden abgeschnitten. Die Bauelemente sollen fest aufliegen, das gilt besonders für Bedienelemente wie Schalter und Taster.

1: Isolierplatte, 2: Kupferbahn, 3: Anschlußdraht, 4: Lötzinn, 5: Bohrung

Verdrahtung

Die Drahtbrücken auf der Bauteilseite bestehen aus blankem Schalt-draht von 0,5 bis 0,6 mm \varnothing , nicht dik-

ker. Sie werden wie Bauteile behandelt.

Die Drahtverbindungen auf der Kupferseite bestehen ausnahmslos aus isolierter dünner Litze (0,14 mm²). Ihre Enden werden auf die Leiterbahnen bzw. Lötäugen gelötet und nicht in die Bohrungen gesteckt. Dadurch soll verhindert werden, daß sich die Litzen auffächern und Kurzschlüsse mit benachbarten Leiterbahnen erzeugen. Massiver Schalterdraht eignet sich für diese Verbindungen nicht, weil er zu steif ist. Die Kupferfolie der Leiterbahnen ist ja nur angeklebt, und wenn man die steifen Drähte bewegt, so kann sich über diese so viel Kraft auf die schmalen Klebestellen übertragen, daß sich die Kupferbahnen an den Lötstellen, wo der Kleber durch die Lötwärme schon geschädigt ist, von den Grundplatten ablösen.

Elegante Methoden zur Verdrahtung ergeben sich aus der Verwendung von HF-Litze oder ca. 0,2 mm starkem „Fädeldraht“. Beiden Drahtsorten ist gemeinsam, daß der Kupferdraht mit einer Lackschicht isoliert ist, die in der Lötwärme schmilzt. Bei HF-Litze kommt noch eine Umspinnung aus Kunstseide hinzu, die ebenfalls in der Lötwärme schmilzt. Daher braucht man das Drahtende nur um den Bauteilanschluß zu schlingen und kann es dann anlöten. Dieses Verfahren hat aber einen Nachteil: Alle Drähte sehen gleich aus; Verdrahtungsirrtümer darf man sich nicht leisten, denn in einem Kabelbaum aus solch feinen Drähten, der eher einem Haarbüschel gleicht als einem Bund Leitungsdrähte, sind einzelne Leitungen sehr mühsam zu verfolgen und reißen bei Versuchen, das Gewirr zu sortieren, außerdem noch leicht ab.

Am besten ist es, Sie beschaffen sich

ein Stück flexible Steuerleitung (LIYY 0,14 mm²) mit 36, 50 oder noch mehr Adern (Bild 1.2). Dann besitzen Sie viele verschiedenfarbig codierte, sehr weiche Litzen, die Sie gut verarbeiten können und bei denen Sie immer die Übersicht behalten.

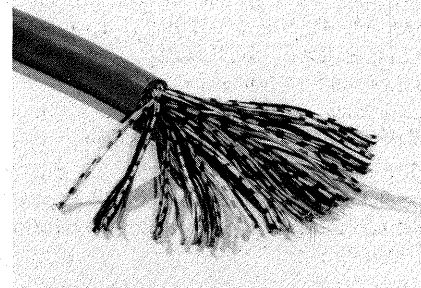


Bild 1.2 Flexible Steuerleitung mit zahlreichen, farblich gekennzeichneten Adern

Beim Platinenaufbau spielt die farbige Codierung wegen der geringen Anzahl der Leitungen nur eine untergeordnete Rolle, aber bei der Verdrahtung auf Experimentierplatten ist sie unentbehrlich.

In der Wahl der Farben sind Sie natürlich frei, nur für die Spannungsversorgung sollten Sie konsequent die übliche Codierung einhalten: rot für +U_b (VCC, +5 V), blau für „Masse“ (0 V, was dem Minuspol der Versorgungsspannung entspricht); wenn Sie den AD-Wandler bauen, benötigen Sie auch -5 V; für diese Leitung nehmen Sie Schwarz. Benutzen Sie diese drei Farben ausschließlich für diese Zwecke, denn eine Verwechslung der Versorgungsspannungen hat für die ICs fatale Folgen.

Bestimmte Leitungsgruppen, z. B. die acht Datenleitungen D0 bis D7 und die Adreßleitungen A0 bis A11 (bzw. A14), kommen immer wieder vor.

Wählen Sie für diese Leitungen einen Farbcode (die oben erwähnte Steuerleitung bietet ja genug Möglichkeiten an), und notieren Sie ihn für jede Leitung, z. B. weiß/blau für D0 (Abkürzungen siehe unten), und benutzen Sie diesen Farbcode, wo immer eine Leitung D0 auftaucht, sonst nie. Sie werden überrascht sein, wie transparent Ihnen Ihr Werk ist, auch dann, wenn es einem Nichteingeweihten als Kabelverhau erscheint.

Es ist so ziemlich alles genormt, natürlich auch die Farbcodierung der Leitungen. Benutzen Sie bei Ihren Notizen die üblichen Abkürzungen; Sie erleichtern sich die Verständigung mit anderen:

weiß:	ws	braun:	br
gelb:	ge	grau:	gr
grün:	gn	blau:	bl
rot:	rt	rosa:	rs
violett:	vi	schwarz:	sw

Jede Ader hat eine Grundfarbe (Körperfarbe), auf welche Farbmarken (bei neuen Kabeln Ringe, bei alten auch Spiralstreifen) aufgebracht sind. Die Körperfarbe wird zuerst notiert; eine weiße Ader mit blauen Ringen trägt demnach das Kürzel: wsbl.

Da die Anzahl der Farben begrenzt ist, aber manchmal 96 oder mehr Adern in einem Steuerkabel zusammengefaßt sind, werden Farbmarkierungen wiederholt. Grundsätzlich werden immer nur zwei Farben verwendet. Die Unterscheidungsmerkmale ergeben sich aus den Abständen der Farbmarkierungen (Bild 1.3): Sie können Farbmarkierungen auch mit Schrägstrichen notieren, dann sind sie leichter zu lesen, z. B. ws/sw statt wssw oder ws/sw/sw statt wssws.



Bild 1.3 Anordnung der Farbbringe auf einzelnen Adern. Nicht nur auf die Farben, sondern auch auf die Abstände der Farbbringe ist zu achten

Abisolieren

An den Litzenenden muß vom PVC-Mantel ca. 1 cm entfernt werden. Auch wenn ein normal kräftiger Dauernagel bei diesen dünnen Litzen ausreicht, ist er doch nicht das rechte Dauerwerkzeug. Auch vom Seitenschneider ist bei diesen dünnen Litzen abzuraten.

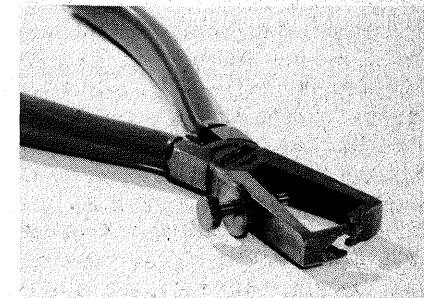


Bild 1.4 Kleine Abisolierzange

Sie benötigen nicht unbedingt eine Abisolierzange (Bild 1.4). Sie ist sehr hilfreich, aber nur dann, wenn sie mit der Stellschraube so eingestellt ist, daß sie zwar den PVC-Mantel noch packt, aber keines der haarfeinen Drähtchen. Außerdem nützt Ihnen auch die besteinstellte Zange nichts, wenn Sie nicht genau in Längsrichtung des Drahtes ziehen

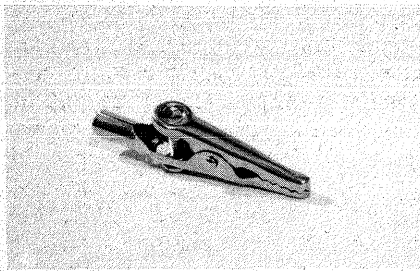


Bild 1.5 Krokodilklemme mit auch vorn gezähntem Maul

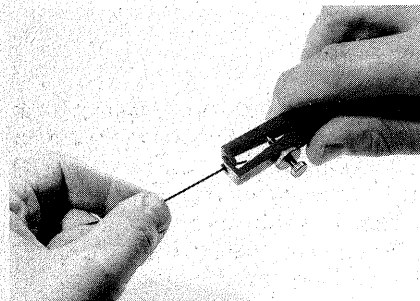
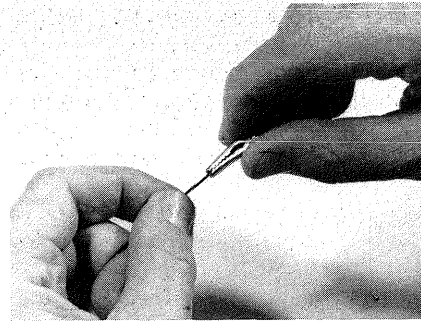


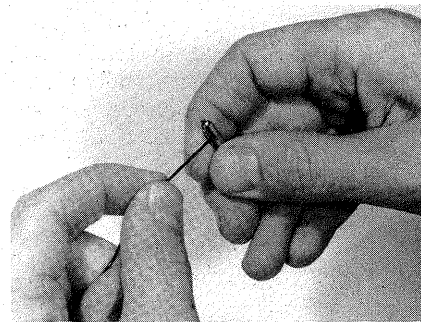
Bild 1.6 Abisolieren mit der Abisolierzange

(Bild 1.6). Verkantet die Zange, so erfaßt sie garantiert einige der feinen Litzendrähtchen und reißt sie ab.

Statt der Zange eignet sich auch eine Krokodilklemme (Bilder 1.5 und 1.7a). Bei diesen kleinen Arbeiten steht sie der vergleichsweise groben Zange nicht nach. Beim Kauf müssen Sie nur darauf achten, daß das „Maul“ der Krokodilklemme nicht nur seitlich, sondern auch vorn Zähne hat. Doch wenn Sie bloß eine Klemme mit nur seitlicher Zahnung auf-treiben können, ist es auch nicht schlimm; Sie benutzen die Klemme dann quer (Bild 1.7b). In beiden Fällen ist es wichtig, daß Sie das Krokodilmaul mit Daumen und Zeigefinger leicht zusammendrücken – und genau in Längsrichtung des Drahtes ziehen.



a)



b)

Bild 1.7 Abisolieren mit der Krokodilklemme, a) „längs“, b) „quer“

Verdrillen

Verdrillen Sie nun die freistehenden feinen Litzendrähtchen **stramm**, damit sie beim Anlöten nicht aufspringen, und verzinnen Sie sie (siehe Verzinnen).

Löten

Sie benötigen einen elektrischen **Löt-kolben** mit einer Leistung von 30 bis 50 W. Diese Leistung ist erforderlich, damit reichlich Wärme schnell genug nachgeliefert werden kann (siehe auch „Entlöten“). Feine Lötnadeln

mit Leistungen von 6 bis 15 W reichen nicht für alle Arbeiten aus.

Ihr LötKolben muß unbedingt eine schlanke, „bleistiftspitze“ Lötspitze haben, am besten ist eine Longlife-Spitze, die galvanisch dauerverzinkt ist.

Ferner brauchen Sie **Kolophoniumlöt-draht**; er besteht aus einem Zinnrohr, das mit Kolophonium oder einem verwandten Flußmittel gefüllt ist. Verwenden Sie kein zusätzliches Flußmittel wie z. B. Lötfett. Es breitet sich weiter über die Leiterplatte aus. Seine Säurebestandteile leiten und schaffen damit nicht nur undefinierte elektrische Verhältnisse, sondern sie verursachen auch die Korrosion Ihrer Lötstelle.

Der Löt-draht sollte 0,7 bis 1 mm dünn sein, nicht dicker.

Oberstes Gebot beim Löten ist **Sauberkeit der Lötstelle**, d. h. Freiheit von Fett und Oxidresten. Die LötKolben-spitze muß gut verzinkt sein, d. h. sie muß im warmen Zustand silbrig glänzen. Schon eine dünne Zunderschicht vermag die Wärme der LötKolben-spitze weitgehend zu isolieren und einwandfreies Löten zu verhindern. Wischen Sie daher die LötKolben-spitze von Zeit zu Zeit an einem feuchten Viskoseschwamm ab und schmelzen neues Lötzinn daran.

Die Leiterbahnen müssen metallisch blank sein.

Die Kupferbahnen der Experimentierplatten sind in der Regel mit Löt-lack überzogen. Er soll sie vor Oxidation schützen; er schmilzt in der Löt-wärme und dient zugleich als Flußmittel. Wenn die Experimentier-platten aber lange gelegen haben, so kann das Kupfer doch oxidiert sein. Sie erkennen es daran, daß die Leiterbahnen nicht blank durch den Lack scheinen, sondern matt und braun

aussehen. In diesem Fall wischen Sie den Löt-lack mit einem acetonge-tränkten Papiertaschentuch ab, putzen die Leiterbahnen in Längs-richtung mit einem Scheuermittel blank, spülen die Platte mit viel klarem Wasser ab, trocknen sie **gut** ab (zuerst mit einem Papiertaschentuch, das saugt gut, dann auf der Heizung oder in der Sonne) und sprühen sie **dünn** mit Löt-lack ein; die kleinste Spraydose, die Sie im Elektronikhandel bekommen können, reicht für viele Platten.

Verzinnen

Die Anschlüsse von Bauelementen sind in der Regel schon verzinkt oder so beschaffen (z. B. versilbert), daß sie sich ohne Schwierigkeiten anlöten lassen. Das gilt auch für die Draht-brücken aus versilbertem Kupfer-draht, die Sie auf der Bestückungs-seite der Leiterplatten verlegen.

Alle anderen Drähte müssen vor dem Anlöten unbedingt verzinkt werden. Dazu bräuchten Sie eigentlich eine dritte Hand, aber wenn Ihr Löt-draht auf eine Rolle gewickelt ist, kommen Sie auch mit Ihren beiden Händen aus (Bild 1.8): Sie stellen die Rolle dicht vor sich hin und lassen das Ende des Löt-drahts ein wenig vorstehen.

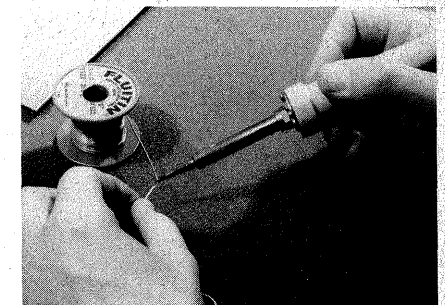


Bild 1.8 Verzinnen am freistehenden Löt-draht

Mit der linken Hand halten Sie das zu verzinnende Drahtende an das Ende des Lötdrahtes und mit der rechten Hand die LötKolbenspitze daran. Sie schmelzen möglichst wenig Zinn, gerade so viel, daß sich das Litzenende vollsaugt.

Dann kürzen Sie das verzinnte Drahtende auf 1 mm (!) Länge; dadurch erreichen Sie, daß kein Stückchen abisolierter Draht über die Lötstelle hinausragt.

Die Lötäugen oder Leiterbahnen, auf welche Sie die Litzen löten (betrifft immer nur die Kupferseite), müssen ebenfalls verzinnt werden. Sie setzen dazu den LötKolben auf die Leiterbahn (das Lötauge) und schieben ein wenig Zinn in die Fuge zwischen LötKolben und Kupfer, so als wollten Sie einen imaginären Draht anlöten. Auf der zukünftigen Lötstelle befindet sich nun ein flacher, glänzender Zinnhügel, und wenn Sie zum Verzinnen nicht zu viel Zeit benötigt haben, ist er noch mit einer hauchdünnen Kolophoniumschicht überzogen.

Lassen Sie diesen Arbeitsgang unter keinen Umständen aus! Wenn Sie nämlich die Litzen anlöten wollen, so setzen Sie das verzinnte Litzenende auf den Zinnhügel und verflüssigen das Zinn mit dem LötKolben. Das Zinn der Leiterbahn und das der Litze laufen ineinander, und Sie haben eine perfekte Lötung zustande gebracht. Sie dürfen das Litzenende sanft mit der LötKolbenspitze in das Zinn drücken, mehr nicht; auf keinen Fall dürfen Sie stark drücken oder gar reiben. Sie müssen nämlich vermeiden, daß Sie mit dem LötKolben die feinen Litzendrähte auseinanderdrücken und auffächern. Dann gibt es Kurzschlüsse, die Sie allenfalls mit einem Messer beseitigen können, und das übersteht oft die Leiterbahn nicht.

Achten Sie außerdem konsequent darauf, daß Sie die Litzen ausschließlich in Richtung der Leiterbahn anlöten, nie quer dazu (Bild 1.9), weil Sie sonst Kurzschlüsse zwischen zwei Leiterbahnen nicht verhindern können. Die PVC-Isolierung schrumpft in der Lötwärme und zieht sich auch nach dem Verzinnen noch ein Stückchen zurück.

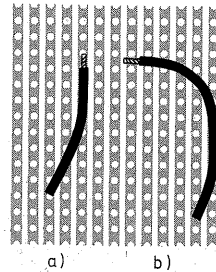


Bild 1.9 Die Gefahr eines Kurzschlusses ist beim Anlöten des Drahtes längs zur Leiterbahn (a) gering, quer dazu (b) dagegen groß

Die Haltung des LötKolbens

Sie halten den LötKolben am besten wie einen Bleistift zwischen Daumen, Zeige- und Mittelfinger. So können Sie ihn am genauesten führen (Bild 1.10). Im Bedarfsfall können Sie ihn sogar in der Hand umdrehen (Bild 1.10).

Sie setzen den LötKolben nie genau senkrecht auf die Leiterplatte; damit hat er eine Längsrichtung. Achten Sie darauf, daß seine Längsrichtung immer mit der Längsrichtung der Leiterbahnen (bei Experimentierplatten) übereinstimmt (Bild 1.11), denn sonst berühren Sie schnell zwei Leiterbahnen, und schon haben Sie sich einen Kurzschluß auf Ihre Platte gezaubert.



Bild 1.10 Haltung des LötKolbens zwischen Daumen, Zeige- und Mittelfinger

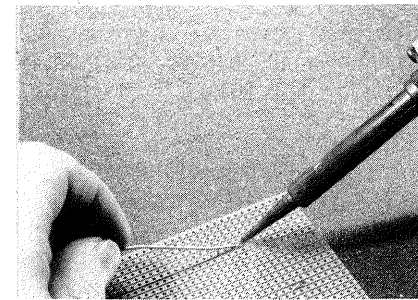


Bild 1.11 Führung des LötKolbens in Längsrichtung der Leiterbahn

Abhilfe: Zinnabsaugen (siehe „Entlöten“). Bei den Platinen brauchen Sie im wesentlichen nur bei den IC-Fassungen darauf zu achten, daß der LötKolben quer zur Reihe der IC-Anschlüsse steht.

Der Lötvorgang

Beim Verlöten von Bauelementanschlüssen, die Sie durch die Bohrungen der Leiterplatte gesteckt haben, setzen Sie die LötKolbenspitze dicht neben den Anschluß auf die Leiterbahn bzw. das Lötauge. Die Kraft, mit der Sie den LötKolben aufsetzen, entspricht etwa dem Druck, den Sie

für das Schreiben mit einem mittelharten Bleistift benötigen.

Dann schieben Sie etwas Zinn in die Kehle zwischen LötKolbenspitze und Leiterbahn. Verwenden Sie so wenig Zinn wie möglich. Es soll sich gerade ein flacher Hügel um den Bauelementanschluß bilden. Achten Sie darauf, daß das Zinn gut fließt (Bild 1.12 links). Dazu müssen Sie die Lötstelle gut durchheizen. Wenn das Zinn von der Lötstelle aberfliegt wie Wasser auf einem gewachsenen Autodach, dann ist die Lötstelle entweder nicht sauber genug oder zu kalt (Bild 1.12 Mitte). Solche Lötstellen bilden keinen oder nur mangelhaften elektrischen Kontakt. Man nennt sie „kalte“ Lötstellen. Sie sind eine der häufigsten Ursachen für das Versagen selbstgebauster Schaltungen.

Abhilfe schafft meist das Nachheizen mit dem LötKolben. Dabei kann es vorkommen, daß wegen des zu langen Heizens das Kolophonium vollständig verdampft. Zwischen dem heißen Lötzinn und der Luft besteht dann nicht mehr der Schutzfilm aus Kolophonium, das Zinn oxidiert. Die Lötstelle sieht nach dem Erkalten grau und rau aus (Bild 1.12 rechts). Auch eine solche Lötstelle ist elek-

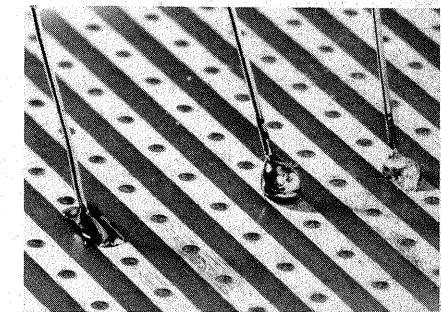


Bild 1.12 Gute Lötstelle (links), schlechte Lötstellen (Mitte, rechts)

trisch kalt. Um sie zu korrigieren, benötigt man etwas Flußmittel. Benetzen Sie die Lötstelle mit etwas Kolophoniumlösung (siehe „Entlöten“), und heizen Sie sie nochmals durch; damit dürfte der Schaden behoben sein.

Das Anlöten der Litzen wurde bereits beim Verzinnen beschrieben. Zu ergänzen ist noch, daß Sie die Litzenenden mit einer schmalen Pinzette halten; am besten eignet sich dazu eine schlanke Pinzette mit abgewinkelten und innen geriffelten Spitzen (Bild 1.13). Halten Sie die Litze gerade nur fest, drücken Sie die Pinzette nicht stark zusammen, denn durch die Lötwärme wird der PVC-Mantel der Litze weich. Sie können ihn mit der Pinzette durchdrücken und haben plötzlich eine blanke Stelle am Draht, die Ihnen unerwarteten Ärger bereiten kann.

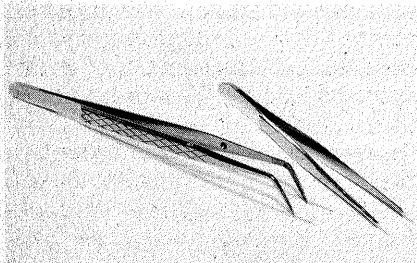


Bild 1.13 Für Elektronikarbeiten geeignete Pinzetten. Wichtig sind schlanke, innen geriffelte Spitzen

Löten üben!

Das Löten ist nicht schwer. Es gelingt nach kurzer Übung – ohne Übung geht es aber nicht! Ehe Sie sich an Ihren Computer wagen, müssen Sie ein Gefühl für das Verhalten des Zinns,

der Litzen usw. entwickeln. Ihre erste Lötstelle sollte nicht gleich eine in Ihrem Computer sein. Falls Sie noch nie gelötet haben, leisten Sie sich eine Experimentierplatte, und löten Sie so viele Drähte fest, wie darauf Platz finden. Das handwerkliche Fingerspitzengefühl, das Sie dabei entwickeln, ist eine unentbehrliche Voraussetzung für alle Elektronikarbeiten.

Entlöten

Es soll Optimisten geben, die Kreuzworträtsel gleich mit dem Kugelschreiber ausfüllen, während normale Sterbliche nicht nur den Bleistift, sondern gelegentlich auch den Radiergummi benutzen. Zählen Sie sich ruhig zur letzten Gruppe, und rechnen Sie mit Verdrahtungsirrtümern, die Sie zu korrigieren haben. Während Sie am Lötwerkzeug nicht sparen sollten, ist es beim Entlöten sogar anzuraten, denn das nachher empfohlene Hilfsmittel ist nicht nur billig, sondern auch unübertrefflich gut.

Zum Entlöten brauchen Sie etwas, das das Zinn von der Lötstelle entfernt. Dazu gibt es im Elektronikhandel Zinnabsauger (Bild 1.14 oben), die flüssiges Zinn „einschlürfen“ können; sie arbeiten im Prinzip wie ein Staubsauger. Sie sind sicherlich nicht



Bild 1.14 Zinnabsauger und Entlötlitze

schlecht, auch wenn sie nicht alles Zinn einsaugen können. Vornehmlich eignen sie sich für große Lötstellen und kosten natürlich auch viel Geld, das Sie ebensogut für eine bessere Lötstation oder für Bauelemente verwenden können.

Daneben führt der Elektronikhandel **Entlötlitze** (Bild 1.14 unten). Sie funktioniert nach dem Löschblattprinzip: Die feinen Kapillaren zwischen den Fasern saugen Flüssigkeit vollständig auf. Entlötlitze ist an kleinen Lötstellen das beste Hilfsmittel, denn damit können Sie das Lötzinn bis auf einen hauchfeinen Film, der auf der Kupferbahn immer zurückbleibt, vollkommen entfernen, auch wenn Sie versuchen, das flüssige Zinn abzuwischen. Die Hersteller sind stolz auf ihre Produkte – entsprechend stolz sind auch die Preise.

Entlötlitze ist ein Drahtgeflecht aus sehr feinen Drähten, an denen etwas Flußmittel, z. B. Kolophonium, klebt. Stellen Sie sich Ihre Entlötlitze selbst her (Bild 1.15). Dazu benötigen Sie **hochflexible** Litze (LiFY) mit 0,5 mm Ø; „normale“ Litze, wie Sie sie z. B. in Netzkabeln für allgemeine Anwendungen (NYFAZ, NYLHY) vorfinden, ist zu grob.

Außerdem brauchen Sie ein kleines Glas mit einem dicht schließenden Deckel, z. B. ein Gewürzglas mit 20 bis 50 ml. Nehmen Sie kein großes Glas. Ein großes Glas bereitet Ihnen außerdem beim Umkippen großen Ärger. Lösen Sie in diesem Glas einige cm³ Kolophonium (Drogerie, Apotheke, Musikalienhandel, Geigenbauer) in Alkohol (90%) oder Brennspritus. Das billigste Kolophonium ist für Ihren Zweck das beste. Nun isolieren Sie die hochflexible Litze einige cm ab, nicht länger; Sie brauchen die übrige Isolation als

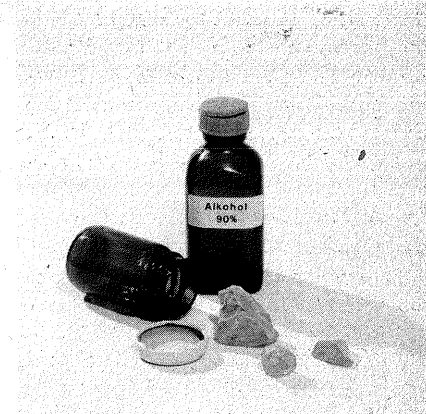


Bild 1.15 Zutaten zur Herstellung von Entlötlitze: Kolophonium und Alkohol

Wärmeschutz. Die freigelegte Litze tauchen Sie in die Kolophoniumlösung, setzen das Ende auf die Entlötlitze und drücken den heißen LötKolben darauf. Sie werden überrascht sein, wie gut der Löschpapiereffekt auch hier wirkt. Eventuell müssen Sie den Vorgang wiederholen, weil Sie mit dem LötKolben immer nur eine kleine Stelle Ihrer Entlötlitze durchheizen können.

Voraussetzung für einwandfreies Entlöten ist eine ausreichende Wärmekapazität Ihres LötKolbens. Seine Leistung sollte nicht weniger als 30 W betragen.

Sollte Ihnen beim Entlöten zu viel Kolophonium auf die Leiterbahn geraten sein, so tauchen Sie einen Tuschepinsel in Alkohol (oder Brennspritus, wenn der Geruch Sie nicht stört), und pinseln Sie es ab.

Eine **Warnung** zu guter Letzt: Sollte Ihnen das Glas auf Ihrem Arbeitsplatz einmal umkippen, so werden Sie noch lange daran denken. Auch wenn Sie Ihren Tisch sofort mit Alkohol abwischen: Er wird noch eine

Weile klebrig bleiben. Sichern Sie Ihr Glas gegen Umkippen, indem Sie es z. B. mit Pattex auf ein Stück kräftige Pappe kleben. Wenn Sie die Möglichkeit zu Holzarbeiten haben, können Sie sich z. B. in einen Holzblock Löcher einlassen, in die dann Glas und Litze passen (Bild 1.16).

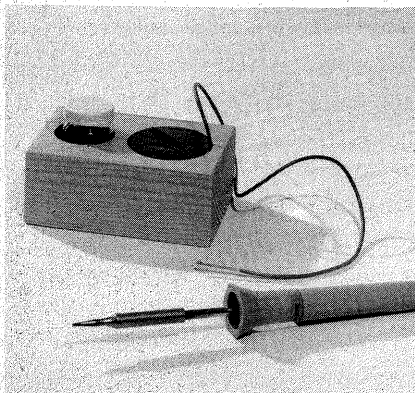


Bild 1.16 Sicherung der klebrigen Kolophoniumlösung in einem Holzblock

Das Setzen von Lötnägeln

Auf die Leiterplatten werden auch sogenannte Lötnägel (RTM-Stifte) mit 1,3 mm Ø montiert. Sie sollen hauptsächlich in der Testphase den Anschluß der Versorgungsspannung und später den Abgriff bestimmter Signale ermöglichen. Die Lötnägel erhalten ihre mechanische Stabilität durch die feste Einpressung in die Leiterplatte, kaum durch das Lötten. Die Lötstelle darf nur unwesentlich belastet werden; die aufgeklebte Kupferbahn ist ja nur 35 µm (0,035 mm) stark und hält mechanischen Belastungen nicht stand.

Lötnägel (Bild 1.17) bestehen aus drei Teilen, dem Steckerstift (1), auf den

ein Kabelschuh geschoben werden kann, dem Kragen (2), bis zu dem der Nagel unbedingt in die Leiterplatte zu schieben ist (der Kragen muß fest aufsitzen!), und dem eigentlichen Nagelteil (3).

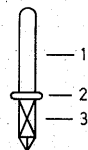


Bild 1.17 Lötnagel mit vierkantigem Querschnitt des Nagelteils (3); der Kragen (2) muß vor dem Anlöten fest auf der Leiterplatte aufsitzen

Bei manchen Lötnagelarten hat der Nagelteil einen runden Querschnitt. Sie sind für Maschinenmontage vorgesehen und eignen sich **nicht für Handmontage**. Das Loch, in das sie gepreßt werden, muß sehr genau gebohrt sein. Ist es nur geringfügig zu eng, sprengt ein runder Nagel eine Pertinaxplatte (in Epoxydplatten bekommen Sie ihn nur mit roher Gewalt); ist es nur geringfügig zu groß, sitzt der Nagel nicht fest.

Für Handmontage sind Lötnägel mit vierkantigem Querschnitt des Nagelteils geeignet.

Bohren Sie die Leiterplatte mit 1,3 mm vor. Wenn Sie den Nagel in die mehr oder weniger genaue Bohrung pressen, indem Sie mit einer Spitzzange auf den Kragen drücken, schneiden sich die scharfen Kanten in die Platte ein; der Lötnagel sitzt sicher. Danach können Sie ihn mit der Leiterbahn verlöten.

Bei Epoxydplatten kann es Schwierigkeiten mit dem Eindringen der Nägel geben, denn sie sind sehr hart. Dann erweitern Sie die Bohrung, in-

dem Sie nochmals bohren und den Bohrer dabei in einer Richtung leicht hin und her verkanten.

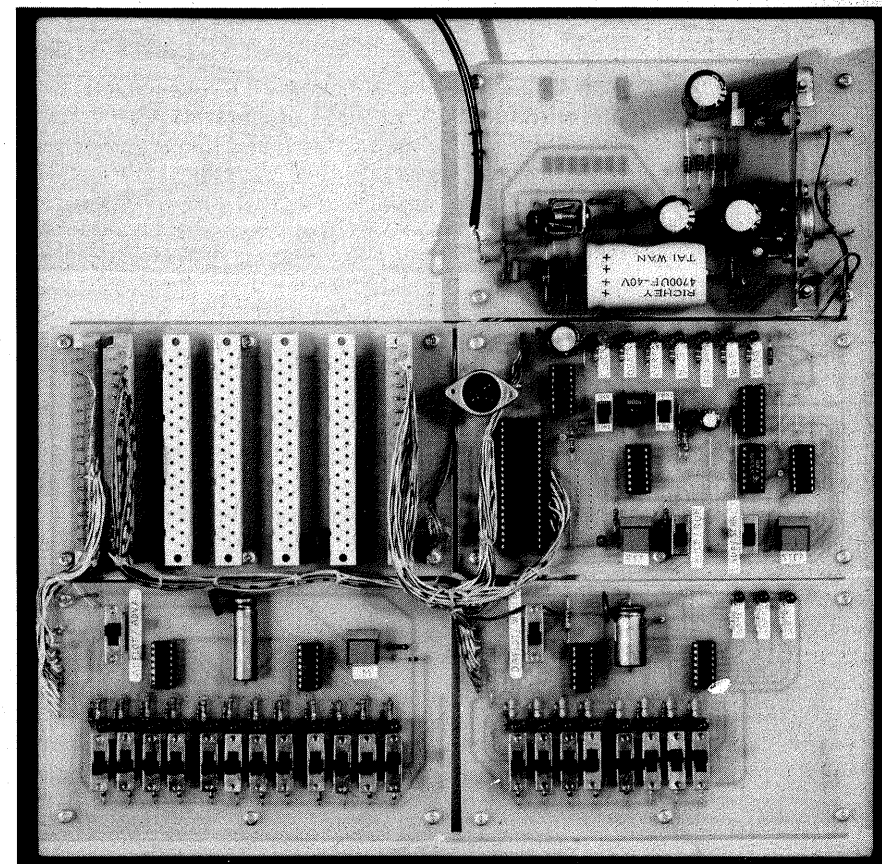
Die Montage der Leiterplatten

Die Grundplatten Ihres Mikrocomputers (Netzteil, Busplatte, CPU-Platte, Daten- und Adreßeinheit) können Sie mit Distanzröllchen (Elektronikhandel) auf eine Sperrholzplatte 8 mm × 330 × 330 mm montieren (Bild 1.18). Die Bohrungen für die Befestigungsschrauben sollen im Endzustand 3,2 mm Ø haben. Zu-

nächst bohren Sie sie aber nur mit 1 mm oder 1,3 mm Ø: Dann legen Sie die Leiterplatten auf die Sperrholzplatte und stechen die Befestigungspunkte durch. Nun können Sie die Stellen, in die Sie die Schrauben setzen wollen, maßgenau vorstechen oder vorbohren.

Wenn Sie Ihre Leiterplatten auf 8 mm hohe Distanzröllchen setzen, so verwenden Sie am besten Spax-Schrau-

Bild 1.18 Montage der ersten Baueinheiten auf einer Sperrholzplatte



ben (Rundkopf mit Kreuzschlitz) 3,0 × 16 mm zusammen mit Unterlegscheiben (möglichst aus Nylon). Senkkopfschrauben sind nicht so gut geeignet. Wenn Sie sie in den Experimentierplatten aus Pertinax versenken, können ihre keilförmigen Köpfe den spröden Werkstoff sprengen. Versenken Sie sie nicht, so stehen die scharfen Kanten ihrer Köpfe frei, und zwar im Bedienbereich. Bei unachtsamer Bedienung können Sie sich daran verletzen, und beim Computern haben Sie an anderes zu denken als an Schraubenköpfe.

Der Aufbau auf Experimentierplatten

Für jede Baueinheit benötigen Sie eine Experimentierplatte im Europaformat (100 × 160 mm), einseitig kaschiert mit Kupferbahnen in Längsrichtung, Rastermaß 2,5 mm. Achten Sie auf das Rastermaß! Es gibt auch das Rastermaß 2,54 mm (1/10 Zoll); in dieses bekommen Sie die Federleisten der Busplatte und die Stiftleisten der anderen Platten nicht hinein. Es genügen Pertinaxplatten (Hartpapier Klasse IV).

Trotz einheitlicher Abmessungen sind die Karten nicht gleich beschaffen; manche haben 39 Kupferbahnen, andere nur 37 (das sind meist die mit dem Rastermaß 2,54 mm, also aufpassen!). In den abgebildeten Beispielen wurden Karten mit 39 Leiterbahnen verwendet. Da die Anzahl der Bahnen jedoch nie ganz ausgenutzt ist, können Sie auch Karten mit 37 Bahnen verwenden. Sie zählen dann beim Nachbau die beiden äußeren Bahnen nicht mit; ansonsten ändert sich nichts am Aufbau.

Die Europakarte besitzt über ihre Länge in der Regel 63 Lochreihen.

Auch ihre Anzahl wird nie voll ausgenutzt. Bisweilen erhält man zu sehr günstigen Preisen Experimentierplatten, die geringfügig zu kurz sind. Wenn **nicht mehr** als vier Lochreihen fehlen (Gesamtlänge der Platte ca. 150 mm), sind die Karten noch zu verwenden; hier lassen sich Sonderangebote ausnahmsweise gut ausnutzen. Hat man verschieden lange Karten, so sind die kürzeren für das Netzteil, die Datenein- und -ausgabe, die Speicherplatte, den IO-Baustein und den AD-Wandler zu verwenden, die längeren für die Busplatte, die CPU-Platte sowie die Adreßeingabe und -anzeige. Schmalere Platten sind oft länger als die Europakarte. Sie haben oft das Rastermaß 2,54 mm! Wenn nicht, so können Sie sie auch verwenden, denn sie lassen sich leicht mit der Laubsäge (feines Metallsägeblatt) kürzen; dabei sägt man von Loch zu Loch, wobei die Kupferseite oben liegen muß, denn sonst könnten die Kupferbahnen ausreißen.

Solche Platten sind meist abschreckend teuer, und daher bietet es sich von selbst an, sie zu meiden und nur im Notfall zu kaufen.

Ferner unterscheiden sich die Platten in der Streifenbreite (Bild 1.19). Die schmalen Bahnen sind etwa 1,5 mm breit; der Abstand voneinander beträgt etwa 1 mm. Wer beim Löten zwischen diesen Bahnen einen Kurzschluß durch überfließendes Zinn erreichen will, muß sich schon mächtig anstrengen. Nachteil: Neben den Bohrungen sind die Kupferbahnen sehr dünn. Die **Hauptstromversorgungsleitungen** (Masse und +5 V) müssen u.U. durch Auflöten eines dünnen Drahtes verstärkt werden. Die Ströme auf den übrigen Leitungen sind so gering, daß sich die Leiterbahnbreite nicht auswirkt.

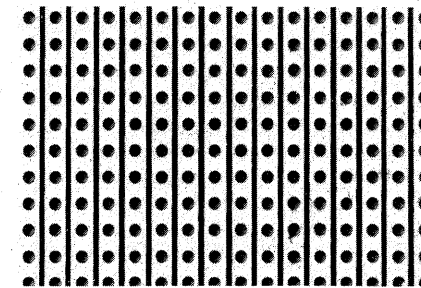
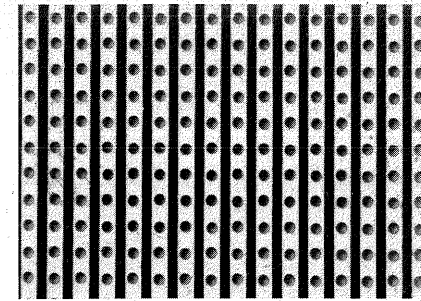


Bild 1.19 Experimentierplatten im Rastermaß 2,5 mm mit unterschiedlicher Breite der Kupferbahnen bzw. der Isolierstreifen

Die breiten Bahnen sind etwa 2,1 mm breit, der Abstand zwischen ihnen beträgt nur ca. 0,4 mm. Hier ist es durchaus möglich, mit überfließendem Zinn zwischen den Bahnen einen Kurzschluß zu erzeugen. Auch muß man bei den Unterbrechungen genauer darauf achten, daß nicht irgendwo eine feine Kupfernadel stehenbleibt (siehe Seite 29). Dafür braucht mit Ausnahme der Netzteilplatte und der Busplatte keine Leiterbahn verstärkt zu werden.

Die Platten mit den breiten Leiterbahnen sind oft erheblich billiger als die mit den schmalen. Hier dürfen Sie sparen; bei den eigentlichen Bauelementen sollten Sie sich jedoch vor Sonderangeboten hüten.

Die Orientierung auf der Experimentierplatte

Die einfachste Art, sich auf der Experimentierplatte zu orientieren, ist es, sich auf der Bestückungsseite mit einem Filzschreiber (keinesfalls Bleistift! Graphit leitet!) die Plätze für die IC-Fassungen zu markieren. Danach stecken Sie die Fassungen ein und löten sie vorerst nur an zwei diagonal zueinander liegenden Stiften, z. B. bei Fassungen DIL 14 an Pin 1 und 8, fest – aber nur an zweien, sonst wird Ihnen das Unterbrechen der Leiterbahnen schwerfallen (siehe unten)! Nun markieren Sie sich den Anschluß 1 des jeweiligen IC, von dem aus werden Sie später beim Verdrahten zählen. Daher lohnt es sich, auch auf der Kupferseite beim Stift 1 einen Punkt anzubringen.

Das Unterbrechen der Leiterbahnen

Unterbrechen Sie Leiterbahnen grundsätzlich nur an Bohrungen, nicht dazwischen. Bei diesem engen Rastermaß spielt der größere Platzbedarf eine nur unwesentliche Rolle, dafür können Sie die Leiterbahnunterbrechungen aber leichter kontrollieren.

Zum Unterbrechen der Leiterbahn benötigen Sie einen **scharfen** Spiralbohrer HSS, 4 mm Ø. Sie setzen seine Spitze bei einer Bohrung auf die Kupferbahn und drehen ihn unter sanftem Druck mit Daumen und Zeigefinger. Dabei schälen Sie die Leiterbahn mit der beginnenden Bohrung ab. Sie erleichtern sich die Arbeit erheblich, wenn Sie über den Bohrer schieben, oder noch besser ist es, sich einen Griff anzufertigen. Dazu bohren Sie mit dem Bohrer, den Sie

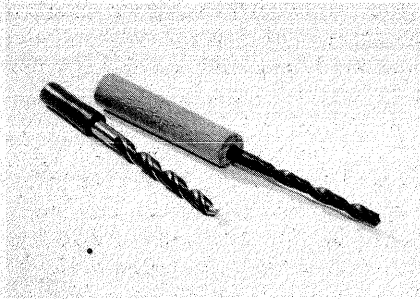


Bild 1.20 Zum Leiterbahnunterbrecher umfunktionalisierter Spiralbohrer mit 4 mm Ø, HSS, mit Isolierschlauch (Mantel eines Kabels, links) oder Dübelschiff (rechts) als Griff. Der Bohrer sollte nicht für andere Arbeiten benutzt werden, damit er lange scharf bleibt

als Leiterbahnunterbrecher benutzen wollen, ein etwa 4 cm langes Dübelschiff (8 bis 10 mm Ø) ein Stück auf und schieben es als Griff auf den Bohrer. Ein Tropfen Alleskleber, vorher in das Bohrloch gegeben, gibt dem Ganzen genügend Halt (Bild 1.20).

Wichtig ist, daß Sie das Dübelschiff wenigstens annähernd genau axial aufbohren. Mit einer Ständerbohrmaschine ist das nicht schwer; wenn Sie es freihändig schaffen müssen (Dübelschiff im Schraubstock, kleine Handbohrmaschine), gelingt es vielleicht nicht beim ersten Mal. Versuchen Sie es mehrmals; die Mühe lohnt sich, denn angesichts der großen Zahl der Leiterbahnunterbrechungen wirkt sich die gewonnene Arbeitserleichterung schnell aus. Nach dem Aufschneiden der Leiterbahnen bürsten Sie die Späne mit einer Zahnbürste in Längsrichtung der Bahnen ab (Bilder 1.21 und 1.22). Sehr viele Leiterbahnunterbrechungen sitzen zwischen den IC-Anschlüssen

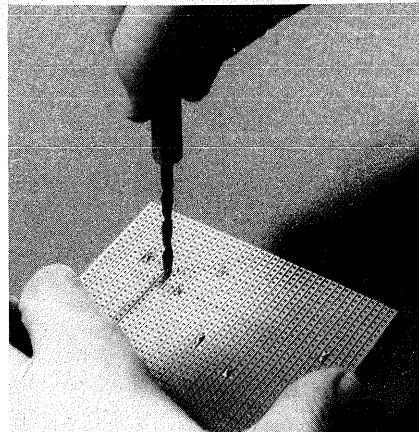


Bild 1.21 Führung des Leiterbahnunterbrechers

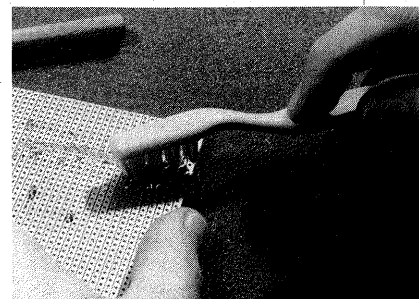


Bild 1.22 Sehr wichtig ist das Abbürsten der Späne. Mancher Kurzschluß entstand durch quergestellte, angelötete, kaum sichtbare Späne.

sen. Da sich bei DIL 8-, 14-, 16-, 18- und 20-Fassungen nur zwei „Löcher“ zwischen den IC-Anschlußreihen befinden, muß die Leiterbahn notgedrungen unmittelbar neben einer IC-Lötstelle unterbrochen werden. Das ist sehr schwer, wenn schon alle Anschlüsse der IC-Fassung angelötet sind, denn in dem Fall muß ja auch ein Teil der Lötstelle aufgeschnitten

werden. Deshalb ist es praktisch, die IC-Fassungen vorerst nur an zwei Anschlüssen anzuheften, dann die Leiterbahnen zu unterbrechen (dann gibt es keine Orientierungsprobleme mehr) und erst zuletzt alle IC-Stifte anzulöten.

Falls Sie lieber nicht zu Beginn Ihrer Arbeit alle Leiterbahnunterbrechungen herstellen, sondern dies kontinuierlich, gewissermaßen von Bauelement zu Bauelement tun wollen (es gibt gute Gründe dafür, z. B. den, daß man beim Aufbau den Stromlaufplan verfolgt und dabei die Gefahr einer falschen Leiterbahnunterbrechung vermeiden will), so gilt grundsätzlich: Liegt eine Leiterbahnunterbrechung unmittelbar neben einer Lötstelle, so ist zuerst die Leiterbahnunterbrechung auszuführen.

Sollten Sie sich beim Auftrennen der Leiterbahn einmal geirrt haben, so löten Sie einen **dünnen** Draht über die Unterbrechung (Bild 1.23). Sie lösen aus einer kräftigen Litze, wie sie z. B. für Netzkabel verwendet wird, ein Einzeldrätchen heraus und winkeln etwa 5 mm davon ab. Sie halten es am „langen“ Ende und verzinnen das „kurze“ sowie die beiden Enden der Leiterbahn. Dann drücken Sie mit dem heißen LötKolben zuerst das dem Winkel abgewandte Drahtende a) in das auf der Leiterbahn befindliche Zinn. Das Drahtende soll höchstens 1 mm auf die Leiterbahn reichen. Anschließend drücken Sie den

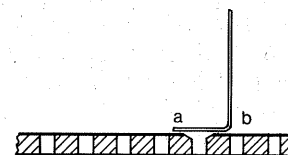


Bild 1.23 Korrektur einer irrtümlichen Leiterbahnunterbrechung

Draht in das Zinn des gegenüberliegenden Leiterbahndendes b) und kappen den überschüssigen Draht nach dem Erkalten der Lötstelle.

Mit dicken Drähten sind unterbrochene Leiterbahnen schlecht zu reparieren. Diese leiten nämlich viel Wärme. Wenn Sie die zweite Seite anlöten, können Sie sehr leicht das vorher angelötete Drahtende wieder ablösen. Außerdem übertragen dicke Drähte die Handbewegungen auf die Leiterbahn. So kann es vorkommen, daß Sie bei wiederholten Lötversuchen die Leiterbahndenden von der Platte ablösen.

Die Kontrolle der Leiterbahnunterbrechungen

Es ist unerlässlich, daß Sie **jede** Leiterbahnunterbrechung **sofort** genau kontrollieren. Oft bleiben am Rande der Unterbrechung haarfeine Kupfernadeln stehen (Bild 1.24). Sie sind so fein, daß sie mit bloßem Auge oft nicht zu erkennen sind. Eine starke Lupe (8fach) hilft in vielen Fällen, auch eine Bewegung im Licht der Schreibtischlampe, denn Lichtreflexe machen auf die Kupfernadeln aufmerksam.

Am sichersten ist die Kontrolle mit

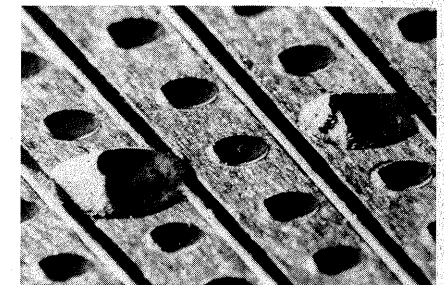


Bild 1.24 Unvollkommene (links) und gute (rechts) Leiterbahnunterbrechung

dem als Ohmmeter geschalteten Vielfachmeßinstrument. Sie wählen einen mittleren Ohmmeßbereich (z. B. $R \times 1k$). Dabei ist es nicht erforderlich, den Nullpunkt genau zu justieren. Sie müssen sich aber vergewissern, daß der Zeiger weit ausschlägt, wenn Sie die Spitzen der Meßkabel aneinanderhalten. Nun setzen Sie je eine Meßspitze links und rechts neben der Leiterbahnunterbrechung auf die Enden der Kupferbahn (Bild 1.25). Schlägt der Zeiger des Meßinstruments aus, zeigt er also einen Strom an, so ist die Leiterbahn nicht vollkommen aufgetrennt. Sie entfernen die stehengebliebene Kupfernadel in der Regel besser mit einem Abbrechklingenmesser als mit dem Bohrer – und vergessen Sie nicht, die Unterbrechung nochmals zu kontrollieren. Nur wenn der Zeiger des Meßinstruments **nicht** ausschlägt, d. h. ∞ Ohm anzeigt, ist die Bahn vollkommen durchtrennt.

Überprüfen Sie also die Leiterbahnunterbrechungen sofort!

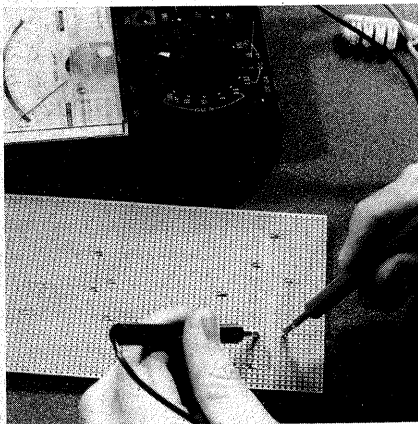


Bild 1.25 Kontrolle einer Leiterbahnunterbrechung mit dem als Ohmmeter geschalteten Vielfachinstrument

In der Fehlerstatistik stehen unvollkommene Leiterbahnunterbrechungen noch vor den berüchtigten kalten Lötstellen. Sie sind leicht zu finden, solange noch keine Bauelemente eingelötet sind; IC-Fassungen gelten hierbei nicht als Bauelemente. Die Fehlersuche in einem fertigen Werk ist im Vergleich dazu fast eine Sisyphusarbeit. Die kleine Mühe der sofortigen Kontrolle lohnt sich!

Sonderfälle beim Verdrahten

Die Verbindungen zwischen den Bauelementen werden überwiegend durch Drähte hergestellt. Soweit einzelne Drähte angelötet werden, gibt es keine Besonderheiten. Oft treffen sich an einem Punkt jedoch zwei Drähte (Bild 1.26). Die können Sie als einen Draht verstehen, der einen Punkt berührt und dann weiterläuft. Solche Verdrahtungsketten beginnen Sie grundsätzlich an einem „Doppelpunkt“, indem Sie dort eine Schlaufe anlöten (Bild 1.27). Dazu isolieren Sie eine mindestens 20 cm lange Litze an beiden Enden gut 1 cm lang ab und verdrillen die blanken Enden stramm bis an den Isoliermantel. Dann verzinnen Sie die Stelle und kürzen sie auf 2 mm, also etwas länger als einzelne Litzen, damit die Verdrillung beim Anlöten nicht aufspringt. Wie einzelne Litzen werden sie immer in Längsrichtung mit der Leiterbahn verlötet.

Nun nehmen Sie mit der Schlaufe Maß bis zur nächsten Lötstelle und geben ca. 1 cm zu. An der Stelle trennen Sie die Schlaufe auf und löten das Ende fest, z. B. an c) in Bild 1.26. Nun führen Sie das zweite Ende der Schlaufe an Punkt b); der ist in Bild 1.26 wieder ein Doppelpunkt. Nehmen Sie Maß, geben Sie wieder ca.

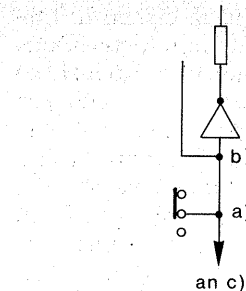


Bild 1.26 Drei Leitungsstriche an einem Punkt kann man als zwei Drähte verstehen, die sich an einem Punkt (Schalter- oder IC-Anschluß, Leiterbahn) treffen, oder auch als einen Draht, der den Punkt berührt und dann weiterläuft

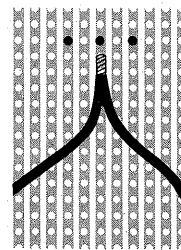


Bild 1.27 Anlöten einer „Doppelleitung“ auf eine Leiterbahn. An der Lötstelle liegen die beiden Drähte übereinander, nicht nebeneinander, sonst würde die unisolierte Stelle, an der sich die Drahtenden verzweigen, zu breit

1 cm zu und isolieren ca. 1 cm ab; isolieren Sie ein gleichfarbendes Litzenstück ebenfalls ca. 1 cm lang ab, verdrillen die beiden Enden, verzinnen und kürzen die Verbindungsstelle und löten sie an b). Weitere „Doppelpunkte“ behandeln Sie ebenso. So können beliebig lange Ketten gebildet werden.

Wenn mehr als zwei Drähte an einen Punkt anzulöten sind, so verdrillen Sie trotzdem immer nur je zwei; sonst wird der Drahtwulst zu dick; außer-

dem springt die Verdrillung um so eher auf, je mehr Drähte Sie miteinander verdrillen, und dann gibt es sehr schnell Kurzschlüsse. Löten Sie daher in einem solchen Fall lieber mehrere Schlaufen hintereinander auf eine Leiterbahn.

Der Aufbau auf Platinen

Sie können die Platinen fertig geätzt, aber nicht gebohrt bei der auf Seite 12 genannten Firma kaufen. Damit dürfte der Nachbau des Mikrocomputers problemlos sein.

Andererseits wird es Sie vielleicht reizen, auch die Platinen selbst zu fertigen. Sie sind nur einseitig kaschiert und außerdem vergleichsweise grob ausgelegt. Das bedingt zwar einige Drahtbrücken mehr, dafür sind die Platinen aber auch ziemlich sicher selbst herzustellen. Im Prinzip geschieht dabei folgendes:

Sie benutzen eine Isolierplatte, auf die eine Kupferfolie von 35 μ m Stärke aufgeklebt ist. Diese Platte ist das „Basismaterial“. Sie übertragen das Muster der Leiterbahnen mittels Fotopositivlack auf das Basismaterial, wobei die späteren Leiterbahnen nach dem Entwickeln des Fotolacks abgedeckt sind, die späteren Isolierstellen dagegen nicht. Im Ätzbad verschwindet alles nicht abgedeckte Kupfer, so daß nur noch die Leiterbahnen vorhanden sind. Die Platte wird an den Lötäugen durchbohrt, und das Werkstück, die Platine, ist fertig.

Das Basismaterial

Als Basismaterial reicht Pertinax (Hartpapier KI.IV). Diese Platten sind nicht nur wesentlich billiger als Platten aus Glasfaser-Epoxydharz,

sondern sie sind auch erheblich leichter zu bohren. Beobachtungen bei Schülerarbeiten ergaben, daß dies ein wesentlicher Gesichtspunkt ist. Besonders gut zu bohren ist Basismaterial aus Epoxydharz ohne Glasfaserverstärkung. Wer in seinem Elektronikladen die Auswahl hat, sollte es unbedingt bevorzugen. Es sieht etwa so aus wie Pertinax, ist aber hellgelb. Am besten kaufen Sie sich schon mit Fotopositivlack beschichtete Platten. Die Selbstbeschichtung lohnt nur, wenn Sie sehr viele Platten herstellen wollen.

Fotopositivlack erhalten Sie in Spraydosen in jedem Elektronikgeschäft, z. B. POSITIV-20 von Kontakt Chemie. Eine 75-ml-Dose (die kleinste, die in den Geschäften handelsüblich ist) ist für Ihren Mikrocomputer reichlich genug. Der Fotolack ist nämlich nur eine bestimmte Zeit haltbar. Was Sie davon nicht bis zum auf der Dose angegebenen Verfallsdatum oder kurz danach verbrauchen, ist für Sie wertlos.

Das Beschichten mit Fotolack

Zuerst scheuern Sie das Kupfer mit Scheuerpulver blank, spülen es mit viel Wasser und trocknen es gut ab. Alle Oxid- und Fettreste müssen verschwunden sein. Nach dem Scheuern dürfen Sie es nicht mehr mit der bloßen Haut berühren.

Die Hauptschwierigkeit beim Selbstbeschichten besteht darin, den Fotolack bei gedämpftem Licht (dicke Gardine zuziehen genügt) **dünn, gleichmäßig und vor allem staubfrei** aufzubringen. Eine staubarme Atmosphäre finden Sie am ehesten in einem feuchten Raum (z. B. Keller). Bewegen Sie sich langsam, um nicht Staub (auch aus Ihrer Kleidung) auf-

zuwirbeln. Wischen Sie einen Schuhkarton mit einem feuchten Lappen aus. Der Karton soll nicht gerade naß, aber doch angefeuchtet sein. Legen Sie den Deckel gewissermaßen auf den Rücken und sprühen Sie aus 20 bis 30 cm Abstand den Lack in Mäanderlinien kreuzweise auf, bis die aufgesprühte Schicht gerade porig-fleckig wie Hammerschlaglack aussieht. Die Lackschicht glättet sich schnell von selbst. Achten Sie darauf, daß der Lack überall „dicht“ ist; er muß nach dem Verlaufen über die gesamte Fläche porenfrei spiegeln; dann ist er dick genug aufgetragen. Zu dick ist er, wenn sich am Rand der Platte ein dicker Rand ansammelt. In dem Fall sollten Sie den Lack lieber mit einem nichtfusselnden Tuch abwischen und neu aufsprühen.

Nun stülpen Sie den eigentlichen Schuhkarton über die Platte und lassen den Lack bei Zimmertemperatur 24 Stunden lang trocknen. Sie können ihn auch im Backofen bei etwa 50 bis höchstens 70°C „einbrennen“; dann brauchen Sie nur etwa eine gute halbe Stunde zu warten. Im Innern des Backofens muß es dunkel sein (Lampe herausrauben, Fenster abdunkeln).

Vorsicht: Beim Sprühen und Trocknen des Fotolacks werden gesundheitsschädliche Dämpfe frei. Sorgen Sie daher für gute Lüftung.

Das Belichten

Der Fotolack reagiert auf UV-Strahlung. Die beste Strahlungsquelle ist die Sonne; ihre Strahlung reicht auch bei leichter bis mittlerer Bewölkung, denn die Wolkendecke absorbiert nur etwa 20% der UV-Strahlung. Die Sonnenstrahlen kommen auf der Erde nahezu parallel an; daher wird das

Leiterbahnmuster auch an den Stellen fehlerfrei übertragen, an denen die Vorlage nicht fest auf dem Fotolack aufliegt.

Geeignet sind ferner alle Lampen mit einem hohen UV-Lichtanteil, z. B. eine Höhensonne oder spezielle Glühlampen, die in Elektronikgeschäften (für viel Geld) zu haben sind, z. B. Vitalux 300 W oder Nitraphot S 250 W.

Die Vorlagen für die Platinen finden Sie im Anhang 1 dieses Buches spiegelbildlich abgedruckt. Lösen Sie die Blätter heraus, und sprühen Sie das Blatt, das Sie kopieren wollen, satt mit Klarpausspray (Zeichenbedarfs-handel) ein. Das Blatt ist als Kopier-vorlage so lange zu verwenden, wie es feucht ist. Sobald es getrocknet ist, nimmt es seine alte Qualität wieder an; Sie können es für weitere Kopien neu besprühen.

Ferner brauchen Sie noch eine feste Unterlage, z. B. Hartfaser- oder Kunststoffplatte, etwa Größe DIN A5, eine gleichgroße Plexiglasplatte von ca. 3 bis 4 mm Stärke (Silikatglas absorbiert UV-Strahlung, weshalb Sie hinter der Fensterscheibe auch nicht braun werden, Plexiglas dagegen kaum) sowie sechs Wäscheklammern.

- Sie legen die Plexiglasplatte auf den Tisch;
- darauf die transparent gemachte Vorlage mit dem Platinenbild **nach oben**;
- darauf die Leiterplatte mit der Fotoschicht **nach unten** (die bedruckte Seite der Vorlage und die Fotolackschicht berühren einander!);
- darauf die Hartfaserplatte o. ä.;
- klemmen Sie alles mit den Wäscheklammern zusammen (Bild 1.28, 1-4)
- und setzen Sie das Paket so der UV-Strahlung aus, daß diese mög-

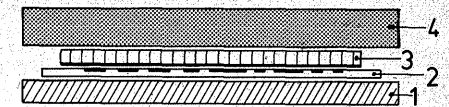


Bild 1.28 Schichtung zur Fotokopie des Leiterbahnbildes:

1: Plexiglasplatte, 2: mit Klarpausspray transparent gemachte Kopiervorlage, bedruckte Seite oben, 3: mit Fotolack beschichtetes Basismaterial, Fotoschicht unten, 4: Hartfaser- oder Kunststoffplatte

lichst senkrecht durch die Plexiglasscheibe (1) und die Vorlage (2) auf den Fotolack (3) trifft.

Die Belichtungszeit müssen Sie an einigen Proben ermitteln, dazu genügen kleine Stücke. Sie hängt von so vielen Variablen ab, daß es unmöglich ist, Ihnen eine genaue Zeit anzugeben (UV-Anteil, Dicke und Alter des Lacks, Transparenz der Vorlage usw.).

Bei hochtransparenten, an den abdeckenden Stellen dagegen sehr dichten Vorlagen, z. B. Filmen, darf die Belichtungszeit in weiten Grenzen schwanken. Das bißchen Drucker-schwärze der Buchvorlagen kann nicht so dicht sein wie ein Film. Daher wird der Rahmen für die Belichtungszeit enger. Im allgemeinen werden Sie eher darauf achten müssen, daß die Belichtungszeit nicht zu lang wird.

Als grobe Richtwerte können gelten: In der „hellen“ Jahreszeit und mit der Höhensonne reichen 90 bis 120 Sekunden, bei bedecktem Himmel geben Sie noch 30 Sekunden, eventuell mehr dazu. In der „dunklen“ Jahreszeit und bei Verwendung von Lampen kann eine Belichtung bis zu 6 Minuten erforderlich sein.

Schon beim Abnehmen der Vorlage können Sie erkennen, ob die Belich-

tung in etwa richtig war; Sie werden nämlich beobachten, daß die belichteten Flächen in der Spiegelung anders schillern als die abgedeckten. Sind keine Unterschiede wahrzunehmen, war die Belichtung ganz falsch. Die Art des Belichtungsfehlers erkennen Sie allerdings erst beim Entwickeln.

Das Entwickeln

Der belichtete Fotolack ist in Natronlauge schnell löslich; der unbelichtete löst sich ebenfalls in Natronlauge, nur sehr viel langsamer.

Lösen Sie für das Entwicklerbad 7 bis 9 g Ätznatron in 1 l Wasser auf. Sie erhalten im Elektronikhandel Tütchen mit der für 1 l Wasser abgewogenen Menge.

Wischen Sie die öligen Reste des Klarpaussprays schnell mit einem Papiertaschentuch ab, legen Sie die Leiterplatte mit der Fotoschicht nach oben in eine **Kunststoffschale** (keinesfalls in ein Metallgefäß), und gießen Sie so viel Natronlauge darauf, daß die Platte schnell und möglichst überall gleichzeitig 3 bis 4 mm hoch bedeckt ist. Schwenken Sie die Schale leicht, oder besser: Wischen Sie mit einem großen, weichen Tuschepinsel über die Fotoschicht. Der Fotolack löst sich mit Schlieren auf. Wischen Sie die Schlieren so lange weg, bis alle Stellen, die frei sein sollen, metallisch blank glänzen.

Der Entwicklungsvorgang kann sehr unterschiedlich lange dauern und hängt nicht zuletzt von der Temperatur des Entwicklerbades ab. Bei kellerkühlem Entwickler kann das Entwickeln 2 Minuten oder länger dauern, bei Temperaturen, die deutlich über 20°C liegen, verkürzt sich die Zeit rapide. Auch die Konzentration

des Entwicklerbades spielt eine erhebliche Rolle. Sie benutzen ja immer nur einen kleinen Teil Ihres Entwicklerbades. Das verbrauchte Bad spülen Sie gleich weg und nehmen für die nächste Platte wieder neuen Entwickler. Sie können den ungebrauchten Entwickler in einer **dicht schließenden Kunststoffflasche** aufbewahren. Die Hersteller des Fotolacks empfehlen, den Entwickler immer frisch anzusetzen.

Wenn die Platine blank wird und es Ihnen um die Feinheiten geht, werden Sie das Entwicklungstempo herabsetzen wollen. Lassen Sie etwas kaltes Wasser in das Entwicklerbad, und Sie können Ihre Platine in Ruhe ausarbeiten.

Belichtungsfehler

Wenn sich im Entwicklerbad nichts tut oder die Ätzstellen nicht richtig blank werden wollen, war die Belichtung zu kurz; lösen sich dagegen auch die Leiterbahnen gleich auf, haben Sie zu lange belichtet.

Entwicklungsfehler

Wenn sich größere Teile des Leiterbahnbildes aufgelöst haben, ist die Lackierung nicht mehr zu retten. Entschichten Sie die Platte (Aceton und Papiertaschentuch), und beginnen Sie von vorn.

Der Lack der Leiterbahnen muß auch nach dem Entwickeln noch glänzen; wenn er matt aussieht, ist auch er vom Entwickler angegriffen. Auch in dem Fall entschichten Sie die Platte und sprühen sie erneut mit Fotolack ein.

Kleine Löcher können Sie mit einem wasserfesten Faserschreiber (Edding 3000, Dalo PC o. ä.) abdecken. Auch

die „Pickel“ von Staubkörnern sollten Sie unbedingt überdecken, weil der Lack an den Stellen meistens undicht ist.

Lackstückchen zwischen zwei Leiterbahnen können Sie mit einem spitzen, aber nicht zu scharfen Messer fortkratzen; und wenn die Kratzer zu groß geraten sind, und Sie versehentlich auch eine Leiterbahn beschädigt haben, korrigieren Sie die Stelle mit dem Faserschreiber.

Beim Ätzen müssen sich **alle** blanken Stellen sofort verfärben (in Eisen-III-Chlorid werden sie braun, in Ammoniumpersulfat matt). Entdecken Sie nach einigen Sekunden noch blank spiegelnde Stellen, so haben Sie nicht genügend entwickelt. An diesen Stellen haftet noch etwas Fotolack, der die Kupferschicht abdeckt.

Nehmen Sie die Platte sofort aus dem Ätzbad, spülen sie ab und entwickeln sie nochmals in Natronlauge, wobei Sie mit dem Pinsel über alle glänzenden Stellen wischen.

Das Ätzen

Sie können mit Eisen-III-Chlorid- oder Ammoniumpersulfatlösungen ätzen. Beide Chemikalien erhalten Sie im Elektronikhandel. **Vor der industriellen Methode, in einem Gemisch aus Salzsäure und Wasserstoffperoxid zu ätzen, sollten Sie sich wegen der Gefährlichkeit des Verfahrens hüten**, auch wenn sie hin und wieder empfohlen wird. Sie ist für den häuslichen Gebrauch viel zu gefährlich.

Das harmloseste Ätzmittel mit zugleich hervorragenden Eigenschaften hinsichtlich Randschärfe und Unterätzungen ist eine gesättigte Eisen-III-Chlorid-Lösung. Dazu lösen Sie so viel Eisen-III-Chlorid in Wasser auf, bis sich nichts mehr davon löst und

sich weiteres auf dem Boden absetzt. Besorgen Sie sich eine 1-l-Kunststoffflasche, geben Sie etwa 400 bis 500 g Eisen-III-Chlorid hinein, und füllen Sie die Flasche mit warmem Wasser. So kann die Lösung nicht ganz falsch sein. Beim Einfüllen des Eisen-III-Chlorids legen Sie sich eine Zeitung auf Ihren Arbeitsplatz und falten sie danach sorgfältig zusammen, ehe Sie sie in den Abfalleimer stecken, sonst werden Sie sich noch lange wundern, woher die vielen braunen Flecken kommen. Ein kleines Körnchen dieses Salzes ist als sehr ergiebiger Farbstoff für viele Überraschungen gut.

Bei Ammoniumpersulfat beträgt das Mischungsverhältnis etwa 350 g mit Wasser aufgefüllt zu 1 l. Im unbenutzten Zustand sieht die Lösung wie klares Wasser aus, sie hinterläßt auch keine braunen Flecken, aber diese Unsichtbarkeit macht sie gewissermaßen „hinterhältig“. Wenn Ihnen z. B. ein kleiner Spritzer auf Ihre schöne Edelstahlspüle gerät, bemerken Sie ihn zuerst gar nicht. Wenn Sie ihn aber bemerken, ist es schon zu spät, denn Sie bemerken nicht ihn, sondern das Loch, das er gefressen hat.

Zum Ätzen legen Sie die entwickelte Platte mit der Kupferseite nach oben in eine flache Kunststoffschale, die nicht wesentlich größer sein sollte als Ihre Platine (Deckel einer Kunststoffverpackung, Gefrierdose o. ä.). Sie gießen die Ätzlösung darüber, bis die Platine etwa 1 cm hoch bedeckt ist.

Wärme und Bewegung beschleunigen den Ätzvorgang. Ammoniumpersulfatlösung muß sogar erwärmt werden. Lassen Sie Ihre Ätzwanne in einer Schüssel mit heißem Wasser schwimmen, und schaukeln Sie die Schale, so daß die Ätzflüssigkeit immer wieder über die Platine spült.

Die Ätzzeit richtet sich nach der Temperatur und Bewegung des Bades, im wesentlichen auch danach, wie frisch die Ätzlösung noch ist, denn je mehr Kupfer darin gelöst ist, desto weniger wirksam wird sie. Die Ätzzeiten können 10 bis 60 Minuten betragen. Achten Sie darauf, daß das Kupfer an allen späteren Isolierstellen der Platine **vollkommen** weggeätzt wird. Zur besseren Beobachtung soll daher der Flüssigkeitsstand über der Platine nicht zu hoch sein.

Nach dem Ätzen spülen Sie die Platine gut ab, erst mit Seifenlauge (sie neutralisiert Säurereste), dann mit viel klarem Wasser.

Dauert das Ätzen länger als eine Stunde, so ist die Ätzlösung verbraucht. Sie erkennen die verbrauchte Lösung auch an der Einfärbung durch das gelöste Kupfer: Ammoniumpersulfatlösung wird blau, Eisen-III-Chloridlösung sehr dunkel braungrün.

Wohin mit der verbrauchten Ätzlösung?

Die verbrauchte Ätzlösung dürfen Sie nur in extremer Verdünnung wegschütten. Erlaubt sind maximal 2 mg Kupfer pro Liter Wasser. 1 mg Kupfer entspricht einer weggeätzten Foliensfläche von ca. 3,2 cm². Rechnen Sie sich einmal aus, was das bedeutet, wenn von den 160 cm² einer Europakarte etwa drei Viertel des Kupfers weggeätzt werden!

Auch wenn Sie sich im Rahmen des Erlaubten bewegen, so tun Sie der Umwelt mit dem einfachen Wegspülen nichts Gutes. Sofern Sie bereit sind, sich den Umweltschutz einige Groschen kosten zu lassen, kaufen Sie sich eine Packung des SENO-GS-System ÄTZEN (Elektronikhandel).

Sie reicht für alle Platinen dieses Mikrocomputers aus und enthält einen Beutel Pulver, mit dem Sie die verbrauchte Ätzlösung umweltfreundlich beseitigen können.

Entschichten

Wischen Sie den Fotolack von der Platine mit einem acetongetränkten Papiertaschentuch ab. Bei manchen Platinen, die Sie beschichtet kaufen können, ist aus Gründen der Produktionsvereinfachung auch die Isolierseite beschichtet. Sollte Ihnen die Platine merkwürdig dunkel vorkommen, so wischen Sie auch diese Seite ab.

Man hört immer wieder, es sei nicht nötig, den Fotolack zu entfernen, denn durch ihn könne man hindurchlöten wie durch Lötack. Es geht auch – manchmal, aber nicht bei allen Laken, und dann auch nur mit einem leistungsstarken LötKolben (50 W) und heißer Spitze (ab 370 °C). Bei der bisweilen zu hörenden Behauptung, der Fotolack wirke zugleich auch wie Lötack als Flußmittel, dürfte wohl der Wunsch der Vater des Gedankens gewesen sein.

Aus der Beobachtung von Schülerarbeiten läßt sich nur der Rat herleiten, sich den kurzen Arbeitsgang des Entschichtens nicht zu ersparen. Übermäßiges Durchheizen der Leiterbahn schädigt den Kleber, mit dem das Kupfer auf der Isolierplatte befestigt ist, und dann lösen sich die Lötäugen sehr schnell ab.

Das Bohren

Zum Bohren der Platinen gibt es im Elektronikhandel Miniaturbohrmaschinen mit hohen Drehzahlen (14000 bis 20000 UpM). Einen Ständer benötigen Sie nicht, denn Sie boh-

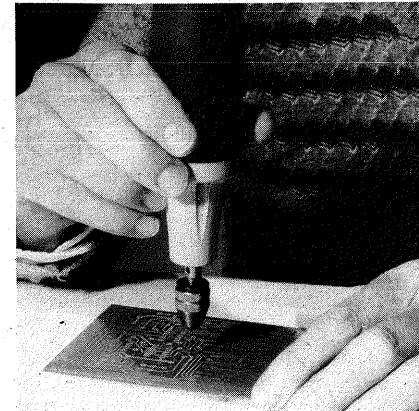


Bild 1.29 Freihändiges Bohren mit der elektrischen Miniaturbohrmaschine; ein Bohrstander ist nur bei Verwendung von Vollhartmetallbohrern erforderlich

ren besser „freihändig“ (Bild 1.29). Investieren Sie das eingesparte Geld lieber in ein möglichst kräftiges Modell, eines mit kugelgelagerter Achse verdient den Vorzug vor anderen. Zu kleine Bohrmaschinen werden schnell heiß, und Sie können von Glück reden, wenn Sie alle Bohrungen einer einzigen Europakarte schaffen, ohne die Arbeit zwecks Abkühlung der Maschine unterbrechen zu müssen. Insgesamt werden Sie für eine Bohrmaschine einschließlich des Transformators für die Stromversorgung allerlei Geld anlegen müssen. Unterschätzen Sie nicht die kleinen Uhrmacher-Drillbohrer (Bild 1.30), die Sie in guten Werkzeuggeschäften, im Uhrmacherbedarfshandel und wenn nicht, dann bei der Firma *Wolf & Schroth, Werkzeugversand, 2000 Hamburg 1, Postfach 10 67 40, Tel. 040/2360150*, bekommen. Sie sind wesentlich billiger (ca. 9,75 DM für das Werkzeug, ca. 5,50 DM für eine Packung mit drei gleichen Eureka-

Bohrern, jeweils zuzüglich MWSt, Preise von 1983) und haben den Vorteil, daß Sie die Bohrer auf einem Abziehstein nachschärfen können.

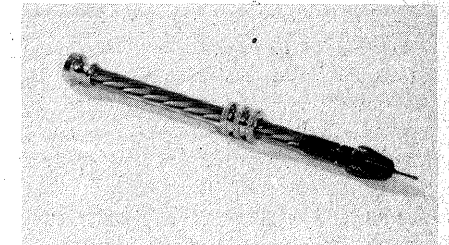


Bild 1.30 Uhrmacher-Drillbohrer

Bild 1.31 zeigt die Handhabung. Sie spannen die Platine auf eine glatte Unterlage, z. B. eine Spanplatte; dazu reicht die Klemme Ihres Laubsäge-tischchens. Kleben Sie sich auf die Innenseite der Klemme einen Streifen dünnes (Handschuh-)Leder, dann

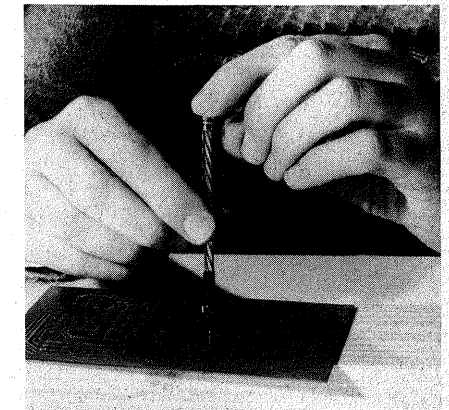


Bild 1.31 Handhabung des Uhrmacher-Drillbohrers

gibt es keine Kratzer. Mit der linken Hand halten Sie den Bohrer und regulieren den Druck. Sie brauchen nicht stark zu drücken. Sie sollten es auch nicht tun, denn sonst verkrampfen Sie sich nur. Wenn ein **neuer** Bohrer einen starken Grat aufwirft, ist der Druck zu stark. Mit der rechten Hand schieben Sie die Rändelmutter gleichmäßig auf und nieder.

Zum Nachschleifen fassen Sie den Bohrer wie einen Bleistift und reiben die Schneide auf dem nassen Abziehstein, als wollten Sie schreiben (Bild 1.32).

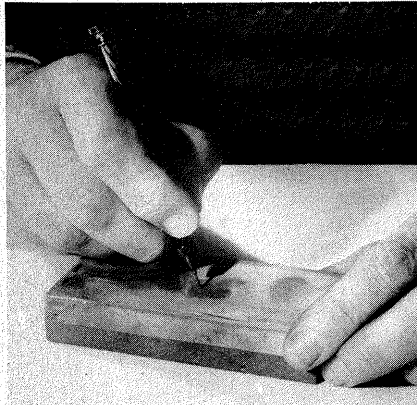


Bild 1.32 Schleifen des Bohreinsatzes

Die Spiralbohrer für die elektrischen Bohrmaschinen müssen mindestens die Qualität HSS haben. Beim Bohren von Pertinaxplatinen halten sie lange; in Epoxydplatten werden sie schnell stumpf. Ein Bohrer reicht dann durchschnittlich für eine Europakarte. Das Stumpfwerden erkennen Sie daran, daß sich die Kupferbahn am Rand der Bohrung aufwirft. Dadurch wird nicht nur das Lötauge geschädigt, die Bauelemente sind auch schwerer anzulöten.

Geschickte Feinmechaniker schleifen sich auch die dünnen Spiralbohrer wieder an. Zu den kleinen Bohrmaschinen gibt es auch spezielle Schleifsteine, die Sie dazu verwenden können. Wenn die Bastelkasse es aber zuläßt, sollten Sie sich überlegen, ob Sie sich nicht lieber Vollhartmetall-Spiralbohrer leisten. Sie sind freilich sehr teuer und kosten ca. 8 DM pro Stück (1983).

Gebohrt wird grundsätzlich von der Kupferseite her. Die Platine muß fest auf einer glatten Unterlage aufliegen, damit sie nicht auf der Isolierseite ausreißt.

Allgemein können Sie mit 1 mm \varnothing bohren, nur für die kleinen Lötungen der Speicherplatte benötigen Sie 0,8-mm-Bohrer. Die Löcher für die Trimmerwiderstände sowie die Löt-nägel bohren Sie mit 1,3 mm \varnothing auf. Können Sie die Bohrstellen nicht an; die Kupferfolie staucht sich auf und löst sich von der Platte. Da in die Löt-augen ein kleiner Punkt eingätzt ist, haben Sie bereits genaue Ansatzpunkte für den Bohrer.

Die Lötungen, auf die die Litzen gelötet werden, bohren Sie nicht auf. Sie erkennen sie am Fehlen der Ätzstellen.

Schutzmaßnahmen bei der Platinenherstellung

Die Chemikalien des Fotolacks, Entwicklers, Ätzmittels und schließlich das Aceton sind alles andere als hautfreundlich. Arbeiten Sie daher mit Einmal-Schutzhandschuhen aus Polyäthylenfolie, die Sie für wenige Groschen in der Apotheke erhalten.

Haben Sie sich die Haut mit dem Entwickler (Natronlauge) benetzt, so spülen Sie sie **sofort** mit viel klarem

Wasser ab. Bei Spritzern, die in die Augen geraten sein sollten, reicht das nicht – suchen Sie **sofort** einen Augenarzt auf!

Ätzmittel waschen Sie sofort mit milder Seife ab; auch aus der Kleidung sollten Sie Ätzmittel sofort auswaschen, weil sie den Stoff bleichen können. Die braunen Flecken des Eisen-III-Chlorids können Sie mit Rostfleckentferner beseitigen.

Sollte Ihnen Lötzinn auf einen glatten Fußboden (z. B. PVC-Belag) getropft sein, so ist das nicht weiter schlimm. Sie können den Zinnlecks mit einem **stumpfen** Messer leicht abheben. Schlimmer ist es schon, wenn Ihnen das Zinn auf eine teppichartige Auslegeware tropft. Die Zinnkügelchen sind manchmal nur schwer herauszuzupfen, und das geht nicht immer ohne ein kleines Loch ab. Dasselbe kann geschehen, wenn Ihnen der Löt-kolben einmal herunterfällt. Kleine Löcher in der Auslegeware können Sie fast unsichtbar dadurch stopfen, daß Sie die verbrannten Fasern auszupfen, sich an einer wenig sichtbaren Stelle einige Ersatzfasern sammeln (z. B. in einer Zimmerecke), das Loch dünn mit Pattex bestreichen und die Ersatzfasern einkleben.

Kapitel 2

Grundlagen der Digitaltechnik

In dem folgenden Kapitel werden die Grundlagen der Digitaltechnik nur so weit dargestellt, wie sie zum Nachbau des Mikrocomputers unbedingt erforderlich sind.

Zugleich werden zusammen mit den logischen Grundschaltungen die entsprechenden ICs aus der TTL-Serie 74XX vorgestellt (TTL = Transistor-Transistor-Logik), weil die den Mikroprozessor umgebenden Bausteine ebenfalls aus der TTL-Familie stammen. Die ICs, deren „Inhalt“ über die Grundschaltungen hinausgeht, werden zusammen mit den Baustufen des Mikrocomputers beschrieben, in denen sie vorkommen.

Analogtechnik – Digitaltechnik

Die Elektronik besteht u. a. aus zwei großen Bereichen, der **Analogtechnik** und der **Digitaltechnik**. „Analog“ kommt aus dem Griechischen und heißt „entsprechend“, „gleichartig“. „Digital“ kommt vom lateinischen Wort „digitus“ = „Finger“ und bezieht sich auf das Rechnen mit ganzen Zahlen. Die Werte „0“, „1“, „2“, „3“, „4“ und „5“ lassen sich z. B. mit

den Fingern einer Hand leicht darstellen, nicht aber die vielen Werte dazwischen, z. B. $2\frac{1}{2}$. Der Blick auf die Finger deutet die Arbeitsweise an, nach der sich Werte irgendwelcher Art immer nur in bestimmten **Sprüngen** ändern können.

Den Unterschied zwischen „analog“ und „digital“ zeigen ältere Bahnhofsuhr sehr schön: Ihr Sekundenzeiger dreht sich ganz gleichmäßig, dem gleichmäßigen Fluß der Zeit entsprechend (eben analog).

Ihr Minutenzeiger steht unterdessen still. Wenn aber der Sekundenzeiger eine Runde vollendet hat, springt er auf den nächsten Minutenwert. Er kann nur ganzzahlige Minutenwerte anzeigen, so wie auch Finger nur ganze Zahlen anzeigen; er arbeitet also digital.

Ähnliches begegnet uns überall, auch dort, wo wir gar nicht auf die Idee kämen, uns tiefeschürfende Gedanken über Analog- oder Digitaltechnik zu machen. Ein Klavierspieler kann Tonhöhenunterschiede nur sprunghaft erzeugen, minimal in Halbtönen – zumindest dann, wenn er sich darauf beschränkt, die Tasten zu benutzen. Die Klaviatur läßt ihn nur „digital“ spielen (man muß es nicht ganz so wörtlich nehmen; auch ein Orgelpedal ist nur „digital“ spielbar).

Sobald der Klavierspieler aber zum Stimmenschlüssel greift und die Saiten des Instruments mehr oder weniger straff spannt, ändert sich die Tonhöhe der Saite genau der Saitenspannung entsprechend – analog.

Ähnliches finden wir beim Tachometer im Kfz, dessen Zeiger der Geschwindigkeit analog mehr oder weniger weit ausschlägt, wohingegen der Kilometerzähler immer ganzzahlig weiterspringt (digital), das 100-m-Rädchen aber analog zur gefahrenen Strecke gleichmäßig mitläuft.

Die Sinusspannung, die ein Mikrofon bei der Aufnahme eines Tons erzeugt, ist eine analoge Spannungsänderung. Der Verstärker, der diese Spannung bis zur Wiedergabe im Lautsprecher verstärkt, muß **analog** arbeiten, denn er soll alle Spannungsänderungen, die an seinem Eingang entstehen, möglichst getreu an seinem Ausgang abbilden.

Wenn man Batteriezellen in Reihe schaltet, sind Spannungsänderungen nur in Sprüngen einer Zellenspannung möglich (eine Zelle 1,5 V, zwei Zellen 3 V, drei Zellen 4,5 V usw.).

Die **Digitalelektronik** schränkt, von Sonderfällen abgesehen, die Spannungssprünge sogar noch gröber ein, nämlich auf die Spannungspegel (nahe) 0 V, abgekürzt L (Low, engl. niedrig) und nahe der vollen Betriebsspannung, abgekürzt H (High, engl. hoch). Zwischenwerte werden übersprungen (Bild 2.1).

Die Schaltelemente der Digitaltechnik arbeiten grundsätzlich **binär** (bi, lat. zwei), d. h. zweiwertig mit den Spannungspegeln L oder H.

Während es technisch sehr schwierig werden kann, eine Spannungsänderung analog zu behandeln, ist es im Vergleich dazu sehr leicht, einen Transistor ein- oder auszuschalten,

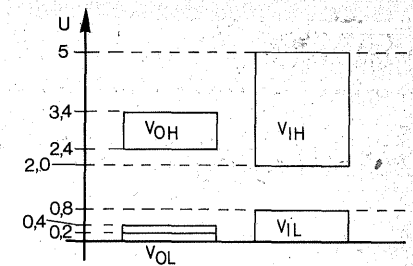


Bild 2.1 H- und L-Pegel der TTL-Bausteine; links: Ausgangsspannungsbereich; rechts: Eingangsspannungsbereich.

V_{OH} : H-Spannung des Ausgangs;
 V_{OL} : L-Spannung des Ausgangs;
 V_{IH} : H-Spannung des Eingangs;
 V_{IL} : L-Spannung des Eingangs

denn um die vielen möglichen Zwischenwerte braucht man sich ja nicht zu kümmern. Dies ist einer der Gründe, warum sich die Digitaltechnik zunehmend durchsetzt.

Bedingung ist freilich, daß man alle Informationen in so kleine Bestandteile untergliedert, daß sie sich in binären Signalen ausdrücken lassen. Dann ist auch die Verknüpfung nach den Regeln der zweiwertigen Aussagenlogik leicht zu realisieren. Die technischen Verknüpfungsglieder sind die „logischen Schaltungen“ oder „Gatter“. Sie sind die Bausteine der Digitaltechnik, von der die Computerei ein großer Bereich ist.

Logische Verknüpfungen

Logik (von griech. *logiké téchne*) ist die Kunst (*téchne*, Technik) des Denkens, also eine Theorie von den **formalen** Beziehungen zwischen Denkinhalten. Die **einfachste** Theorie ist

die **zweiwertige Aussagenlogik**; „zweiwertig“ heißt: Eine Aussage ist entweder „wahr“ oder „falsch“ („nicht wahr“), etwas Drittes gibt es nicht. Ein „Jain“, ein „Ja – aber“, das uns im täglichen Leben so oft aus der Klemme hilft, hat in der zweiwertigen Logik keinen Platz – die berühmten „Halbwahrheiten“ schon gar nicht. Ohne sie mag es vielleicht im Leben oft nicht gehen, mit ihnen geht aber in der Computerei nichts.

Warum? Eine Aussage im Sinn der zweiwertigen Logik ist ein Satz (ein Ausdruck, eine Zeichengruppe), von dem **in einem speziellen Fall** entschieden werden kann, ob er wahr oder falsch ist. Wenn eine eindeutige Entscheidung nicht möglich ist, paßt sie nicht ins System und kann nicht verarbeitet werden. Daraus mag man schon erahnen, auf welchem eingegrenztem Bereich die zweiwertige Logik anwendbar ist.

In logisch einfachen (nicht zusammengesetzten) Aussagen wird einem Subjekt ein Prädikat zugeordnet, z. B. „Der Motor läuft.“ Diese Aussage kann auch negiert („bestritten“) werden: „Der Motor läuft nicht“, oder: „Es ist nicht wahr, daß der Motor läuft.“

Neue Aussagen gewinnt man dadurch, daß man zwei oder mehr einfache Aussagen zusammensetzt und dann nach den Regeln der Kunst entscheidet, ob die so neu gewonnene Aussage wahr oder falsch ist. Man kann auch mehrere zusammengesetzte Aussagen wieder miteinander verknüpfen, und so kann es weitergehen, bis auch sehr umfangreiche Probleme entschieden sind.

Voraussetzung ist aber immer, daß jedes Problem in so kleine Aussagen zerlegt wird, daß bei jeder einzelnen zu entscheiden ist, ob sie wahr oder

falsch ist. Das ist der Berührungspunkt zwischen der zweiwertigen Aussagenlogik und der Digitaltechnik.

Was sich in der allgemeinen Theorie so wenig greifbar liest, ist in der Praxis glücklicherweise bei weitem nicht so schwer. Für eine Aussage steht eine Leitung, ein Stück Draht, ein bestimmter Punkt in einer Schaltung usw. Im obigen Beispiel kann es z. B. die Leitung sein, die den Motor mit der Betriebsspannung (gegen Masse) versorgt.

Aussagen kürzt man mit Großbuchstaben ab. Die Aussage „Der Motor läuft“ soll z. B. das Kürzel „A“ erhalten.

Für „wahr“ schreibt man „(logisch) 1“, für „falsch“ schreibt man „(logisch) 0“.

Wenn die Aussage „Der Motor läuft“ wahr ist ($A=1$), so steht auf der Leitung A die (fast) volle Betriebsspannung.

Wenn die Aussage „Der Motor läuft“ falsch ist ($A=0$), so steht auf der Leitung A keine Betriebsspannung.

Man schreibt „1“ oder „0“, wenn man den Sachverhalt von der logischen Seite betrachtet. Die **Spannungspegel**, die den logischen Zuständen entsprechen, drückt man durch H (High, engl. hoch) oder L (Low, engl. niedrig) aus.

H entspricht einer 1,

L entspricht einer 0.

Und was ist nun, wenn auf der abgefragten Leitung nur ein bißchen Spannung steht, vielleicht die halbe Betriebsspannung, gerade so viel, daß der Motor noch nicht anläuft? Das ist gerade die „Halbwahrheit“, die aus dem logischen System ausgeklammert ist. In der technischen Realisation wird dieser Fall durch geeignete

Maßnahmen verhindert. Dabei wird stillschweigend vorausgesetzt, daß kein Schaltungsteil technisch defekt ist.

Beispiel: Die Logikschaltungen, mit denen Sie Ihren Mikrocomputer bauen, werden mit einer Betriebsspannung von +5 V (gegen Masse) gespeist. Da nichts auf dieser Welt absoluten Idealen entspricht, sind für die Werte L oder H Spannungsbereiche vorgesehen:

H (logisch 1) umfaßt in der TTL-Technik den Bereich von 5 V bis herab zum absoluten Minimum von 2 V. L (logisch 0) umfaßt den Bereich 0 V bis zum absoluten Maximum von 0,8 V.

Dies sind die beiden Bereiche, die ein **TTL-Eingang** noch als H bzw. L erkennt.

TTL-Ausgänge liefern im Normalfall eine H-Spannung von 3,4 V, die bei Belastung durch Stromentnahme bis auf minimal 2,4 V absinken darf; die L-Ausgangsspannung beträgt normal 0,2 V, sie darf aber durch Belastung auf maximal 0,4 V ansteigen. Zwischen den Grenzen der Eingangs- und Ausgangsbereiche liegt jeweils ein „Störabstand“ von 0,4 V.

Der Bereich zwischen L und H (0,8 V bis 2,0 V) ist undefiniert und wird übersprungen (Bild 2.1).

Jetzt fehlen nur noch die „Denkregeln“, nach denen die Aussagen bzw. die Spannungspegel miteinander verknüpft werden. Auch da ist der Aufwand gering; man kommt nämlich mit nur drei Verknüpfungsregeln aus, mit UND, ODER, NICHT.

Wohlgemerkt: Das sind die Grundverknüpfungen. Mit ihnen sind aber auch die kompliziertesten Schaltungen zu realisieren. Vielfach sind schon zahlreiche Verknüpfungsglieder zu neuen Bausteinen zusammen-

gefaßt, so daß man meinen könnte, es kämen noch neue Prinzipien hinzu – dem ist aber nicht so. Die Anzahl der Verknüpfungsmöglichkeiten ist Legion, und daraus ergibt sich die ungeheure Vielfalt, die uns die Digitaltechnik bietet.

Die Schaltglieder, die die Logik verkörpern, heißen **Gatter** (Tor, nach dem engl. gate). Dieser Begriff bezeichnet das Verhalten sehr anschaulich. Man kann ein Gatter mit der Schranke an einem Grenzübergang vergleichen; diese geht nur hoch (erreicht den Zustand H), wenn bestimmte Bedingungen erfüllt sind. Die Symbole für die Gatter sind nicht überall gleich; daher werden hier die weit verbreiteten amerikanischen, die alten deutschen und die jetzt gültigen deutschen mitgeteilt.

In den Logikplänen werden normalerweise nur die Symbole der Gatter oder der daraus zusammengesetzten Schaltglieder gezeichnet. Selbstverständlich müssen alle Gatter mit Energie versorgt werden. Die Leitungen für die Spannungsversorgung werden in der Regel nicht mitgezeichnet, weil nur der Logikplan interessiert.

Die UND-Verknüpfung (Konjunktion)

Das Ergebnis mehrerer mit UND verknüpften Aussagen ist formal genau dann wahr, wenn **sämtliche** miteinander verknüpften Einzelaussagen wahr sind. Ist auch nur eine Einzelaussage falsch, so ist auch das Ergebnis der Verknüpfung falsch. Beispiel: Der Kassettenrecorder spielt (Q), wenn eine Kassette eingelegt ist (A) UND wenn die Wiedergabetaste gedrückt ist (B). A und B sind die Einzelaussagen, die mit UND verknüpft

sind, und von denen abhängt, ob das Ergebnis der Verknüpfung Q wahr oder falsch ist.

Die UND-Verknüpfung entspricht der Schnittmenge: Ein Element gehört genau dann zur Schnittmenge, wenn es Element aller miteinander geschnittenen Mengen ist (Bild 2.2).

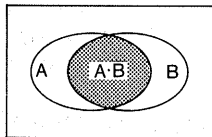


Bild 2.2 Venn-Diagramm der Schnittmenge. Ihr entspricht die UND-Verknüpfung

UND (engl. AND) wird meistens mit Großbuchstaben geschrieben, um das logische Verständnis des Wortes anzudeuten. Umgangssprachlich wird „und“ auch anders gebraucht, wie z.B. Bild 2.3 zeigt. Wenn man das „und“ auf dem Schild der Baustellenampel „logisch“ versteht, muß der Autofahrer nur bei „gelb/rot“ halten („gelb“ UND „rot“), und er darf so-



Bild 2.3 Schild an einer Baustellenampel. Es ist für pedantische Logiker lebensgefährlich

wohl bei „rot“ als auch bei „grün“ als auch bei „gelb“ fahren.

In Formeln wird die Konjunktion nach Heyting (niederländischer Mathematiker, *1898) mit dem Zeichen „ \wedge “ ausgedrückt; das ist ein umgekehrtes „v“ (von lat. vel = oder), denn die UND-Verknüpfung ist das genaue Gegenteil der ODER-Verknüpfung. In der englischsprachigen Literatur wird die Konjunktion oft mit dem Multiplikationspunkt (nach George Boole, 1815–1864, Begründer der mathematischen Logik) oder dem Zeichen „&“ (nach David Hilbert, deutscher Mathematiker, 1862–1943) ausgedrückt. Das obige Beispiel wird also als Formel notiert:

$$Q = A \wedge B \text{ bzw. } Q = A \cdot B \\ \text{bzw. } Q = A \& B$$

Die Kombinationsmöglichkeiten solcher Formeln werden in sogenannten **Wahrheitstafeln** oder **Funktionstafeln** dargestellt:

A	B	$Q = A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

In Datenbüchern werden oft auch die Spannungspegel L und H angegeben, um das Verhalten des Bausteins darzustellen. Dadurch ergibt sich im Prinzip die gleiche Tabelle:

A	B	$Q = A \cdot B$
L	L	L
L	H	L
H	L	L
H	H	H

Man kann die Wahrheitstafeln auswendig lernen; besser und leichter ist es, sich die allgemeine Schaltbedingung zu merken, etwa: **UND hat am Ausgang nur dann H, wenn alle Eingänge H sind.** Oder: **Schon ein einziges L an einem Eingang zwingt den Ausgang von UND auf L.**

Bild 2.4 zeigt die gebräuchlichen Symbole für das UND-Gatter, dazu die Anschlußbelegung des 7408, welcher vier UND-Gatter mit je zwei Eingängen enthält.

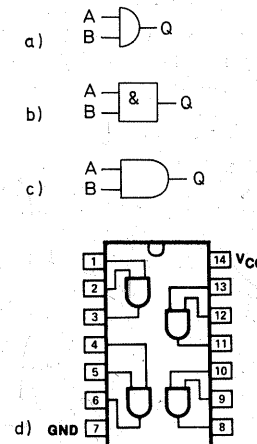


Bild 2.4 UND-Gatter: Symbole a) deutsch (alt), b) deutsch (neu), c) amerikanisch; d) Anschlußbelegung des Bausteins 7408 (nach VALVO-Unterlagen)

Die Leitungen A und B sind die Eingänge, Q ist der Ausgang des Gatters. Die Anzahl der Eingänge kann beliebig erweitert werden, dann ist es möglich, entsprechend viele Eingangsvariablen miteinander zu verknüpfen. Es gibt dann auch entsprechend mehr mögliche Fälle; bei drei Eingangsvariablen sind es bereits $2^3 = 8$, bei vier sind es $2^4 = 16$, bei fünf sind es $2^5 = 32$ usw. Es gibt aber immer nur einen

einzigsten Fall, in dem die Ausgangsvariable logisch 1 ist, wenn nämlich sämtliche Eingangsvariablen logisch 1 sind.

Die ODER-Verknüpfung (Disjunktion)

Die Disjunktion ist das genaue Gegenteil der Konjunktion. Es genügt, daß mindestens eine Eingangsvariable wahr ist, damit die Ausgangsvariable wahr ist. Es können auch mehrere oder alle Eingangsvariablen wahr sein, um zu dem gleichen Ergebnis zu führen.

Es gibt nur einen einzigen Fall, in dem die Ausgangsvariable falsch ist, nämlich den, daß **sämtliche** Eingangsvariablen falsch sind.

Die ODER-Verknüpfung entspricht der Vereinigungsmenge. Ein Element gehört dann zur Vereinigungsmenge, wenn es Element wenigstens einer der vereinigten Mengen ist. Die Schnittmenge ist also eingeschlossen (Bild 2.5).

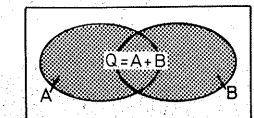


Bild 2.5 Venn-Diagramm der Vereinigungsmenge. Ihr entspricht die ODER-Verknüpfung

Das logische ODER unterscheidet sich von unserem normalen Sprachgebrauch. In der Umgangssprache benutzen wir das Wort „oder“ meist ausschließend; wenn z.B. ein Händler sagt: „Sie können sich die Ware abholen, **oder** ich schicke sie Ihnen mit der Post“, so schließt die eine Möglichkeit die andere aus. **Entweder** holt sich der Kunde die Ware ab (und

bekommt sie nicht per Post), oder er (holt sie sich nicht ab und) erhält sie per Post. Das wäre das **exklusive ODER (EXOR)**, das in der Digitaltechnik ebenfalls eine Rolle spielt (siehe Seite 52); doch das Beispiel zeigt, daß es bereits eine Verbindung mit UND, ODER und NICHT ist.

Das **logische ODER (OR)** ist das **ein-schließende (inklusive) ODER (lat. vel)**. Am besten macht man sich das an einer alltäglichen Situation klar: In einem mehrgeschossigen Haus gehören zu jeder Wohnungsglocke (Y) zwei Taster, einer am Hauseingang (A) und einer an der Wohnungstür (B). Die Glocke läutet (Y), wenn der Taster am Hauseingang (A) ODER der Taster an der Wohnungstür (B), ODER wenn beide zugleich gedrückt werden. Nur wenn weder A noch B betätigt werden, läutet die Glocke nicht.

Der „logische“ Gebrauch des „oder“ beginnt indessen seinen Einzug in unsere Umgangssprache. Bild 2.6 zeigt ein neueres Schild an einer Baustelle-



Bild 2.6 Schild an einer Baustellenampel – auch für Logiker korrekt, sofern sie sich nicht für Grammatik interessieren

lenampel. Die Verkehrsteilnehmer sollen nun bei „rot“ ODER „gelb“ halten, und das heißt sowohl bei „rot“ als auch bei „gelb“, einschließlich des Falles „rot/gelb“.

Die Disjunktion wird durch ein „v“, in der englischsprachigen Literatur gemäß der Booleschen Algebra mit dem Zeichen „+“ ausgedrückt. Unser Beispiel heißt in der Formelsprache:

$$Y = A \vee B \text{ bzw. } Y = A + B.$$

In der Digitaltechnik, die die direkte Umsetzung der Booleschen Algebra ist, setzen sich deren Formelzeichen zunehmend durch und finden sich auch in Datenbüchern. Daher werden sie auch in diesem Buch benutzt.

Die Wahrheitstafel zeigt die ODER-Verknüpfung in ihren Einzelfunktionen:

A	B	$Y = A + B$
0	0	0
0	1	1
1	0	1
1	1	1

Ihr entspricht die Funktionstabelle für die Spannungspegel:

A	B	$Y = A + B$
L	L	L
L	H	H
H	L	H
H	H	H

Merksatz: Schon ein einziges H an einem Eingang zwingt den Ausgang auf H.

Bild 2.7 zeigt die Schaltzeichen für

ODER (OR), dazu die Anschlußbelegung des 7432, der vier ODER-Gatter mit je zwei Eingängen enthält.

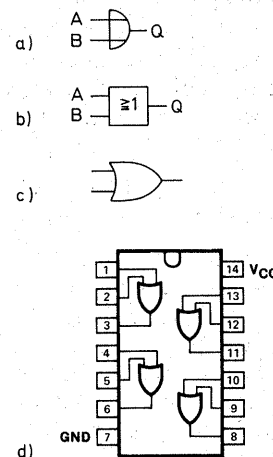


Bild 2.7 ODER-Gatter: Symbole a) deutsch (alt), b) deutsch (neu), c) amerikanisch; d) Anschlußbelegung des Bausteins 7432 (nach VALVO-Unterlagen)

Die Negation (NICHT, NOT)

Die Negation kehrt eine Aussage von wahr nach falsch oder von falsch nach wahr um, genau so, wie es in der Umgangssprache üblich ist. Sie wird in Formeln durch ein vorangestelltes Minuszeichen („-“, nach Giuseppe Peano, italienischer Mathematiker, 1858–1932) oder einen vorangestellten Haken („¬“, nach Heyting) ausgedrückt. In der Technik hat sich die Überstreichung (nach Hilbert) durchgesetzt.

Beispiel: Ich komme (Q), wenn es NICHT regnet (\bar{A}).

$$Q = \neg A \text{ bzw. } \bar{A} \text{ (gelesen „A nicht“ bzw. „A quer“).}$$

Die Lesart „A quer“ ist weit verbreitet, weil sie sich leichter der grammatischen Konstruktion der Umgangssprache anpaßt. Die Negation entspricht der Komplementmenge (Bild 2.8).

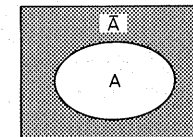


Bild 2.8 Komplementmenge. Ihr entspricht die Negation

In der zweiwertigen Logik ist eine **doppelte Negation** wieder eine Bejahung. So ist es auch in der Umgangssprache; wenn wir z. B. sagen: „Er ist kein schlechter Kerl“, meinen wir: „Er ist ein guter Kerl.“ Eine gerade Anzahl von nacheinander ausgeführten Negationen ergibt wieder die ursprüngliche Aussage; eine ungerade Anzahl von Negationen kehrt die Aussage um.

So plausibel das anfangs erscheinen mag, so viele Probleme kann man in der Praxis mit den Negationen haben, denn nur zu leicht übersieht man eine Signalumkehrung durch ein Schaltglied, und schon läuft nichts mehr.

Bild 2.9 zeigt die Schaltzeichen für das Negationsglied, meist **Inverter** (lat. Umkehrer) genannt, dazu die Anschlußbelegungen der ICs 7404 und 7405. Sie gleichen sich in der Anschlußbelegung, unterscheiden sich aber dadurch, daß der 7404 die normale TTL-Ausgangsstufe enthält (Bild 2.29 a), der 7405 einen Open-Collector-Ausgang (Bild 2.29 b). Die gleiche Anschlußbelegung hat der 7414, dessen Eingänge jedoch mit

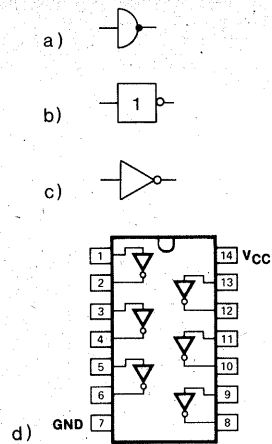


Bild 2.9 Inverter: Symbole
a) deutsch (alt), b) deutsch (neu),
c) amerikanisch; d) Anschlußbelegung der
Bausteine 7404, 7405, 7414, 7416 und 7417
(nach VALVO-Unterlagen)

Schmitt-Trigger ausgestattet (siehe auch Seite 53) sind. Diese Funktion benötigen wir z.B. im Taktoszillator unseres Mikrocomputers.

Das eigentliche Invertierungszeichen im Logikplan ist der Punkt bzw. der kleine Kreis. Er braucht nicht nur am Ausgang eines Schaltgliedes zu stehen. Man kann ihn auch am Eingang eines Gatters oder eines anderen

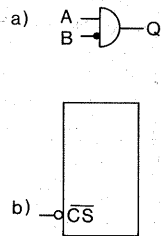


Bild 2.10 Negation an Eingängen (siehe auch \overline{MR} in Bild 2.28)

Bausteines finden (Bild 2.10). Da bedeutet er, daß ein Signal erst invertiert werden muß, ehe es weiterverknüpft wird. Der logische Ausdruck für die Schaltung aus Bild 2.10 heißt $Q = A \cdot \overline{B}$.

Bei komplexeren Bausteinen bedeutet der Punkt auch, daß der ganze Baustein durch L an dem betreffenden Steuereingang aktiviert wird (Bild 2.10b).

Das Dreieck des amerikanischen Invertersymbols deutet zugleich die Verstärkerfunktion an. Oft werden Inverter als Verstärker benutzt (siehe Seite 101). Viele Signalquellen geben zwar die den Pegeln entsprechenden Spannungen ab, können aber nicht die erforderlichen Ströme für die folgenden Gatter oder Verbraucher (z. B. Anzeigeeinheiten) schalten. Daher werden Inverter, die z. T. erhebliche Ströme liefern können, als **Puffer** (engl. buffer) zwischengeschaltet.

Gatterkombinationen

Ihre größte Variabilität erhalten Gatter durch die Verbindung mit der Negation. Die Negation ergibt sich oft schon durch die Notwendigkeit, ein Gatter mit einem Verstärker auszustatten. So wird der Inverter automatisch mitgeliefert und erzeugt Universalgatter.

Das NAND-Gatter

Die Verbindung eines UND-Gatters mit einem **nachfolgenden** Inverter ergibt das NAND-Gatter (aus Not-AND). Die Wahrheitstafel zeigt die genaue Umkehrung des UND-Gatters; die logische Funktion des NAND-Gatters ist $Q = \overline{A \cdot B}$:

A	B	$Q = \overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

Als Funktionstafel:

A	B	$Q = \overline{A \cdot B}$
L	L	H
L	H	H
H	L	H
H	H	L

Merksatz: Schon ein einziges L an einem der Eingänge zwingt den Ausgang eines NAND-Gatters auf H. Es gibt nur einen einzigen Fall, in dem ein NAND-Gatter auf L schaltet, wenn nämlich sämtliche Eingänge auf H sind.

Bild 2.11 zeigt die Schaltzeichen für NAND sowie die Anschlußbelegung des TTL-Bausteins 7400; er enthält vier NAND-Gatter mit je zwei Eingängen und ist **der** Universalbaustein, mit dem man praktisch alles machen kann. Der 7403 hat die gleiche Anschlußbelegung. Er unterscheidet sich vom 7400 dadurch, daß er Open-Collector-Ausgänge besitzt (siehe auch Bild 2.29b). Identisch ist die Anschlußbelegung des 74132, die vier NAND-Gatter haben, aber Schmitt-Trigger-Eingänge (siehe Seite 53). Am logischen Verhalten ändern diese Varianten nichts, man wird aber je nach den technischen Rahmenbedingungen das am besten geeignete IC auswählen.

Es gibt auch Bausteine mit NAND-Gattern, die mehr Eingänge haben,

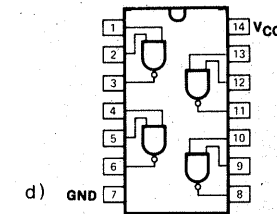
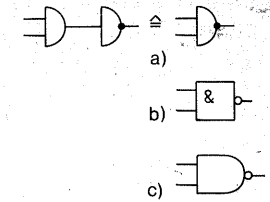


Bild 2.11 NAND-Gatter: Symbole
a) deutsch (alt), b) deutsch (neu),
c) amerikanisch; d) Anschlußbelegung der
Bausteine 7400, 7426 (o.c.) und 74132 (ST)
(nach VALVO-Unterlagen)

z. B. 7410 (3 NAND mit je drei Eingängen) oder 7420 (2 NAND mit je vier Eingängen) oder 7430 (1 NAND mit acht Eingängen).

Das NOR-Gatter

Das NOR-Gatter (aus Not-OR) ist die Verbindung eines ODER-Gatters mit einem nachfolgenden Inverter (Bild 2.12). Sein Ausgang ist nur dann H, wenn alle Eingänge L sind; die logische Funktion ist $Q = \overline{A + B}$:

A	B	$Q = \overline{A + B}$
0	0	1
0	1	0
1	0	0
1	1	0

Die technische Funktion ist:

A	B	$Q = \overline{A+B}$
L	L	H
L	H	L
H	L	L
H	H	L

Merksatz: Bereits ein einziges H an einem der Eingänge zwingt den Ausgang des NOR-Gatters auf L.

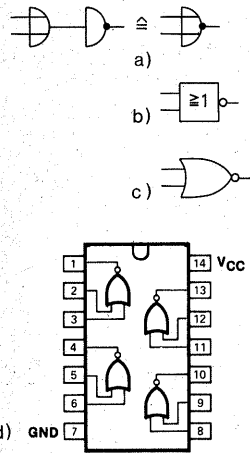


Bild 2.12 NOR-Gatter: Symbole a) deutsch (alt), b) deutsch (neu), c) amerikanisch; d) Anschlußbelegung des Bausteins 7402 (nach VALVO-Unterlagen)

Der Baustein 7402 (Bild 2.12d) ist ebenso universell einsetzbar wie der 7400. Auch für NOR-Gatter gibt es Bausteine mit mehr als zwei Gattereingängen, z. B. 7427 (3 NOR mit je drei Eingängen) oder 7425 (2 NOR mit je vier Eingängen).

Inverter aus NAND- oder NOR-Gattern

Man kann NAND- oder NOR-Gatter als Inverter benutzen; dazu sind lediglich die Eingänge parallel zu schalten (Bild 2.13). Das mag zunächst wie Verschwendung aussehen.

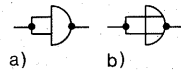


Bild 2.13 NICHT aus NAND (a) oder NOR (b) durch Zusammenschalten der Eingänge

Oft hat man aber in den Bausteinen mit mehreren NAND- oder NOR-Gattern noch ein Gatter frei, das kann man dann als Inverter benutzen. Täte man das nicht, müßte man ein zusätzliches IC mit Invertern investieren, und dann wäre der Aufwand an Material und Strom erheblich größer.

UND aus NAND, ODER aus NOR

Eine doppelte Negation stellt den ursprünglichen Wahrheitswert ja wieder her:

$$\overline{\overline{A}} = A$$

Daher ist es möglich, aus zwei NAND-Gattern wieder ein UND-Gatter zu gewinnen, indem man den NAND-Ausgang wieder invertiert (Bild 2.14a).

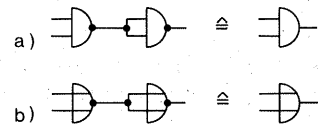


Bild 2.14 a) UND-Gatter aus zwei NAND-Gattern, b) ODER-Gatter aus zwei NOR-Gattern

Auf gleiche Weise läßt sich aus zwei NOR-Gattern wieder ein ODER-Gatter gewinnen (Bild 2.14b).

NAND und NOR als Universalgatter

Es kommt aber noch besser: Aus nur NAND- oder nur NOR-Gattern läßt sich gemäß dem „de Morganschen Gesetz“ (Augustus de Morgan, britischer Mathematiker, 1806–1871) jede beliebige Verknüpfung herstellen.

Im Klartext ausgedrückt, bedeutet das „de Morgansche Gesetz“, daß sich bei der Negation der einzelnen Aussagen die Junktoren UND bzw. ODER vertauschen. Dieser Zusammenhang wird an einem Beispiel aus der Umgangssprache leicht plausibel: Wir greifen auf das ODER-Beispiel von Seite 46 zurück: Die Haustürglocke läutet (Y), wenn der Taster am Hauseingang gedrückt wird (A) ODER wenn der Taster an der Wohnungstür gedrückt wird (B):

$$Y = A + B$$

Nun betrachten wir die Negation der Einzelaussagen:

Die Haustürglocke läutet NICHT (\overline{Y}), wenn der Taster am Hauseingang NICHT gedrückt wird (\overline{A}), UND wenn der Taster an der Wohnungstür NICHT gedrückt wird (\overline{B})

$$\overline{Y} = \overline{A} \cdot \overline{B}$$

Nun drehen wir das Ganze nochmals um:

$$\overline{\overline{Y}} = \overline{\overline{A} \cdot \overline{B}}$$

Da nach der Regel der doppelten Negation $\overline{\overline{Y}} = Y$ ist, erhalten wir also:

$$Y = A + B = \overline{\overline{A} \cdot \overline{B}}$$

Das Umformungsergebnis, das bei der Negation des ODER entstanden

ist, sieht nach der Funktion des NAND aus; sie ist es auch, nur mit dem Unterschied, daß die Eingänge negiert sind. Aus einem NAND wird ODER, wenn man die NAND-Eingänge invertiert.

Die ODER-Verknüpfung ist daher mit drei NAND-Gattern zu bewerkstelligen (Bild 2.15a). Zuerst werden die Eingänge A und B invertiert und dann mit NAND verknüpft. Benötigt man eine NOR-Funktion, so invertiert ein viertes NAND-Gatter die ODER-Verknüpfung (Bild 2.15b). Wie praktisch, daß der 7400 gerade vier NAND-Gatter enthält!

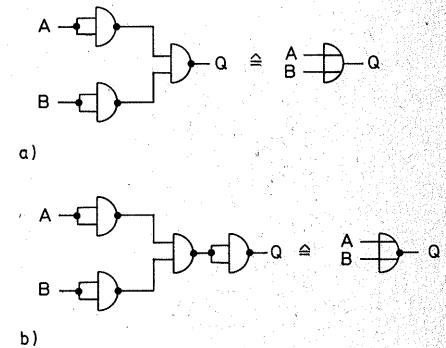


Bild 2.15 a) ODER-Gatter aus drei NAND-Gattern, b) NOR-Gatter aus vier NAND-Gattern

Umgekehrt von UND nach ODER geht es auch:

Der Kassettenrecorder spielt (Q), wenn eine Kassette eingelegt ist (A) UND wenn die Wiedergabetaste gedrückt ist (B).

$$Q = A \cdot B$$

Nun die Negationen: Der Kassettenrecorder spielt NICHT (\overline{Q}), wenn eine Kassette NICHT eingelegt ist (\overline{A})

ODER wenn die Wiedergabetaste NICHT gedrückt ist (\bar{B}):

$$\bar{Q} = \bar{A} + \bar{B}$$

Nun folgt wieder die doppelte Negation:

$$\bar{\bar{Q}} = Q = \bar{\bar{A} + \bar{B}} = A \cdot B$$

Also läßt sich mit NOR-Gattern ein UND-Gatter herstellen, indem wieder zuerst die Eingänge invertiert und anschließend mit NOR verknüpft werden (Bild 2.16a). Durch nochmalige Invertierung des Ausgangs erhält man ein NAND-Gatter (Bild 2.16b). Auch hier erweist sich die Auslegung des Grundbausteins 7402 mit vier Gattern als enorm praktisch.

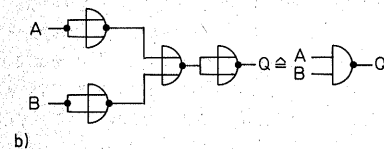
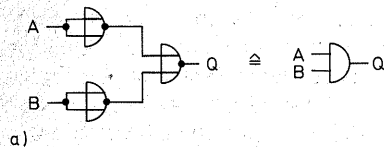


Bild 2.16 a) UND-Gatter aus drei NOR-Gattern, b) NAND-Gatter aus vier NOR-Gattern

Wozu das Ganze, wenn doch der Gatteraufwand so groß ist? Es ist für einen Hersteller sehr viel leichter und billiger, nur NAND-Gatter oder nur NOR-Gatter zu fertigen als viele verschiedene Gatter. Der scheinbare Mehraufwand ist tatsächlich eine Verbilligung.

Das exklusive ODER (EXOR, Antivalenzschaltung)

Das EXOR ist bereits eine Zusammensetzung aus UND, ODER, NICHT (siehe Seite 43). Seine logische Formel heißt:

$$Q = (A \cdot \bar{B}) + (\bar{A} \cdot B)$$

In Formeln findet man auch eine Kurzform, in der das EXOR mit einem eingekreisten OR-Zeichen „ \oplus “ ausgedrückt ist. Die Wahrheitstafel gibt das Schaltverhalten wieder:

A	B	Q = A \oplus B	bzw.	A	B	Q = A \oplus B
0	0	0		L	L	L
0	1	1		L	H	H
1	0	1		H	L	H
1	1	0		H	H	L

Merksatz: Der Ausgang des EXOR ist nur dann H, wenn die Eingänge verschiedene Werte haben. Sind die Eingangswerte gleich, seien sie beide L oder beide H, so ist der Ausgang L. Bild 2.17a zeigt die der Formel entsprechende Schaltung mit fünf Ver-

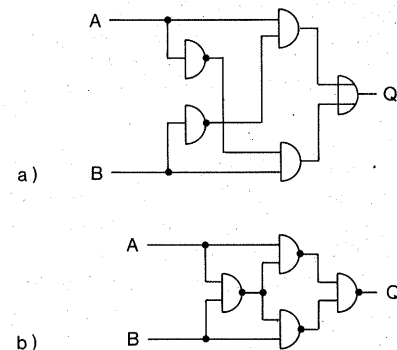


Bild 2.17 Exklusiv-ODER (EXOR), a) Grundschaltung; b) EXOR-Gatter aus vier NAND-Gattern

knüpfungsgliedern, wobei alle drei Grundglieder vorkommen. Mit nur NAND-Gattern realisiert, benötigt man nur vier Schaltglieder (Bild 2.17b); also reicht wieder ein Baustein 7400.

EXOR-Verknüpfungen werden sehr oft benötigt; daher gibt es sie auch als fertige Bausteine (z. B. 7486; vier EXOR mit je zwei Eingängen). Auch in Logikplänen wird die EXOR-Funktion als ein Schaltglied gezeichnet (Bild 2.18).

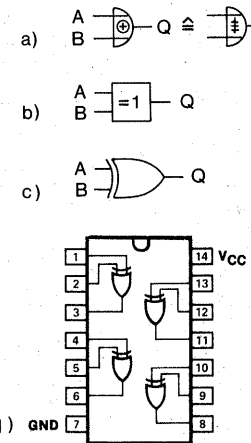


Bild 2.18 EXOR-Gatter: Symbole a) deutsch (alt, ungenormt), b) deutsch (neu), c) amerikanisch; d) Anschlußbelegung des Bausteins 7486 (nach VALVO-Unterlagen)

In unserem Mikrocomputer kommt zwar kein EXOR-Gatter vor, die Funktion spielt aber in der Programmierung eine große Rolle.

Was geschieht, wenn man einen Eingang (A) mit einer „1“ EXOR-verknüpft?

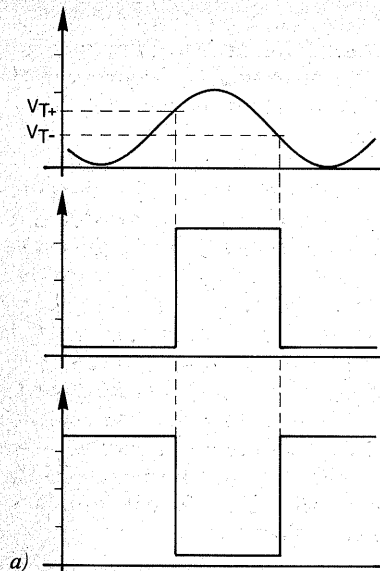
Aus einer „1“ an A wird eine „0“ am Ausgang, aus einer „0“ an A wird eine „1“ am Ausgang.

Der Ausgang des EXOR-Gatters bildet das Komplement des Eingangs; es wird in der Rechentechnik gebraucht.

Der Schmitt-Trigger

Es wurde bereits erwähnt, daß der Spannungsbereich zwischen den Pegeln H und L schnell übersprungen werden muß. Digitalschaltungen benötigen sehr schnelle Spannungswechsel, steile Anstiegs- bzw. Abfallflanken der Signale, d. h. saubere Rechtecksignale. Das kann zum Problem werden, wenn die Rechtecke z. B. durch die hohen Induktivitäten und Kapazitäten langer Leitungen „verschliffen“ werden und die Spannungen den „verbotenen Bereich“ nur langsam durchlaufen. Andererseits sollen auch verarbeitbare Signale aus Spannungen gewonnen werden, die den Spannungsbereich von L nach H nur langsam durchlaufen. Viele netzbetriebene Digitaluhren gewinnen ihren Takt aus der Sinusspannung der Netzversorgung. Die Sinusspannung muß erst in eine Rechteckspannung umgeformt werden, ehe sie für den Uhrenzähler geeignet ist.

Dazu dient der **Schmitt-Trigger** (to trigger, engl. auslösen; O. H. Schmitt veröffentlichte 1938 das Schaltungsprinzip als „Ein Elektronenröhren-Trigger“). Es handelt sich dabei um einen rückgekoppelten Verstärker mit einem Eingang und einem Ausgang. Solange die Eingangsspannung unterhalb einer bestimmten Schwelle bleibt, ist der Ausgang auf L; beim (beliebig langsamen) Überschreiten der Schwelle springt der Ausgang auf H. Sinkt die Eingangsspannung wieder, so geschieht erst einmal nichts, bis die Eingangsspannung eine bestimmte Schwelle wieder unterschreitet.



(54/74, 54LS/74LS)
VIN vs VOUT
TRANSFER FUNCTION

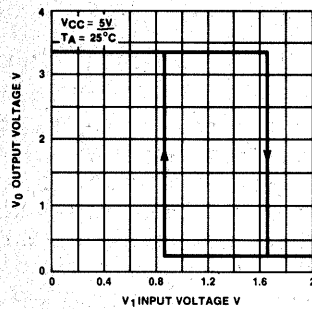


Bild 2.19 Schaltverhalten des Schmitt-Triggers. Aus einer Spannungskurve mit beliebig flachem Anstieg oder Abfall werden steile Impulsflanken erzeugt, je nach Baustein nichtinvertierend (Mitte) oder invertierend (unten). b) typische Hystereseschleife bei TTL-Bausteinen (7414, nach VALVO-Unterlagen)

tet; dann springt der Ausgang wieder auf L zurück (Bild 2.19 a). Die „Abstiegsschwelle“ V_{T-} (V von engl. Voltage, Spannung; T von engl. Threshold, Schwelle) liegt niedriger als die „Anstiegsschwelle“ V_{T+} . Die Differenz zwischen den beiden Schwellen ist die **Hysteresis** (hysteresis, griech. Verspätung; Bild 2.19 b). Sie ist notwendig, damit der Ausgang nicht ins Zittern gerät, wenn die Eingangsspannung gerade eben um die Anstiegsschwelle herumpendelt. Nach dem Erreichen der Anstiegsschwelle muß die Eingangsspannung deutlich sinken, damit der Schmitt-Trigger reagiert.

Viele Logikbausteine sind mit Schmitt-Trigger-Eingängen ausgestattet. Im Schaltzeichen deutet man die Schmitt-Trigger-Funktion oft durch Einzeichnen der Hystereseschleife an (Bild 2.20) oder auch durch Hinzufügen der Buchstaben ST.



Bild 2.20 Die Hystereseschleife im Gattersymbol gibt an, daß die Eingänge mit Schmitt-Trigger ausgestattet sind

Das RS-Flip-Flop

Das RS-Flip-Flop besteht aus zwei rückgekoppelten NAND- oder NOR-Gattern (Bild 2.21). Die Schaltung hat zwei Eingänge, S (Setzeingang, engl. set) und R (Rücksetzeingang, engl. reset). Außerdem hat sie zwei Ausgänge Q und \bar{Q} . Aus deren Bezeichnung geht schon hervor, daß sie sich immer einander entgegengesetzt verhalten.

„Setzen“ heißt:
Die Schaltung so beeinflussen, daß Q

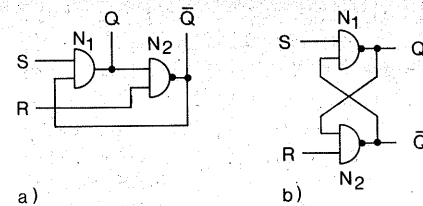


Bild 2.21 RS-Flip-Flop durch Rückkopplung zweier NAND-Gatter (a), b) übliche Darstellungsweise

den Zustand logisch 1 (H) annimmt; dadurch wird \bar{Q} logisch 0 (L).

„Rücksetzen“ heißt:
Die Schaltung so beeinflussen, daß Q den Zustand logisch 0 (L) annimmt; dabei wird \bar{Q} logisch 1 (H).

Die Schaltung hat ihren Namen „Flip-Flop“ von dem Hin- und Herkippen der Ausgänge (to flip, engl. schnellen; to flop, engl. niederfallen). Die Abkürzung ist FF.

Wie geschieht das? In Bild 2.21 ist das FF aus NAND-Gattern aufgebaut. Sie erinnern sich: Schon ein einziges L an einem NAND-Eingang zwingt den Ausgang auf H.

Im Ruhezustand sollen die Eingänge R und S auf H liegen. Man kann diesen Zustand dadurch herbeiführen, daß man sie z. B. über einen Widerstand von 1 k Ω mit der Betriebsspannung verbindet.

Wenn man S für einen kurzen Augenblick auf L legt, geht Q auf H und überträgt sein H auf den zweiten Eingang des Gatters N_2 . Da R ebenfalls H hat, muß der Ausgang von N_2 (= \bar{Q}) auf L gehen. Dieses L wird auf den zweiten Eingang von N_1 rückgekoppelt und zwingt den Ausgang Q auf H. Der Zustand bleibt durch die Rückkopplung erhalten, auch dann, wenn S wieder auf H geschaltet wird, denn ein Eingang von N_1 ist ja auf L.

Es ändert sich auch dann nichts, wenn der Pegel an S mehrmals wechselt. Der **erste Impuls** (Sprung von H auf L an S) ist **gespeichert**, das FF ist **gesetzt**.

Der Zustand kann nur geändert werden, wenn sich der gleiche Vorgang am Rückstelleingang R wiederholt: Ein L an R zwingt \bar{Q} auf H. Dieses H wird an N_1 weitergereicht, und weil auch S auf H ist, geht Q auf L, koppelt dieses L auf N_2 zurück, so daß nun der neue Zustand $Q=L$, $\bar{Q}=H$ gespeichert ist. Das FF ist **rückgesetzt**, der vorangehende Speicherzustand ist gelöscht. Eine Bedingung ist immer dabei: R und S dürfen nie gleichzeitig 1 sein. Die Tabelle gibt das Schaltverhalten an:

R	S	Q	\bar{Q}	bzw.	R	S	Q	\bar{Q}	
0	0	1*	1*		L	L	H*	H*	nicht zulässig
0	1	0	1		L	H	L	H	
1	0	1	0		H	L	H	L	
1	1	-	-		H	H	-	-	keine Änderung

Dazu ist anzumerken, daß das FF nach dem Einschalten der Versorgungsspannung einen zufälligen Zustand annimmt, sofern nicht durch Beschalten eines Eingangs mit L ein definierter Zustand herbeigeführt wird.

Und welches IC nimmt man dafür? Den Universalbaustein 7400. Er reicht für zwei RS-Flip-Flops. Das RS-FF ist auf gleiche Weise mit NOR-Gattern (z. B. 7402) zu bewerkstelligen (Bild 2.22). Der einzige Unterschied besteht darin, daß es durch einen Spannungssprung von L nach H gesetzt bzw. rückgesetzt wird. Verboten ist in dem Fall, daß R und S gleichzeitig H sind:

R	S	Q	\bar{Q}	bzw.	R	S	Q	\bar{Q}	
0	0	-	-		L	L	-	-	keine Änderung
0	1	1	0		L	H	H	L	
1	0	0	1		H	L	L	H	
1	1	0*	0*		H	H	L*	L*	nicht zulässig

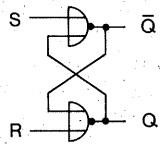


Bild 2.22 RS-Flip-Flop durch Rückkopplung zweier NOR-Gatter

Diese FF-Grundschialtung ist die einfachste **Speicherzelle**. Die Schaltung ist übrigens gar nicht mehr so jung. Sie wurde bereits 1919 von W.H. Eccles und F.W. Jordan vorgestellt – damals natürlich mit den gerade neu entwickelten Röhren. Ihr kommt eine kaum zu überschätzende Bedeutung in der Digitaltechnik, besonders in der Computerei zu, auch wenn meist weiterentwickelte FF-Schaltungen benutzt werden, von denen sie wiederum nur ein Teil ist. In unserem Mikrocomputer wird das RS-FF auch in seiner Grundschialtung benutzt (siehe CPU-Platte), z. B. zur Tastenentprellung. Jeder mechanische Schalter prellt, d.h. er springt mehrmals hin und her, bis er in seiner neuen Stellung zur Ruhe kommt und den endgültigen Zustand herstellt. Dabei erzeugt er eine nicht zu kontrollierende Anzahl von Impulsen und schafft damit undefinierte Verhältnisse. Abhilfe schafft es, den prellenden Schalter ein FF setzen oder rücksetzen zu lassen. Schon der erste Impuls wird gespeichert, stellt den gewünschten Schalt-

zustand her, und die durch das Prellen weiter entstehenden Impulse zählen nicht mehr. Auch kann man durch einen kurzen Tastendruck einen Dauerzustand herstellen, wo sonst ein einrastender Schalter erforderlich wäre.

Das getaktete Flip-Flop

Durch die Erweiterung mit einem Steuerteil ist es möglich, einen Takteingang zu schaffen. Dazu sind zwei UND-Funktionen erforderlich, die ja auch in den NAND-Gattern enthalten sind; nur darf man nie die Invertierungen außer acht lassen (Bild 2.23). Durch die Inversion wird das FF jetzt durch H gesetzt oder rückgesetzt, aber nur dann, wenn am Takteingang ebenfalls H vorhanden ist. Dadurch ist es möglich, den Zeitpunkt zu bestimmen, wann das FF schalten soll. Über den Takteingang lassen sich – zentral gesteuert von einem Taktoszillator – viele FFs koordinieren.

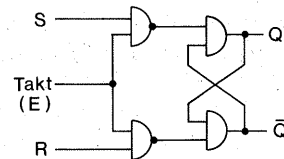
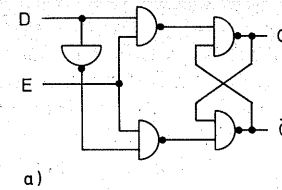


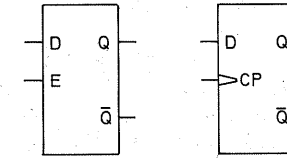
Bild 2.23 Prinzipschialtung eines RS-Flip-Flops mit Takteingang

Der Takt wird nach dem Englischen durchweg **Clock** (Uhr, Kontrolluhr) genannt. Ein Taktimpuls heißt dementsprechend Clock-Pulse, meist abgekürzt mit CP.

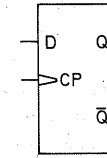
Eine besondere Stellung nimmt das **D-Flip-Flop** ein (Bild 2.24). Es besitzt nur einen Dateneingang D, er entspricht dem Setzeingang S. Da S und



a)



b)



c)

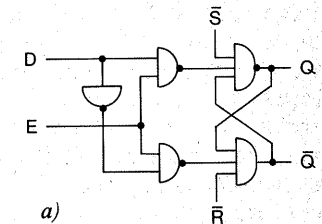
Bild 2.24 a) Prinzipschialtung eines D-Flip-Flops, b) Schaltzeichen des D-Latch, c) Schaltzeichen des D-Flip-Flop

R niemals zusammen L haben dürfen, ist R über einen Inverter mit D verbunden und kann daher nie denselben Pegel haben wie D. So gibt es keinen verbotenen Zustand beim D-Flip-Flop.

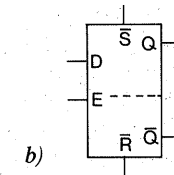
Man unterscheidet zwischen D-Latch und D-Flip-Flop:

Das D-Latch übernimmt den an D stehenden Pegel, solange am Takteingang H steht. Das bedingt, daß sich während der H-Zeit des Taktimpulses das Signal an D nicht ändern darf, es sei denn, das jeweils zuletzt vorhandene soll gelten. Das Problem ist in der Praxis meist nicht so groß, wie es zunächst erscheinen mag, weil in einem größeren System alle Vorgänge durch das Taktsignal koordiniert werden. Das Latch (to latch, engl. erfassen) ist der typische Auffangspeicher. Wegen der Eigenheit, während der gesamten aktivierenden Taktzeit die Daten vom Eingang zum Ausgang zu transportieren, auch wenn sie sich ändern, heißen solche Speicher auch **transparente Latches**. Der Takteingang wird

meist mit „E“ (enable, engl. Freigabe-eingang) bezeichnet (Bild 2.24 b). **Latches werden zu Abertausenden in Computern und anderen Digitalgeräten als Auffangspeicher benutzt.** In erweiterten Formen hat das D-Latch noch einen Setzeingang \bar{S}_D (set data, Setzen mit Vorrang vor dem Clock) und einen Rücksetzeingang \bar{R}_D (reset data), über die das Latch unabhängig vom Clockimpuls gesetzt oder, was meist viel wichtiger ist, rückgesetzt werden kann. Erreicht wird das z. B. dadurch, daß die beiden eigentlichen FF-Gatter drei Eingänge bekommen. Der dritte Eingang ist dann \bar{S}_D bzw. \bar{R}_D . Die Negation deutet an, daß diese Eingänge wie die des RS-FF aus NAND-Gattern durch L aktiviert werden (Bild 2.25).



a)



b)

Bild 2.25 a) Prinzipschialtung eines D-Latch mit asynchronen Setz- und Rücksetzeingängen, b) Schaltzeichen

Das D-Flip-Flop unterscheidet sich vom D-Latch durch das Verhalten des Takteinganges. Das D-Flip-Flop wird nur mit der **Fanke** eines Taktimpulses (Clock-Pulse, CP) gesetzt, es wird „flankengetriggert“, je nach Typ mit

dem Spannungssprung von L nach H (ansteigende Flanke) oder mit dem Spannungssprung von H nach L (abfallende Flanke). Während einer „Ruhezeit“ am Takteingang, sei sie L oder H, ändert sich der Speicherzustand im Gegensatz zum Latch nicht. Der Takteingang der D-Flip-Flops wird meist mit „CP“ bezeichnet und im Schaltsymbol mit einem kleinen Dreieck markiert (Bild 2.24 c).

Eine weitere sehr wichtige Form des getakteten FF ist das **JK-Flip-Flop** (Bild 2.26). Es hat zwei Dateneingänge J und K, einen Takteingang CP, die Ausgänge Q und \bar{Q} und kann außerdem noch taktunabhängige („asynchrone“) Setz- bzw. Rücksetzeingänge \bar{S}_D und R_D haben. Es bietet gegenüber den bisher beschriebenen FF-Arten allerlei Komfort. Dieser läßt sich am besten durch eine Funktionstafel veranschaulichen. Darin bedeutet „ t_n “ die Zeit vor einem Clockimpuls, „ t_{n+1} “ die Zeit nach dem folgenden Clockimpuls. Die asynchronen Eingänge sind in der Tafel nicht berücksichtigt.

Zeile	J	K	t_n		t_{n+1}	
			Q	\bar{Q}	Q	\bar{Q}
0	0	0	0	1	0	1
1	0	1	0	1	0	1
2	1	0	0	1	1	0
3	1	1	0	1	1	0
4	0	0	1	0	1	0
5	0	1	1	0	0	1
6	1	0	1	0	1	0
7	1	1	1	0	0	1

Im Klartext bedeutet das:

1. Wenn J und K (also beide) 0 sind, ändert sich durch Takten nichts (Zeilen 0 und 4).
2. Wenn J=1 und K=0 ist, nimmt Q den Zustand „1“ an (Zeile 2) oder be-

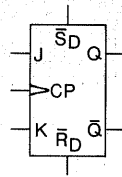


Bild 2.26 a) Schaltsymbol des JK-Flip-Flops, b) Anschlußbelegung des Bausteins 7476 mit zwei JK-Flip-Flops (nach VALVO-Unterlagen)

hält ihn bei (Zeile 6). Mit J=1 und K=0 wird das FF durch den folgenden Clockimpuls gesetzt.

3. Wenn K=1 und J=0 ist, nimmt Q den Zustand „0“ ein (Zeile 5) oder behält ihn bei (Zeile 1). Mit dieser Kombination wird das FF durch den folgenden Clockimpuls rückgesetzt.

4. Wenn J und K (also beide) = 1 sind, schaltet das Flip-Flop um; Q springt entweder von 0 auf 1 (Zeile 3) oder von 1 auf 0 (Zeile 7).

Das letztgenannte Verhalten ist etwas ganz Neues. Bei J·K=1 zählt der Ausgang Q jeden zweiten Clockimpuls. Durch Hintereinanderschalten mehrerer solcher FFs kann man daher Zähler für das Dualsystem (siehe Seite 146) bauen.

Diese Funktion spielt in unzähligen Geräten die wesentliche Rolle. Daher nennt man die Gruppe der FF-Arten, die diese Funktion besitzen, auch Zähl-Flip-Flops.

Die Zählfähigkeit brauchen wir in unserem Mikrocomputer; sie ist im Mikroprozessor und im AD-Wandler versteckt.

Mit der Zählfähigkeit fällt aber auch noch eine andere sehr willkommene Eigenschaft gewissermaßen als Nebenprodukt ab (Bild 2.27). Der Clockeingang kann durch eine Impulsfolge mit sehr verschiedenem H/L-Verhältnis („Tastverhältnis“, „duty cycle“)

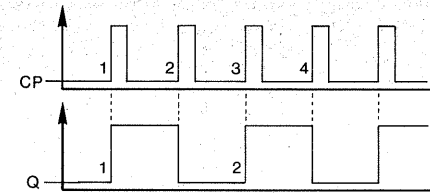


Bild 2.27 Schaltverhalten des Zähl-Flip-Flops: Die Ausgangsimpulse haben unabhängig vom Tastverhältnis der Clockimpulse ein Tastverhältnis 1:1, die Clockfrequenz wird im Verhältnis 1:2 geteilt

gesteuert werden. Durch das Umschalten entstehen an Q bzw. \bar{Q} Impulse mit **genau gleicher Dauer von H und L**. Die Pulsfrequenz wird dabei halbiert, denn aus je zwei Clockimpulsen wird eine neue Q-Periode. Diese Impulsaufbereitung benutzen wir beim Clockoszillator unseres Mikrocomputers.

Für die asynchronen Eingänge \bar{S}_D und R_D gilt wie für das RS-FF, daß nie beide zugleich L sein dürfen.

Abschließend sei ausdrücklich darauf hingewiesen, daß mit dieser kurzen Beschreibung bei weitem nicht alle Eigenschaften der Familie der JK-Flip-Flops beschrieben sind. Zum Verständnis unseres Mikrocomputers mag sie aber fürs erste reichen.

Das Register

Ein einziges Latch oder Flip-Flop kann nur den logischen Wert jeweils einer einzigen Leitung, d. h. **1 Bit** (von engl. binary digit, Binärzahl, siehe Seite 94) speichern – also höchstens zwei Informationen, entweder eine 0 oder eine 1. Das ist nicht viel. Mikrocomputer haben daher eine Reihe paralleler Datenleitungen, z. B. 4 oder 8, neuere 16 oder gar 32. Das heute übliche Format ist eine „Breite“ aus

acht parallelen Leitungen (8 Bit). Damit ist die Darstellung von $2^8 = 256$ verschiedenen Kombinationen aus Nullen oder Einsen möglich. Um solch ein Datenwort (Byte) aus 8 Bit speichern zu können, benötigt man acht Latches oder andere Flip-Flops. Die Anordnung so vieler Speicherzellen, wie nötig sind, um ein Datenwort zu speichern, nennt man **Register**; in unserem Beispiel besteht ein Register aus acht Latches. Register spielen in der Computertechnik eine große Rolle. So enthält z. B. jeder Mikroprozessor eine Anzahl von Registern, die jeweils ein Datenwort speichern können.

Man kann sich ein Register in übersichtlicher Form dadurch herstellen, daß man Latches in der für die Wortbreite benötigten Anzahl – im obigen Beispiel acht – nebeneinander anordnet und jede Datenleitung mit dem D-Eingang eines Latches verbindet. Wenn man außerdem noch alle Takteingänge zusammenschaltet, übernehmen alle Latches beim Taktimpuls **gleichzeitig** die augenblicklichen Zustände der Datenleitungen.

Derartige Registerbausteine gibt es bereits fertig als integrierte Schaltungen, z. B. in dem TTL-Baustein 74LS273 (Bild 2.28). Die Takteingänge sind über einen Puffer am Anschluß CP (Clock-Pulse) zusammengefaßt. Der Puffer bewirkt, daß am Eingang CP nur ein einziger Lastfaktor (siehe Seite 64) vorhanden ist, während ohne Puffer acht Lastfaktoren vorhanden wären. Sobald CP auf H geschaltet wird, übernehmen die Dateneingänge Q_0 bis Q_7 die an den Dateneingängen D_0 bis D_7 vorhandenen logischen Zustände und speichern sie.

Die Rücksetzeingänge R_D sind – ebenfalls über einen Puffer – zu ei-

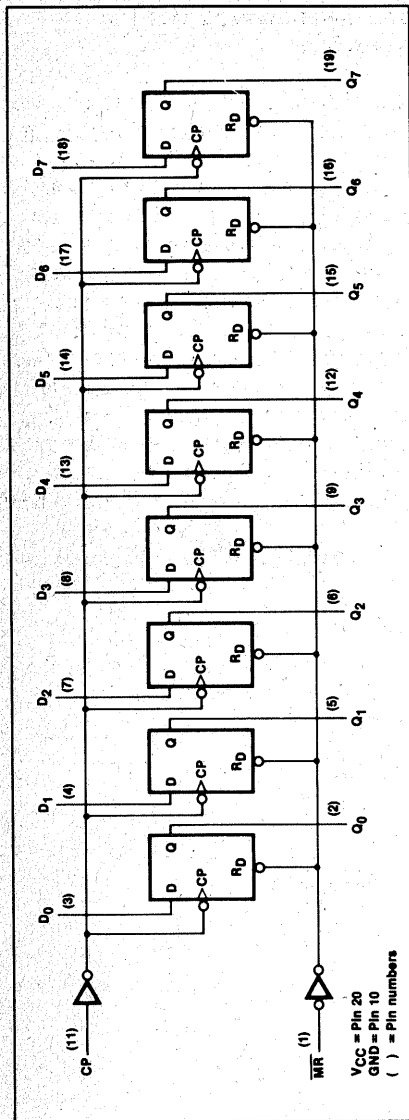


Bild 2.28 Register mit acht D-Flip-Flops (74LS273, nach VALVO-Unterlagen)

nem gemeinsamen Rücksetzeingang MR (Master Reset, engl. Hauptrücksetzeingang) zusammengefaßt. Sobald MR auf L geschaltet ist (Negation der Bezeichnung!), nehmen alle Q den Wert 0 an – unabhängig vom Geschehen am Takteingang.

Dieser Rücksetzeingang ist keine notwendige Eigenschaft des Registers. Den Effekt, alle Speicherzellen auf 0 zu stellen, würde man auch dadurch erreichen, daß man sie alle mit 0 lädt. Das ist etwa der übliche Weg, prozessorinterne Register auf 0 zu stellen. Ein gemeinsamer Rücksetzeingang ist aber sehr bequem zu bedienen, so z. B. durch die Clear-Taste bei Taschenrechnern.

Ein Register mit paralleler Ein- und paralleler Ausgabe ist nur eine Registerform unter vielen. Eine grundsätzlich andere Registerform ist das **Schieberegister**. In dieses werden die Bits eines Datenwortes nacheinander auf einer einzigen Datenleitung hineingeschoben und können auch nacheinander (seriell) wieder gelesen werden. Bei anderen Registern kann seriell eingespeichert und parallel ausgelesen werden – oder umgekehrt. Da wir diese Registerformen aber nicht in unserem Computer benutzen, mag dieser kurze Hinweis darauf genügen.

Wichtige technische Eigenschaften der TTL-Familien 74XX und 74LSXX

Die Datenbücher stecken voll von genauen Angaben über diese ICs. Sie sind für den Schaltungsentwickler nötig. Für Sie als Nachbauer sind nur wenige wirklich bedeutend. Die sollten Sie sich aber immer vergegenwärtigen.

Die Betriebsspannung

Die Betriebsspannung der TTL-ICs beträgt $5V \pm 5\%$, d. h. sie darf zwischen 4,75 V und 5,25 V schwanken, mehr nicht! TTL-ICs können allerlei Mißhandlungen ertragen, nicht aber eine falsche Betriebsspannung, schon gar keine zu hohe! Ab 6 V wird es für die ICs gefährlich, 7 V ist die **absolute Obergrenze**. Benutzen Sie daher kein Netzgerät mit einstellbarer Ausgangsspannung. Ein Fehlgriff kann sehr teuer werden.

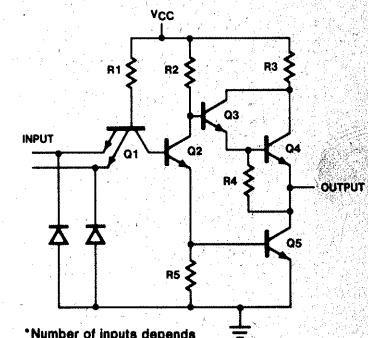
In den Logikplänen fehlen meist die Leitungen für die Spannungsversorgung, weil sie als selbstverständlich angesehen werden und mit der „Logik“ nichts zu tun haben.

Für diese Logikfamilie ist der Minuspol der Spannungsquelle, das Bezugspotential (0 V, „Masse“), an den IC-Anschlüssen mit GND (ground, engl. Erde) gekennzeichnet.

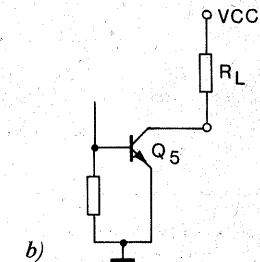
Der Pluspol wird jeweils an den Anschluß VCC geschaltet (V von voltage, engl. Spannung; C von collector; die Verdopplung ist das Pluralzeichen).

Das Zusammenschalten von Bausteinen

Grundsätzlich darf ein Ausgang nur auf einen oder mehrere Eingänge geschaltet werden. **Im Normalfall dürfen nie Ausgänge zusammengeschaltet (parallelgeschaltet) werden**, denn wenn der eine Ausgang H, der andere L hat, gibt es einen Kurzschluß. Bild 2.29 a zeigt das TTL-Normalgatter (NAND). Q₄ und Q₅ sind die Ausgangsstufe. Bei L am Ausgang schaltet Q₅ durch und Q₄ sperrt, bei H leitet Q₄ und Q₅ sperrt. Der Kurzschluß beim Zusammenschalten wird das IC nicht sofort zerstören, weil der Schutzwiderstand R₃ (= 130 Ω) den



a)



b)

Bild 2.29 a) Normalschaltung eines TTL-NAND-Gatters (nach VALVO-Unterlagen), b) Open-Collector-Ausgang

Strom begrenzt. Es wird aber in keinem Fall mehr definierte Ausgangsverhältnisse geben.

Zwei Ausnahmen gibt es von dieser Regel, den Open-Collector-Ausgang und den Tri-State-Ausgang.

Der Open-Collector-Ausgang

Bei Varianten der Gatterschaltungen besteht der Ausgang nicht aus der Gegentaktendstufe, sondern nur aus deren „unterem“ Transistor (Bild 2.29b; der Transistor entspricht Q_2 in Bild 2.29a), dessen Kollektor „offen“, d.h. ohne Verbindungen innerhalb des IC nach außen geführt ist, daher der Name Open-Collector (open, engl. offen). Damit am Ausgang überhaupt H erscheinen kann, muß der Kollektor mit einem „Pull-Up-Widerstand“ (to pull up, engl. hochziehen) mit VCC verbunden werden. Dieser Widerstand kann auch ein kleiner Verbraucher sein, z.B. eine Anzeigeeinheit aus Widerstand und LED (siehe Bild 6.5).

Der Open-Collector-Ausgang (abgekürzt o.c.) kann schon allein wegen des Fehlens von R_3 besser als Schalter für einen Verbraucher eingesetzt werden.

Sein zweiter Vorteil ist, daß er mit anderen seinesgleichen parallelgeschal-

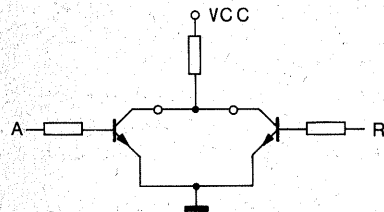


Bild 2.30 Wired-NOR durch Zusammenschaltung zweier Open-Collector-Ausgänge an einem gemeinsamen Pull-Up-Widerstand

tet werden darf und dabei wieder zu logischen Verknüpfungen führt. In Bild 2.30 sind zwei solcher Ausgänge mit einem gemeinsamen Pull-Up-Widerstand parallelgeschaltet. Es muß nicht unbedingt ein gemeinsamer Widerstand sein. Jeder Ausgang darf einen eigenen Widerstand haben, und parallelgeschaltet wirken sie ja wie ein einziger, wenn auch ein kleinerer. Was geschieht, wenn an A ein H und an B ein L liegt? T_1 schaltet durch, und Q geht auf L (ein Transistor in Emitterschaltung ist ein Inverter). Das gleiche geschieht, wenn $B=H$ und $A=L$ ist. Wenn sowohl A als auch $B=H$ sind, schalten beide Transistoren durch, und nun ist Q „erst recht“ L. Nur dann, wenn weder A noch B durchschalten, ist $Q=H$. In der Funktionstafel sieht das Verhalten wie folgt aus:

A	B	Q
0	0	1
0	1	0
1	0	0
1	1	0

und das ist die NOR-Funktion (siehe Seite 49). Da dieses NOR verdrahtet ist, heißt es „Wired NOR“ (wire, engl. Draht). Sie finden es in dem beschriebenen Mikrocomputer an der Leitung WRITE (Datenein-/ausgabe), IC2, Pin 9/8 und CPU, IC4, Pin 5/6).

Tri-State-Logik

In einem Computer, gleichgültig wie groß er ist, werden Daten (das sind immer nur H- oder L-Pegel) von einer Stelle zur anderen verschoben, und es sind viele Stellen, zwischen denen Daten ausgetauscht werden. Bild 2.31 zeigt z.B. eine Zusammenschaltung

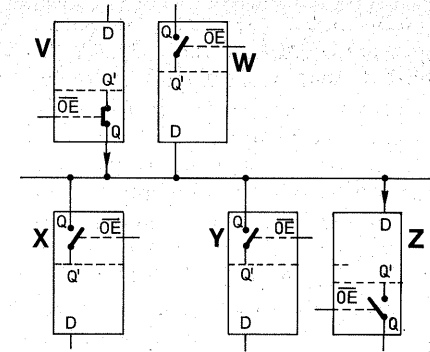


Bild 2.31 Mehrere Tri-State-Ausgänge von Logikbausteinen können zusammengeschaltet werden, sofern sie durch eine Steuerlogik so verwaltet werden, daß jeweils nur ein einziger aktiv ist

von Flip-Flops, die entweder mit ihren Eingängen oder ihren Ausgängen an eine Leitung angeschlossen sind. Es sind also scheinbar mehrere Ausgänge parallelgeschaltet – in Wirklichkeit sind sie es aber doch nicht. Ein Transportbeispiel mag das verdeutlichen:

Es soll ein Signal von V nach Z transportiert werden. Die Flip-Flops W, X und Y sollen nicht betroffen sein bzw. das Signal nicht beeinflussen können; das könnten z.B. W, X und Y durch die angeschlossenen Ausgänge. Daher ist in dem IC zwischen dem eigentlichen FF-Ausgang Q' und dem Ausgangsanschluß Q ein elektronischer Schalter gesetzt. Dieser Schalter kann den Ausgang des Flip-Flops an die Leitung anschließen oder von ihr abtrennen. „Abtrennen“ heißt, Q sowohl nach VCC (=H) wie nach GND (=L) hochohmig machen.

Ein solcher Ausgang hat drei Zustände, nämlich H, L oder „hochohmig“; daher der Name Tri-State (oder auch 3-State; tri, lat. drei). Der dritte, hoch-

ohmige Zustand ist kein logischer Zustand, er bedeutet, daß das Schaltglied von der gemeinsamen Leitung abgetrennt ist.

Der Schalter wird von dem Anschluß OE (Output Enable, engl. Ausgang Freigabe) bedient; die Negation über den Buchstaben deutet an, daß der Ausgang bei $\overline{OE}=L$ aktiv ist; bei $\overline{OE}=H$ ist er hochohmig. Über die OE-Anschlüsse werden jeweils nur die Ausgänge aktiviert, die während einem gewissen kurzen Zeitabschnitt benutzt werden sollen, so daß in Wirklichkeit doch immer nur ein einziger Ausgang auf eine Leitung geschaltet ist, obwohl doch alle angelötet sind. In „Ruhezeiten“ sind sogar alle Ausgänge elektronisch abgetrennt.

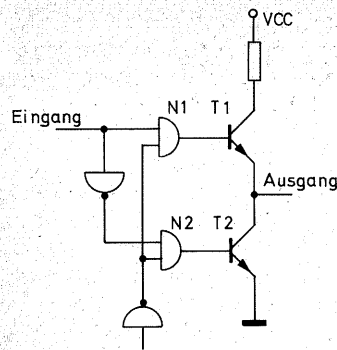
Wie funktioniert das Abschalten des Ausgangs?

Auf das eigentliche Logikglied folgt ein Puffer mit einer Gegentaktendstufe (Bild 2.32). Wenn man die Basen der Endstufentransistoren gleichzeitig auf 0 V schaltet, sperren beide, und der Ausgang ist sowohl nach VCC als auch nach 0 V hochohmig – also abgetrennt.

Wenn der Steuereingang $\overline{OE}=L$ ist, steht an den beiden UND-Gattern N1 und N2 aufgrund der Negation je ein H; beide UND-Gatter sind für ankommende Signale geöffnet. Ein H am Eingang schaltet N1 auf H und N2 auf L, dadurch öffnet T1 und T2 sperrt. Der Ausgang ist H.

Ein L am Eingang schaltet N1 auf L, dagegen N2 auf H; folglich sperrt T1, während T2 öffnet. Der Ausgang ist L.

Steht am Steuereingang $\overline{OE}=H$, so wirkt es wegen der Negation an den UND-Gattern als L. Beide UND-Gatter schalten unabhängig vom Geschehen am Eingang auf L. Dadurch



a) Steureingang \overline{OE}

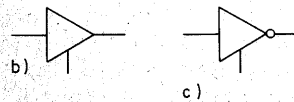


Bild 2.32 a) Schaltungsprinzip des Tri-State-Puffers, b) Schaltzeichen für den nichtinvertierenden, c) Schaltzeichen für den invertierenden Puffer

fehlt beiden Transistoren die Basisspannung, und deswegen sperren sie beide. Der Ausgang ist folglich sowohl nach VCC als auch nach 0V hochohmig. Er befindet sich im „Tri-State-Zustand“, der in Funktionstafeln meist mit „Z“ bezeichnet wird. Wichtig ist nur, daß die \overline{OE} -Anschlüsse richtig und termingerecht „veraltet“ werden. Doch solche Verwaltungsarbeit ist gerade das, was der Computer besonders gut kann.

Eingangs- und Ausgangslastfaktoren (fan in, fan out)

Um die Bedeutung der Lastfaktoren zu verstehen, muß man sich vergegenwärtigen, wie ein TTL-Eingang funktioniert. Das Prinzip sei an einem UND-Eingang erklärt (Bilder 2.29a und 2.33). Der Gattereingang in Bild 2.29a besteht aus einem Multi-

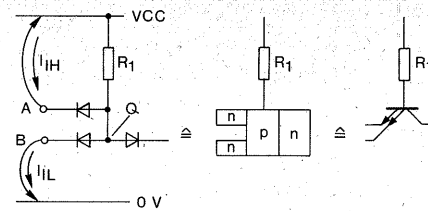


Bild 2.33 Der Multi-Emitter-Transistor entsteht durch die Zusammenfassung der Schaltdioden, deren Anoden von einer gemeinsamen p-Zone gebildet werden

Emitter-Transistor (multi, lat. viele). Die beiden Dioden gegen Masse sind Schutzdioden, die die Eingänge vor negativen Spannungen schützen sollen, und haben mit der logischen Funktion nichts zu tun. Der Transistor Q_1 ist nicht als Transistor zu sehen, sondern als Zusammenschaltung von Dioden (pn-Übergängen). Die Dioden sind in Bild 2.33 (links) einzeln gezeichnet. Ihre p-Zonen (Anoden) sind aber auf dem Kristall zu einer Zone zusammengefaßt und bilden dadurch gewissermaßen die Basis des Transistors (Bild 2.33 Mitte), während die n-Zone von D_3 den Kollektor des Transistors bildet.

Wenn ein Eingang an VCC (= H) geschaltet ist (z. B. A in Bild 2.33), fließt kein Strom hinein (Diode in Sperrrichtung), abgesehen von dem geringen Leckstrom in der Größenordnung weniger μA . Der Eingangsstrom für den H-Fall I_{IH} (Input HIGH Current) ist vernachlässigbar gering.

Ist ein Eingang aber auf GND (= L) geschaltet, so fließt aus dem Eingang ein Strom hinaus. In Datenbüchern heißt er I_{IL} (Input LOW Current). Seine Größe richtet sich nach dem Widerstand R_1 (Ohmsches Gesetz). Bei TTL-Gattern beträgt er von wenigen Ausnahmen abgesehen einheitlich

maximal $-1,6 \text{ mA}$ (das Minuszeichen gibt an, daß der Strom aus dem Eingang fließt). Meist ist er glücklicherweise geringer; er stört nämlich sehr.

R_1 beträgt bei Standard-TTL-Gattern ca. $4 \text{ k}\Omega$. Bei einem Spannungsabfall von $6,4 \text{ V}$ (7 V absolutes Maximum abzüglich einer Diodenschwellenspannung von $0,6 \text{ V}$) fließen die genannten $-1,6 \text{ mA}$. Bei $V_{CC} = 5 \text{ V}$ entsteht ein Spannungsabfall von $4,4 \text{ V}$, und wenn man ca. $0,4 \text{ V}$ Spannungsabfall am steuernden Schaltglied in die Rechnung einbezieht, bleiben für R_1 nur noch 4 V übrig; I_{IL} beträgt dann „nur“ noch ca. -1 mA . Dieser Strom ist eine Lasteinheit (μL , unity Load). Die Anzahl der Lasteinheiten, die ein Eingang im L-Zustand abgibt, ist sein fan in (fan, engl. Fächer; in, von input). Normalerweise hat ein TTL-Eingang ein fan in = 1. Es gibt allerdings auch einige wenige Ausnahmen.

Warum ist das so wichtig? Das vorangehende Schaltglied, das einen Ausgang auf L schaltet, muß den Eingangsstrom I_{IL} nämlich aufnehmen können, ohne daß sich durch Spannungsabfall an seinem Innenwiderstand seine Ausgangsspannung über die erlaubte L-Grenze von $0,4 \text{ V}$ (Bild 2.1) erhöht. Das heißt, daß ein Ausgang nur eine bestimmte Anzahl von Eingängen auf L ziehen kann. Diese Anzahl ist das fan out.

Bei TTL-Gattern beträgt das fan out in der Regel mindestens zehn; d. h. ein TTL-Ausgang kann sich auf mindestens zehn Eingänge auffächern und sie auf L ziehen. Die im Datenbuch angegebene Anzahl darf aber nicht überschritten werden, weil sonst die Ausgangsspannung unzulässig ansteigt.

Die Ausgänge von Mikroprozessoren,

die TTL-kompatibel (lat./engl. verträglich mit) sind, haben durchweg ein fan out = 1; d. h. sie können nur einen einzigen TTL-Eingang (garantiert) schalten.

Zum Glück gibt es neben den Standard-TTL-Schaltungen noch die Serie 74LSXX (LS steht für Low Power Schottky). Die für unseren Zweck wichtigste Eigenschaft ist, daß R_1 mit 15 bis $20 \text{ k}\Omega$ erheblich größer ist und damit I_{IL} nur noch maximal ein Viertel eines Standard-TTL-Eingangsstromes beträgt. An einen Mikroprozessorausgang kann man daher vier LS-Eingänge anschließen. Daher benutzen wir für die umgebende Logik ausschließlich Bausteine der Serie 74LSXX.

Eingangsspannungen

Die in Bild 2.1 angegebenen Bereiche für L und H sind bis auf die absoluten Grenzen ausgedehnt. Interessant ist wieder der L-Wert; er darf bis zu $0,8 \text{ V}$ ansteigen – dann ist allerdings auch kein Störabstand mehr vorhanden. Diese $0,8 \text{ V}$, die der Eingang noch als L erkennt, sind wichtig, denn wenn man ein Signal mit einer Diode (analog zu Bild 2.31) auf L zieht, kann die Spannung wegen der Schwellenspannung der Diode ja nicht unter $0,6 \text{ V}$ sinken.

Offene (unbeschaltete) TTL-Eingänge verhalten sich so, als wären sie H. Das ist aus Bild 2.33 plausibel. Wenn eine der Eingangsdiode nicht beschaltet ist, liegt Q ja über R_1 an VCC, also H. Wo ein Dauer-H erwünscht ist, könnte man den Eingang offen lassen. Trotzdem sollte man es nicht tun, denn das Gatter wird dadurch anfällig für Störimpulse; das gilt besonders für LS-Bausteine.

Kapitel 3

Die Baustufen unseres Mikrocomputers

Bild 3.1 zeigt alle vorläufigen Baustufen unseres Mikrocomputers als Blockschaltbild. „Vorläufig“ bedeutet, daß der Computer noch weiter ausbaufähig ist. Die vorhandenen Baustufen sind die typischen Funktionseinheiten, wie sie so oder so ähnlich in jedem Mikrocomputersystem zu finden sind.

Das **Netzteil** (1) ist gewissermaßen die Grundlage des ganzen Werks, denn ohne Energieversorgung funktioniert nichts. Es liefert in der Grundversion die Versorgungsspannung $V_{CC} = +5\text{ V}$ für eine Stromentnahme von ca. 0,5 A. Nur der Analog-Digital-Wandler benötigt noch eine Spannung von -5 V gegen Masse,

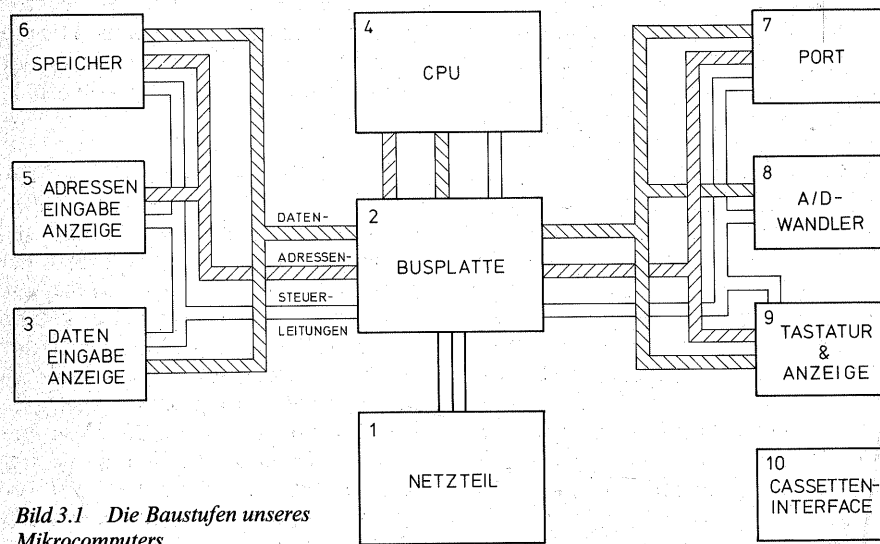


Bild 3.1 Die Baustufen unseres Mikrocomputers

mit einem Strombedarf von wenigen mA. In einer weiteren Ausbaustufe kann das Netzteil auch diese Spannung liefern. Außerdem kann es noch einen von der Netzfrequenz abgeleiteten 50-Hz-Takt liefern, der für viele Programme, in denen die Zeit eine Rolle spielt (Uhr, Geschicklichkeitsspiele usw.), sehr nützlich ist.

Die **Busplatte** (2) besteht aus einer Anzahl von parallelgeschalteten Buchsen, so daß alle wichtigen Systemleitungen zu greifen sind. Sie ist kein eigentliches Computerteil, sondern eine fürs Experimentieren sehr bequeme Einrichtung, um (weitere) Baueinheiten an den Computer anzuschließen oder von ihm abzutrennen, ohne gleich zum LötKolben greifen zu müssen.

Die **CPU** (3) (Central Processing Unit, engl. Zentraleinheit) ist die Baueinheit, auf der der Mikroprozessor nebst allerlei Anzeigen untergebracht ist, an denen seine augenblickliche Tätigkeit abzulesen ist. Daneben befinden sich auf der CPU-Platte einige Logikgatter, die die Steuersignale, welche der Prozessor abgibt oder empfängt, speziell für die Eigenschaften unseres Computers aufbereiten.

Auf dieser Platte geschieht die eigentliche Datenverarbeitung. Allein kann der Prozessor allerdings noch nichts. Er braucht zumindest die Daten, die er verarbeiten soll, und dazu natürlich die genauen Arbeitsanweisungen („Befehle“, „Instruktionen“), aus denen hervorgeht, was er mit den Daten machen soll.

Im „Normalfall“ erhält der Prozessor sein **Programm**, das ist die Folge von Befehlen und Daten, aus dem Speicher.

Der **Speicher** (6). Das typische Anwendungsgebiet des Mikrocomputers ist die Steuer- und Regeltechnik. Man muß dabei nicht immer gleich an Großroboter denken. Der Mikrocomputer ist für viele „kleine“ Anwendungszwecke gut, zur Steuerung einer Näh- oder Waschmaschine, eines Kassettendecks, einer Waage usw. Wird ein Mikrocomputer in einem Gerät für nur einen Zweck eingesetzt, erhält er sein Programm in einem **ROM** (Read Only Memory, engl. Nur-Lese-Speicher), einem unveränderlichen „Festwertspeicher“.

Soll ein Mikrocomputer abwechselnd die unterschiedlichsten Aufgaben übernehmen, baut man den Speicher so auf, daß man Programme eingeben („einschreiben“) und nach Bedarf beliebig durch neue ersetzen kann. Ein solcher Speicher ist ein **RAM** (Random Access Memory, engl. Speicher mit wahlfreiem Zugriff). Der Begriff ist leicht irreführend, denn gemeint ist ein „Schreib-Lese-Speicher“, in den man nach Belieben etwas einschreiben oder aus dem man den Mikroprozessor lesen lassen kann. Unsere Speicherkarte kann sowohl Festwert- als auch Schreib-Lese-Speicher aufnehmen.

Der **Port** (7). Die Arbeit des Mikrocomputers hat nur dann einen Sinn, wenn er mit seiner Umwelt Kontakt aufnehmen kann, wenn er z. B. Temperaturen, Drücke usw. „wahrnehmen“ und das Ergebnis seiner Arbeit an irgendwelche ausführenden Organe (Motor oder Ventil einer Maschine, Heizung, Anzeige usw.) weitergeben kann. Dazu dient der **Portbaustein**. Er bildet das Verbindungsglied, das **Interface**, zur Außenwelt, der **Peripherie** (griech., das „Drumherum“, die Umgebung).

Mit den genannten Baugruppen „Netzteil“ (oder Batterie), „CPU“, „Speicher“ (ROM und kleines RAM) und „Input/Output-Port“ ist ein Mikrocomputer komplett und wird so millionenfach in Geräten verschiedenster Art angewandt. Meist ist er dabei so klein, daß alle Bauelemente auf eine Europakarte passen.

Die übrigen Baugruppen bilden eine Ergänzung, die unseren Mikrocomputer für vielfältige Zwecke flexibel machen.

Die **Dateneingabe und -anzeige** (4). Zum Lernen und Üben (aber nicht nur dazu) ist es sehr angenehm, dem Prozessor Befehle und Daten direkt einzugeben und seine Arbeitsergebnisse Schritt um Schritt abzulesen. Dazu dient die Dateneingabe und -anzeige. Sie ist die besonders einfache Ausführung eines Ports.

Die **Adresseingabe und -anzeige** (5). Da wir den Speicher selbst mit Programmen zu füllen gedenken, benötigen wir eine Adresseingabe und -anzeige. Der Speicher besteht aus einer großen Anzahl von Registern, von denen jedes seine eigene „Hausnummer“, seine Adresse hat. Wenn wir ein Daten- oder Befehlswort speichern wollen, müssen wir dem Speicher nicht nur das Wort mitteilen, sondern auch die Adresse, unter der das Wort abgelegt werden soll.

Die **Tastatur & Anzeige** (9) erhöht den Eingabe- und Ablesekomfort erheblich. Im Prinzip leistet sie das, was Daten- und Adreßbaustein auch leisten, aber sie erspart es dem Benutzer, sich mit Ketten von Nullen und Einsen abzuplagen.

Der **Analog-Digital-Wandler** (8) setzt analoge Spannungen, wie sie z.B. ein Wärme-, Licht-, Gas- oder Drucksensor liefert, in die Sprache des Computers um, also in ein Muster von Nullen und Einsen. Damit kann der Computer direkt für Meß-, Steuer- und Regelzwecke eingesetzt werden. Beispiele sind etwa die Verwendung als Digitalvoltmeter, Thermometer, Luxmeter usw.

Das **Kassetteninterface** (10) ermöglicht es, Programme auf eine Tonbandkassette zu überspielen und darauf zu speichern, während man den Speicherbaustein für andere Zwecke benutzt. Wenn man sie wiederverwenden will, kann man sie über das Kassetteninterface wieder in den Speicher übertragen. Die Tonbandkassette ist das ideale Medium, Programme mit anderen Computerfreunden auszutauschen.

Abschließend sei angemerkt, daß es für die Inbetriebnahme und zum Kennenlernen des Systems vorteilhaft ist, die Baugruppen in der Reihenfolge herzustellen, wie sie in dem Blockschaltbild numeriert sind. Wer noch keine oder nur sehr geringe Übung im Löten hat, sollte jedoch lieber mit der Busplatte (Baugruppe 2, Kapitel 5) beginnen, weil die Lötarbeiten daran am leichtesten sind.

Kapitel 4

Das Netzteil (Baugruppe 1)

Schaltungstechnik

Das Netzteil stellt alle Versorgungsspannungen für das gesamte System zur Verfügung. Der eigentliche Mikrocomputer (CPU, Speicher, Ein- und Ausgabebausteine) benötigt eine einheitliche Versorgungsspannung von $+5\text{ V} \pm 5\%$ (4,75 bis 5,25 V). Bild 4.1 zeigt die einfachste Art der Spannungsregelung: (Transformator), Brückengleichrichter (Gl 1), Lade-Elko (C3), Spannungsregler (IC 1). Die Anzeige (R 1 + LED) ist nicht unbedingt nötig, als Einschaltkontrolle aber doch sehr nützlich. Die Kondensatoren C1 und C2 sollen Störimpulse

unterdrücken, die über das Stromnetz in den Computer eindringen könnten, z. B. Hochfrequenzeinstrahlungen des Ortssenders.

Der Netztransformator

Dem aufmerksamen Betrachter wird nicht entgehen, daß der Transformator im Stromlaufplan fehlt. Er fehlt absichtlich; **niemand soll dazu verleitet werden, Leitungen für Netzspannung zu verdrahten und möglicherweise dadurch zu Schaden zu kommen.** Als Netztransformator ist daher ein **Spielzeug- oder Experimentiertransformator** vorgesehen, z. B. ein Eisenbahntransformator o. ä. (Bild 4.2). Wichtig

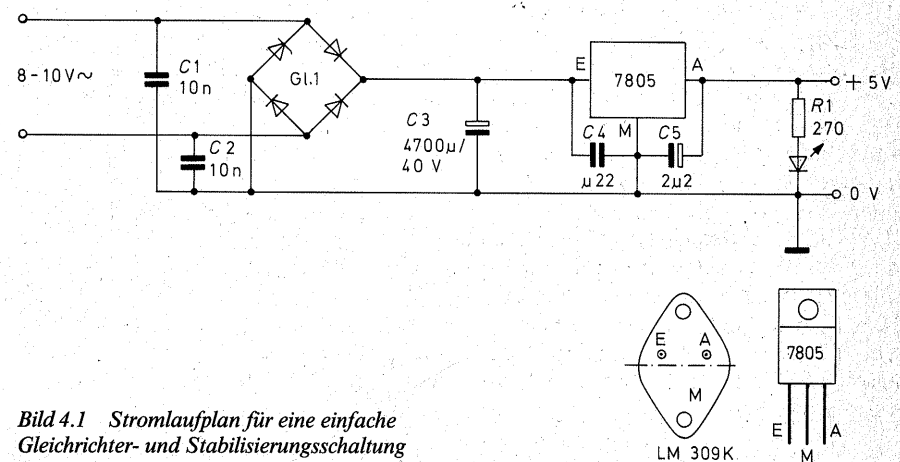
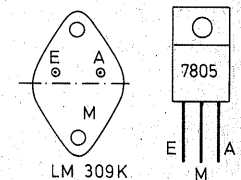


Bild 4.1 Stromlaufplan für eine einfache Gleichrichter- und Stabilisierungsschaltung



ist, daß er vollgekapselt und sein Netzanschlußkabel bereits berührungssicher montiert ist. Seine **Ausgangsspannung muß mindestens 7,5 V und darf höchstens 24 V** betragen. Wegen dieses weiten Spannungsbereichs muß der Lade-Elko C3 (in der Erwei-

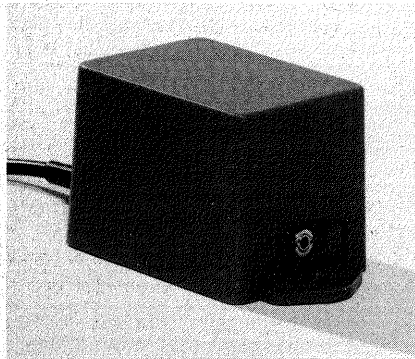
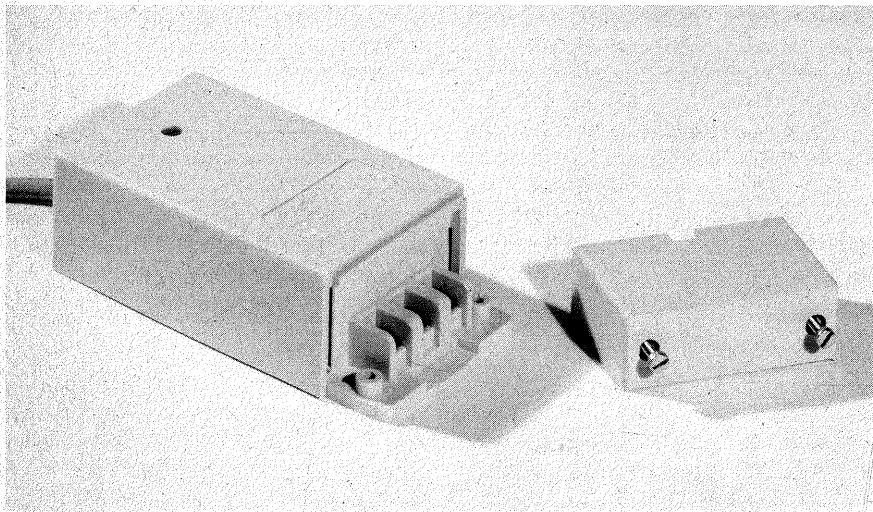


Bild 4.2 Vollgekapselter Experimentiertransformator

Bild 4.3 Vollgekapselter Klingeltransformator. Die Haube für die Sekundäranschlüsse ist abgenommen



terung auch C6, 7 und 8) für eine hohe Betriebsspannung vorgesehen sein, mindestens 35/40 V. Die Sekundärspannung des Transformators darf deswegen $24 V_{\text{eff}}$ nicht überschreiten, weil die Spannungsregler 7805 bzw. LM 309 K eine maximale Eingangsspannung von 35 V vertragen können. Da sich der Lade-Elko auf das $\sqrt{2}$ -fache der effektiven Wechselspannung abzüglich zweier Diodenschwellenspannungen auflädt, ist damit die Obergrenze erreicht:

$$24 \text{ V} \cdot 1,41 - 2 \cdot 0,6 \text{ V} \approx 33 \text{ V}$$

Manche Spielzeugtransformatoren liefern keine Wechselspannung, sondern sie enthalten bereits einen Gleichrichter und geben eine pulsierende Gleichspannung ab. Für die einfache Spannungsversorgung mit +5 V sind auch solche Transformatoren geeignet. Wenn es freilich darum geht, auch die -5 V für den AD-Wandler zu erzeugen, hilft nur ein Griff in die Trickkiste, z.B. der Bau eines kleinen Spannungswandlers.

Der Transformator sollte für einen **Ausgangsstrom von ca. 1 A** vorgesehen sein, damit sein Innenwiderstand nicht zu groß ist. Die **absolute Untergrenze** ist 0,5 A.

Klingeltransformatoren (Bild 4.3) sind bedingt geeignet, denn sie sind zu „weich“, d.h. ihr Innenwiderstand ist im Vergleich zu anderen Netztransformatoren groß. Dafür sind sie allerdings auch kurzschlußfest – ein beim Experimentieren nicht zu unterschätzender Vorteil. Klingeltransformatoren gibt es in den verschiedensten Ausführungen, z.B. 3,5,6,8 (12) V, für Ströme von 0,5 A, 0,63 A, 0,8 A, 1 A, 1,5 A oder 2 A. Eine 0,8-A-Ausführung (möglichst 12 V) reicht aus. Der Computer wurde mit den Bausteinen 1 (nach Bild 4.1), 2, 3, 4, 5, 6, 7 und 9 aus einem Klingeltransformator 8 V/0,5 A gespeist, das ging gerade noch. Bei vollem Betrieb zeigt VCC schon geringe Einbrüche von 40 bis 50 mV, die beim Anschluß jedes weiteren Verbrauchers (z.B. LED) größer wurden. Das ist ein sicheres Zeichen dafür, daß dieser Transformator überfordert wurde, und die Spannung schon vor dem Spannungsregler-IC zusammenbrach.

Für alle Klingeltransformatoren gilt, daß nur die Gleichrichter- und Stabilisierungsschaltung nach Bild 4.1 bzw. 4.6 in Frage kommt. Für die Thyristorschaltung nach Bild 4.7 sind sie nicht zu gebrauchen. Die Schaltung ist allerdings auch nicht nötig, weil die Sekundärspannung eines Klingeltransformators ja nicht so hoch ist (siehe Seite 74).

Sofern Sie einen Klingeltransformator verwenden, montieren Sie (oder lassen Sie es am besten gleich im Elektronikgeschäft erledigen) ein Netzkabel mit angegossenem Stecker. Auf der Leiterplatte (Bilder 4.8 und

4.9) ist Platz für einen Transformator der Baureihe 54 vorgesehen, z. B. den Typ 5408-1 (8 V/1,5 A) oder 5412-1 (12 V/1 A) oder 5412-2 ($2 \times 6 \text{ V}/1 \text{ A}$). Wenn Sie im Umgang mit der Netzspannung ganz sicher sind, können Sie auch einen dieser Transformatoren verwenden, sie sind in fast allen Elektronikläden zu haben. Diese Transformatoren sind mit Lötschaltern für Printmontage versehen. Im übrigen sind sie voll gekapselt. Auf der Oberseite der Leiterplatte erscheint daher keine Netzspannung. Die Lötstellen sollten Sie aber unbedingt mit Isolierband überkleben. Sichern Sie das Netzkabel durch eine Zugentlastungsschelle auf der **Grundplatte** und nicht nur an der Leiterplatte.

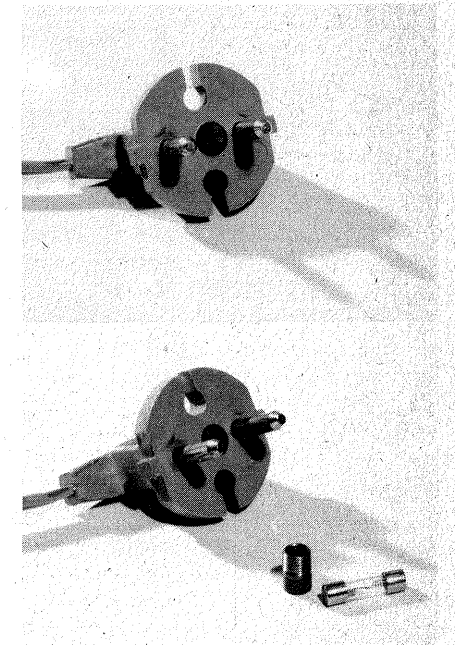


Bild 4.4 Netzstecker mit integriertem Sicherungshalter

Außerdem müssen Sie noch irgendwo eine Sicherung (0,15 AT) gut isoliert unterbringen. Früher (als bekanntlich alles viel besser war) gab es Netzkabel mit angegossenem Stecker, in den ein Sicherungshalter integriert war (Bild 4.4). Wenn Sie solch ein Netzkabel in einem Elektronikgeschäft sehen, greifen Sie zu! Es ist für Ihren Zweck ideal! An älteren Fernsehgeräten (Sperrmüll) sind diese Netzkabel oft zu finden.

Die Erzeugung der negativen Spannung für den AD-Wandler

Die Normalschaltung zur Erzeugung einer positiven und einer negativen Spannung zeigt Bild 4.5. Dazu ist ein Transformator mit zwei Sekundärwicklungen bzw. einer Sekundärwicklung mit einer Mittenanzapfung erforderlich. Spielzeug- oder Experimentiertransformatoren können diese Bedingung in der Regel nicht erfüllen. Darum helfen wir uns mit einem Kunstgriff, der freilich nur deswegen möglich ist, weil der AD-Wandler mit wenigen mA Strom auskommt: Über zwei spannungsfeste Elkos (C6, C7 in Bild 4.6) wird Wechselspannung ausgekoppelt; die Elkos dienen zur Gleichspannungstrennung. Die übrige Schaltung entspricht der Standard-

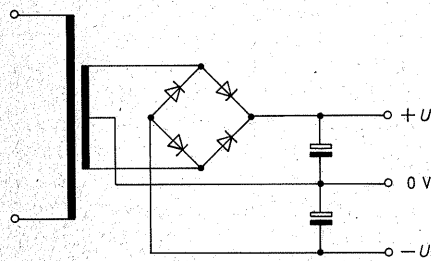


Bild 4.5 Standardschaltung zur Erzeugung einer positiven und einer negativen Spannung

schaltung zur Erzeugung der positiven Spannung, nur eben alles mit umgekehrtem Vorzeichen.

Ein Thyristorschaltnetzteil für Transformatoren mit hoher Sekundärspannung

Die Eingangsspannung U_{ein} am Spannungsregler-IC muß mindestens 2 V (besser 3 V) höher sein als die Ausgangsspannung U_{aus} . Die Differenz $U_{\text{ein}} - U_{\text{aus}}$ wird vom Spannungsregler „wegeregelt“; sie fällt am IC wie an einem Widerstand ab. Sobald ein Strom fließt, entsteht am Spannungsregler eine Verlustleistung, die, wie bei einem Ohmschen Widerstand, in Wärme umgesetzt wird.

Beispiel: $U_{\text{ein}} = 8 \text{ V}$, $U_{\text{aus}} = 5 \text{ V}$, also fallen am IC 3 V ab. Bei einem Strom von 0,5 A beträgt die Verlustleistung:

$$3 \text{ V} \cdot 0,5 \text{ A} = 1,5 \text{ W}$$

Diese Leistung muß als Verlustwärme über den Kühlkörper an die Umgebungsluft abgegeben werden. Verlustleistung und -wärme steigen mit der Eingangsspannung. Deswegen ist es wichtig, dafür zu sorgen, daß U_{ein} nicht unnötig hoch ist.

Beispiel: Es steht ein Transformator mit einer Sekundärspannung von 15 V_{eff} zur Verfügung. Die Ladespannung am Elko beträgt dann:

$$15 \text{ V} \cdot \sqrt{2} - 2 \cdot 0,6 \text{ V} \approx 20 \text{ V}$$

An einem 5-V-Spannungsregler müssen 15 V abfallen; dadurch entsteht bei einem Strom von 0,5 A eine Verlustleistung von 7,5 W, und die ist schon nicht mehr so leicht zu beherrschen, abgesehen davon, daß es eine ungeheure Energieverschwendung ist, für eine Nutzleistung von 2,5 W ($5 \text{ V} \cdot 0,5 \text{ A}$) 7,5 W in Wärme umzusetzen.

Wenn es gelingt zu verhindern, daß

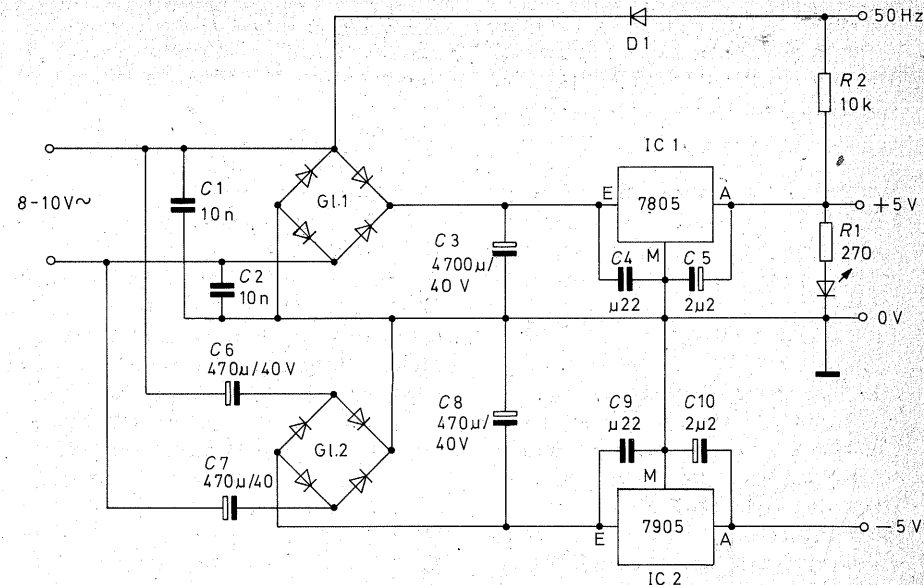
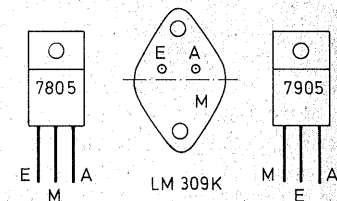


Bild 4.6 Erweiterung des Netzteils zur Erzeugung der negativen Spannung von -5 V



sich der Lade-Elko C3 über Gebühr auflädt, gibt es keine Wärmeprobleme, und außerdem wird noch Energie gespart.

Die Aufgabe, die Ladespannung von C3 zu begrenzen, übernimmt der Thyristor (Bild 4.7) zusammen mit dem Transistor und den ihn umgebenden Bauelementen.

Die Basisspannung U_B des Transistors wird mit der Zenerdiode festgehalten, und zwar auf:

$$U_B = U_{\text{aus}} + U_z = 5 \text{ V} + 3,9 \text{ V} = 8,9 \text{ V}$$

Solange die Emitterspannung (= Ladespannung von C3 + eine Diodenschwellenspannung vom Gate des

Thyristors) um 0,6 V niedriger ist als U_B , kann ein Basisstrom über den Transistor fließen. Der Transistor leitet und zündet den Thyristor. Der läßt eine oder mehrere positive Halbwellen durch, und C3 lädt sich auf. Die Ladespannung von C3 steigt. Sobald sie so hoch ist, daß die Emitterspannung des Transistors nicht mehr um die Schwellenspannung der Basis-Emitter-Diode des Transistors niedriger ist als U_B , sperrt der Transistor, und mit dem nächsten Nullpunkt der pulsierenden Gleichspannung sperrt auch der Thyristor, weil sein Haltestrom unterschritten ist. C3 kann sich nicht weiter aufladen.

Durch den Stromverbrauch des Gerä-

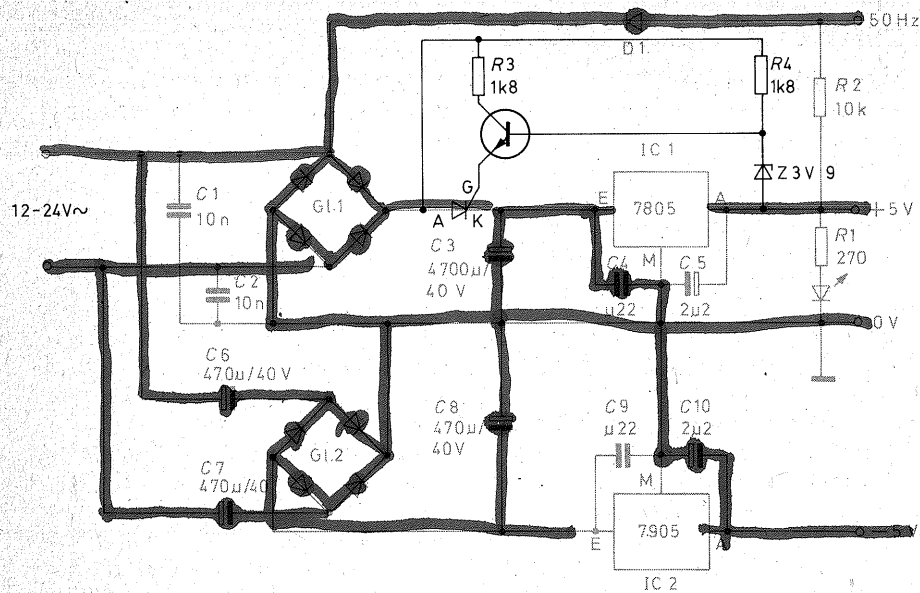


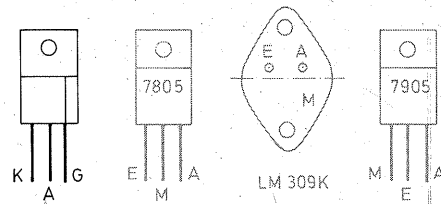
Bild 4.7 Die Thyristorschaltung verhindert eine übermäßige Aufladung von C3

tes entlädt sich C3 wieder. Die Ladespannung sinkt, bis der Transistor wieder leitet und den Thyristor zündet. Nun lädt sich C3 erneut auf, und der Lade- bzw. Entladezyklus wiederholen sich.

Die Ladespannung von C3 kann also durch die Thyristorschaltung nicht ins Unermessliche steigen. Wenn der Transformator eine Sekundärspannung von weniger als $12 V_{eff}$ abgibt, hat die Schaltung keinen Sinn. Wenn Sie aber einen Stelltransformator benutzen, etwa ein in den Schulen übliches Stromversorgungsgerät, der sich vielleicht sogar bis $24 V_{eff}$ einstellen läßt, sollten Sie auf die Thyristorschaltung nicht verzichten. Wie leicht

könnte es passieren, daß Sie Ihr Gerät einschalten, und der Transformator ist auf 24 V gestellt!

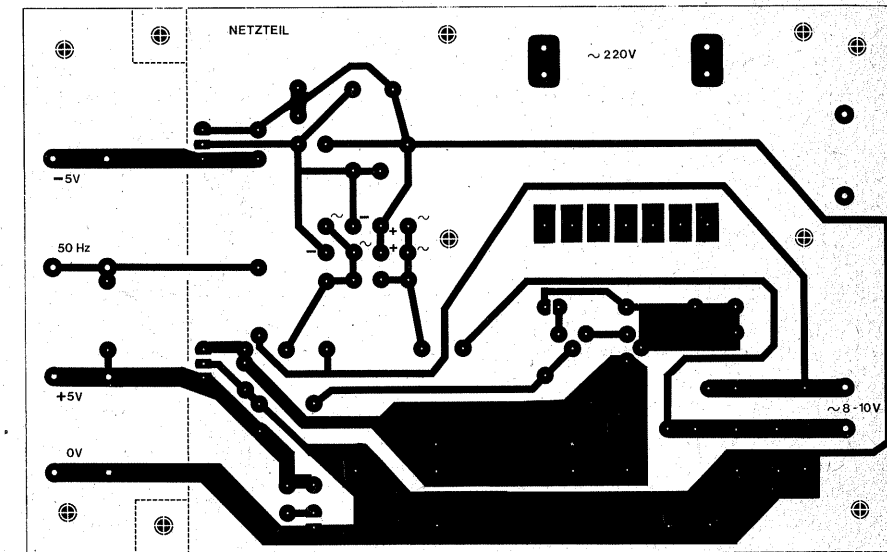
Die Gewinnung des 50-Hz-Taktes
Der 50-Hz-Ausgang liegt über den Widerstand R2 ($10 k\Omega$) auf H. Der Widerstand darf so groß sein, weil die TTL-Eingänge keinen nennenswerten H-Eingangsstrom aufnehmen (siehe Seite 64). Die Diode D1 läßt die negative Halbwelle der Wechselfspannung durch. Dadurch wird der 50-Hz-Ausgang auf L gezogen. Während der positiven Halbwelle sperrt D1, daher kann der H-Pegel die +5 V der Versorgungsspannung nicht überschreiten.



Hinweise zum Aufbau des Netzteiles

Die Leiterplatte (Bild 4.8) ist so entwickelt, daß verschiedene Bauformen der Gleichrichter oder auch Einzeldioden zu verwenden sind. Auch für den Elko C3 können Sie unterschiedliche Bauformen verwenden (Bild 4.9), Elkos mit axialen Drahtanschlüssen oder solche in Töpfchenform. Sie können auch zwei Elkos $2200 \mu F$ parallel schalten. Die Platine läßt Ihnen die Freiheit, sich nach den Vorräten Ihrer Bastelkiste zu richten. Achten Sie aber sorgfältig auf die Polung der Elektrolytkondensatoren! Statt des 7805 können Sie auch einen LM 309 K verwenden, um die positive Spannung zu erzeugen. Dieser Baustein kann einen größeren Strom regeln (ca. 1,5 A). Das könnte für Sie

Bild 4.8 Leiterplatte für das Netzteil, Kupferseite



dann wichtig werden, wenn Sie weitere Bausteine (z.B. eine Ampelschaltung) aus dieser Quelle speisen möchten. Die Wahl dieses Spannungsreglers hat aber nur dann einen Sinn, wenn Ihr Transformator einen entsprechend starken Strom liefern kann.

IC1 muß unbedingt gekühlt werden; IC2 benötigt wegen der geringen Stromentnahme keine Kühlung. Zugunsten der mechanischen Stabilität, die ja bei der offenen Bauweise wichtig ist, sollten Sie auch IC2 an dem Kühlblech festschrauben.

Das Kühlblech fertigen Sie sich am besten selbst (Bild 4.10). Sie sägen mit der Laubsäge (feines Metallsägeblatt) aus einem Stück Kupfer- oder Aluminiumblech der Größe 95×50 mm einen Streifen von 10×65 mm aus. Sie bohren Löcher mit $3,2$ mm \varnothing in die Füßchen und entgraten sie. Danach können Sie die Füßchen mit einer Flach- oder Kombizange abwinkeln. Entgraten Sie auch das gesamte

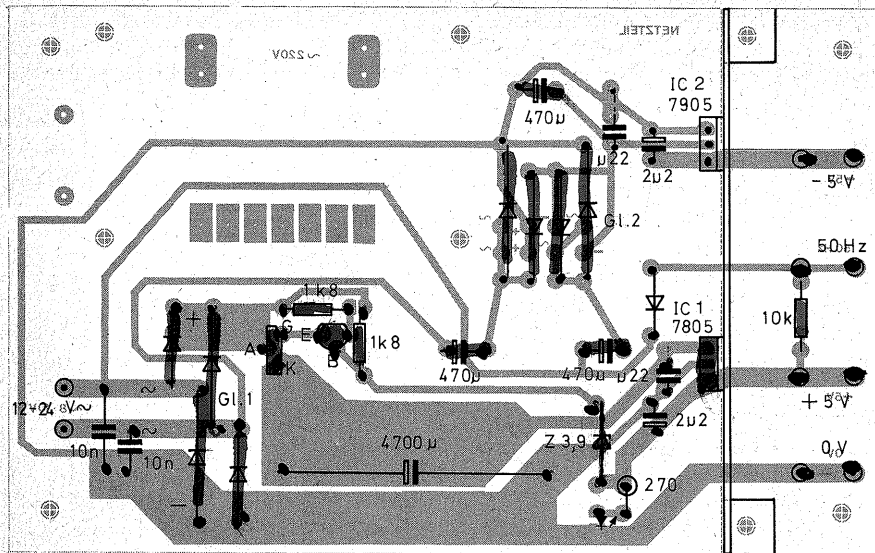


Bild 4.9 Bestückungsplan für das Netzteil, Bauteileseite

Kühlblech und runden die Ecken ab, damit Sie sich später nicht daran verletzen.

Nun können Sie Kühlblech und Spannungsregler montieren – übrigens die einzige feinmechanisch etwas kompliziertere Montagearbeit an dem ganzen Computer. Hier die Arbeitsschritte:

1. Kühlblech etwa 2 mm „hinter“ den Bohrungen für IC1 und IC2 an-

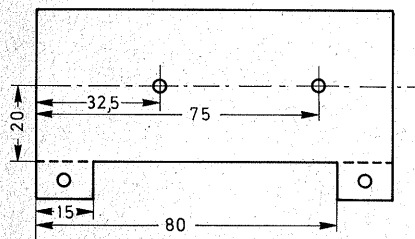


Bild 4.10 Das Kühlblech

schrauben (Bild 4.9), Schrauben M3 × 10, Kopf unter der Platine, Gewindeende und Mutter auf den Füßchen des Kühlblechs.

2. IC1 und IC2 so weit in die Platine einstecken, daß die Anschlüsse etwa 1 mm auf der Kupferseite vorstehen.

3. Bohrungen für die Befestigungsschrauben (Bild 4.11) auf dem Kühlblech anreißen.

4. Kühlblech wieder von der Leiterplatte lösen, Befestigungslöcher mit 3,8 mm Ø bohren und **sorgfältig entgraten**.

5. Kühlblech wieder gemäß Punkt 1 befestigen.

6. IC1 und IC2 an ihre Plätze stecken und gemäß der Explosionszeichnung (Bild 4.11) isoliert an das Kühlblech anschrauben. Die ICs sowie das Kühlblech eventuell vorher **dünn** mit Wärmeleitpaste einstreichen.

7. **Mit dem Ohmmeter prüfen, ob beide ICs auch wirklich vom Kühlblech isoliert sind.** Wenn nicht, die ICs lösen, einen eventuell doch noch vor-

handenen Bohrgrat entfernen, auf genaue Passung von Isoliernippel (3) und Isolierscheibe (5) achten.

8. IC-Anschlüsse anlöten.

Sofern Sie einen LM 309 K verwenden, zeichnen Sie sich die Bohrungen für die Anschlüsse E und A sowie für die beiden Befestigungsschrauben mit Hilfe der Isolierscheibe an, die ja diese Löcher maßgenau enthält. Die Bohrungen für E und A sollten einen Durchmesser von 2,5 mm haben, damit Sie Ungenauigkeiten ausgleichen können. Die Löcher für die Befestigungsschrauben bohren Sie mit 4 mm Ø, weil die Isoliernippel für die TO-3-Gehäuse dicker sind.

Das IC schrauben Sie so an, daß die Anschlüsse zur Platinenmitte zeigen. Dann können Sie kurze Drahtverbindungen zu den entsprechenden Bohrungen in der Leiterplatte führen.

Der Isoliernippel liegt auf dem Kühlblech auf und ragt in die Bohrung des LM 309 K hinein. Die Befestigungsschraube muß das IC-Gehäuse berühren, weil sie den Kontakt M herstellt (Bild 4.11).

Die Reihenfolge der Bestückung ist gleichgültig. Es ist aber praktisch, mit den kleinen Bauelementen wie Dioden und Widerständen zu beginnen.

Der Bestückungsplan (Bild 4.9) zeigt die volle Bestückung nach Bild 4.7. Wenn Sie einen Klingeltransformator oder einen anderen Transformator mit einer Sekundärspannung bis 12 V verwenden, entfällt die Thyristorschaltung. Sie lassen dann den Transistor, die Zenerdiode, R3, R4 sowie den Thyristor fort; an seine Stelle setzen Sie zwischen den Anschlüssen A und K eine Drahtbrücke ein. Die Bilder 4.12 und 4.13 zeigen Ihnen den fertigen Aufbau.

Die Verbindungen zum Transforma-

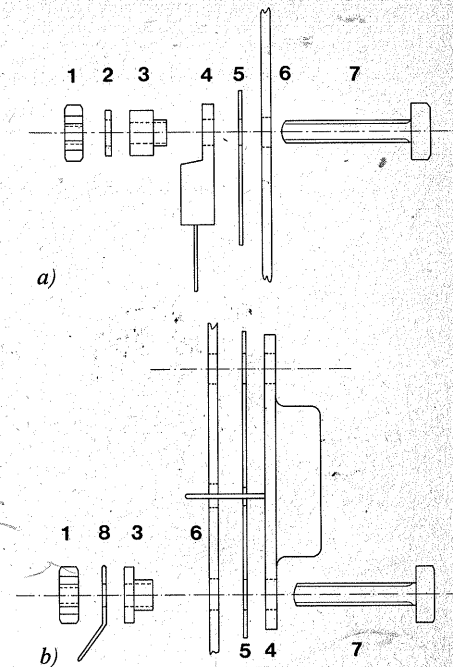


Bild 4.11 Isolierte Montage der Spannungsregler

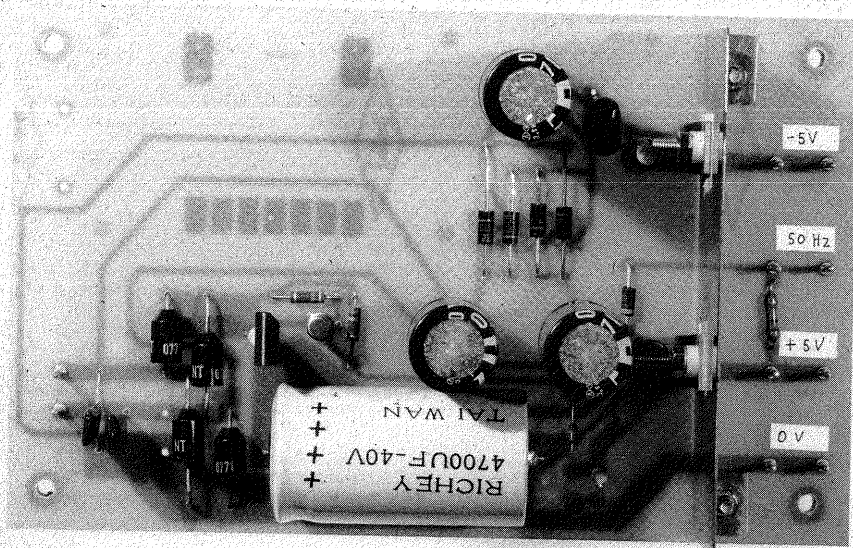
a) 7805/7905, b) LM 309 K

1: Mutter M 3; 2: Unterlegscheibe; 3: Isoliernippel; 4: IC; 5: Isolierscheibe; 6: Schraube M 3 × 10; 8: Lötöse

tor sowie zur Busplatte werden über steckbare Litzen hergestellt. Die Leitungen für 0 V (GND) und +5 V (VCC) sollten nicht gar zu dünn sein. Ein Querschnitt von 0,5 mm reicht aber aus.

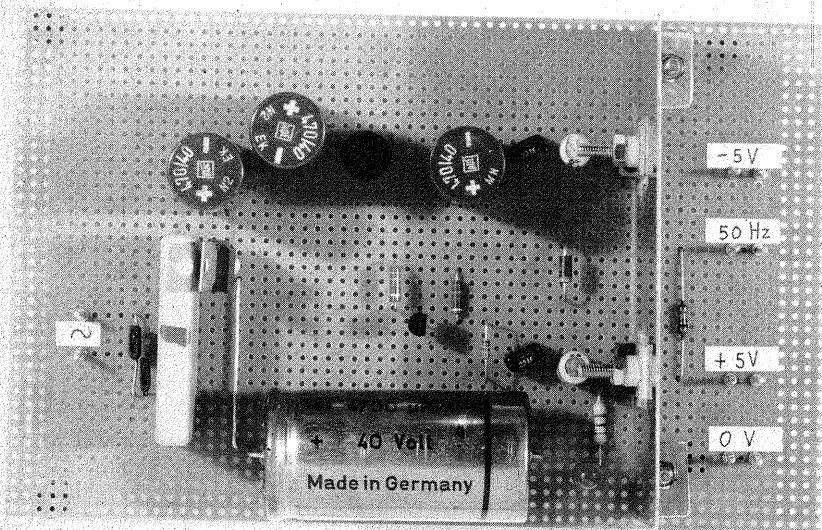
Die Steckschuhe sollten Sie isolieren. Dazu benötigen Sie pro Stück etwa 13 mm Fahrradventilschlauch:

1. Litzen abisolieren, verzinnen und verzinnendes Ende auf 1 bis 2 mm kürzen.
2. Steckschuhe auf die Stifte setzen und Anschlußfahnen verzinnen.
3. Zwei Enden Ventilschlauch auf die Litzen schieben.



Oben:
Bild 4.12 Das fertige Netzteil auf der geätzten Leiterplatte

Unten:
Bild 4.13 Das fertige Netzteil auf einer Experimentierplatte



4. Litze anlöten, und während die Lötstelle noch heiß ist, rasch das Ventilgummi über den Steckschuh schieben.

5. Ebenso den zweiten Steckschuh anlöten und isolieren.

Hinweise für den Aufbau auf einer Experimentierplatte

1. Unterbrechen Sie die Leiterbahnen rund um die Schrauben, mit denen Sie das Kühlblech befestigen. Dadurch erreichen Sie einerseits, daß das Kühlblech von allen Strompfaden isoliert ist, andererseits verhindern Sie auch unerwünschte Verbindungen von Leiterbahnen über die Schrauben oder das Kühlblech.

2. Die Verbindungen zwischen dem Gleichrichter G1, dem Thyristor und C3 müssen aus vergleichsweise dickem Schalt Draht (0,6 bis 0,8 mm Ø) bestehen. Eine Experimentierplattenleiterbahn, die ja wegen der vielen Bohrungen praktisch sehr schmal ist, reicht für die starken Ladestromstöße nicht aus.

Inbetriebnahme und Prüfung des Netzteiles

Ehe Sie die Transformatorspannung anlegen, unterziehen Sie Ihr Netzteil einer genauen Sichtprüfung:

Ist das Zinn an allen Lötstellen gut verlaufen?

Klebt irgendwo ein Zinnkügelchen? Sind alle Leiterbahnen in Ordnung? (Haarrisie und Kupfernadeln finden Sie am ehesten, wenn Sie die Platte gegen starkes Licht halten.)

1. Prüfung der Spannung +5 V (VCC)

Nach dem Anschließen der Transformatorspannung muß die LED aufleuchten. Messen Sie aber sicherheits halber nach, ob VCC wirklich +5 V beträgt (Minuskabel des Voltmeters an 0 V, Pluskabel an +5 V). Belasten Sie den Ausgang mit einem Glühlämpchen, z.B. 6,3 V/0,3 A. Die Spannung darf nicht merklich sinken.

Wenn sie aber doch sinkt:

Reicht die Spannung an C3 aus?

Wenn ja: Ist IC1 richtig angelötet? Auch C4 oder C5? Ohne diese Kondensatoren neigen die Spannungsregler zum Schwingen. Sie merken es daran, daß sie auch ohne Belastung heiß werden, und daß Sie ihnen praktisch keine Leistung entnehmen können. Sind die ICs auch wirklich vom Kühlblech isoliert?

Wenn die Spannung an C3 kleiner als +7 V ist:

Beträgt die Transformatorspannung mindestens 7 V?

Sind die Dioden von G1 bzw. der Brückengleichrichter richtig angeschlossen?

Sind die Elkos richtig gepolt? (Dem vollschwarzen Balken entspricht der Minuspol.)

Bei Benutzung der Thyristorschaltung:

Zündet der Thyristor vielleicht nicht? (A und K überbrücken, dann muß sich C3 aufladen.)

Wenn sich C3 durch Überbrücken des Thyristors auflädt, liegt der Fehler in der Thyristorzündung:

Ist der Thyristor richtig eingesetzt?

Überprüfen Sie auch den Transistor mit R3 und die Zenerdiode (Polung beachten!) mit R4.

Wenn Sie prüfen wollen, ob die Thyristorschaltung richtig arbeitet, messen Sie die Spannung an C3. Sie muß um etwa +8 bis +9 V pulsieren. Ohne einen zusätzlichen Verbraucher pulsiert sie so langsam, daß der Zeiger des Meßinstruments in einem gewissen Bereich folgen kann. Sobald Sie einen Verbraucher anschließen, z.B. das Glühlämpchen, pulsiert die Spannung schneller, so daß der Zeiger des Meßinstruments nicht mehr folgen kann. Er zittert nur noch geringfügig oder steht still.

Wird das Netzteil aus einem Transformator gespeist, dessen Sekundärspannung sehr hoch ist, 20 oder gar 24 V, so stellt sich trotz der Thyristorschaltung an C3 eine vergleichsweise hohe Spannung von +12 bis +15 V ein. Der Thyristor kann ja erst abschalten, wenn die Halbwelle, während der der Transistor gesperrt wird, wieder auf Null zurückgegangen ist. Bei hoher Transformatorspannung reicht dann der Ladestrom aus, um C3 bis auf +15 V aufzuladen. Auch deswegen bildet eine Transformatorspannung von 24 V die absolute Obergrenze.

2. Prüfung der negativen Spannung – 5 V

Verbinden Sie das Minuskabel Ihres Meßinstruments mit dem Ausgang – 5 V, das Pluskabel mit 0 V. Sie müssen – 5 V messen. Belasten Sie diesen Ausgang mit einem Widerstand zwischen 560 Ω und 1 k Ω . Die Spannung darf nicht zusammenbrechen.

Falls Sie nicht – 5 V messen, suchen Sie den Fehler analog zur obigen Beschreibung.

3. Prüfung des 50-Hz-Ausgangs

Wenn Sie über einen Oszillographen verfügen, schauen Sie sich an, ob der Ausgang pulsiert. Es genügt aber auch, eine Telefonhörkapsel oder einen Kopfhörer mit hohem Innenwiderstand an 0 V und den 50-Hz-Ausgang anzuschließen. Sie müssen ein lautes Brummen hören können. Schließen Sie außerdem Ihr Vielfachmeßinstrument an. Es zeigt den arithmetischen Mittelwert zwischen L und H an. Da die H-Zeiten länger dauern als die L-Zeiten, das Tastverhältnis aber im übrigen stark von der Höhe der Transformatorspannung abhängt, wird sich ein Meßergebnis zwischen +3 und +4 V einstellen. Wenn Sie eine Spannung von weniger als +5 V messen, ist alles in Ordnung; zeigt Ihr Meßinstrument dagegen deutlich mehr als +5 V an, so ist wahrscheinlich die Diode D1 falsch gepolt.

Kapitel 5

Die Busplatte (Baugruppe 2)

Für alle diejenigen, die noch keine oder nur geringe Übung im Löten haben, eignet sich die Busplatte als Anfangsstück am besten. Sie enthält sehr viele Lötstellen, die vergleichsweise weit auseinander liegen, und während des Lötens kann nichts auseinanderfallen, kurz: Sie ist das ideale Objekt, löten zu üben.

Schaltungstechnik

Die Busplatte besteht aus 31 parallelen Leiterbahnen, auf die in gewissen Abständen Buchsenleisten („Federleisten“) montiert sind, so daß jede Leitung mehrfach angezapft werden kann. Man kann die Steckplätze der Busplatte mit den Steckdosen in einem Haus vergleichen. Sie sind alle gleichwertig. Für die Funktion eines Radios, das Sie an eine Steckdose anschließen, ist es gleichgültig, ob Sie dazu eine Steckdose im Wohnzimmer oder eine in der Küche benutzen; wichtig ist nur, daß das Radio an die Strom führenden Leitungen angeschlossen ist. So sind auch die Steckplätze auf der Busplatte untereinander gleichwertig.

Die Busplatte faßt die wichtigsten Leitungen zusammen, an die alle

Baugruppen des Mikrocomputers angeschlossen werden (siehe auch Bild 3.1):

- Die Datenleitungen D0 bis D7, den **Datenbus**;
- die (bei unserem Computer) 15 Adreßleitungen A0 bis A14; den **Adreßbus**,
- die wichtigsten Steuerleitungen, den **Control-Bus** (to control, engl. steuern) sowie
- die Leitungen zur Spannungsversorgung.

Über die letztgenannten Leitungen spricht man im allgemeinen nicht, denn es gilt als selbstverständlich, daß jede Baugruppe mit ihren Betriebsspannungen zu versorgen ist. So unentbehrlich sie auch sind, als „Bus“ bezeichnet man die Betriebsspannung führenden Leitungen nicht. Nur die **Leitungsbündel**, auf denen **Signale** (Befehle, Daten, Adressen, Steuersignale) von einer Einheit zur anderen hin- oder hertransportiert werden, sind „Busse“.

Das Bussystem ist eines der wichtigsten Konstruktionsmerkmale der Computertechnik im Gegensatz zur festverdrahteten Logik, in der auf einer Leitung ein Datenfluß in nur einer Richtung vorgesehen ist. Möglich ist das Bussystem durch Schaltglieder mit Tri-State-Ausgängen.

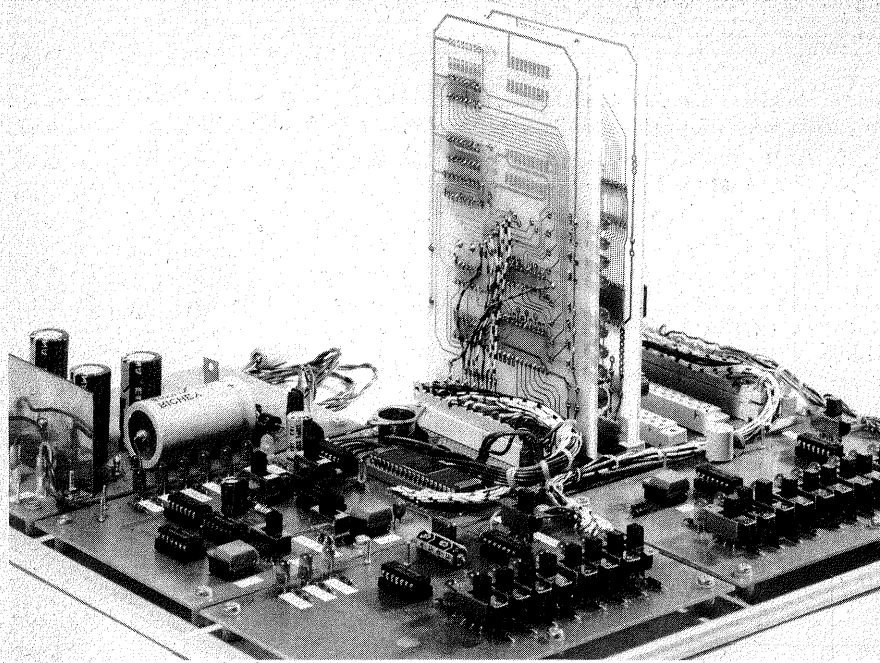


Bild 5.1 Busplatte mit eingesteckten Baugruppen

Es ist Aufgabe der Busplatte, den Computer flexibel zu machen, denn mit ihr lassen sich Baugruppen wie (zusätzliche) Speicher oder Peripherien beliebig anschließen oder abtrennen (Bild 5.1), sei es, um sie auszutauschen, zu verändern oder zu ergänzen – kurz: Die Busplatte erleichtert Ihnen das Experimentieren mit Ihrem Computer.

Als Steckleiste wurde die 31polige Leiste nach DIN 41617 ausgewählt. Die Zahl von 31 Polen ist nicht übermäßig groß; sie liegt nahe dem Minimum, das man für ein Computersystem braucht. Heutzutage sind schon 64polige Steckverbinder weit verbreitet, aber mit den 31 Polen kommen Sie zumindest für den Anfang gut

aus. Der Vorteil dieser Leiste ist, daß ihre Lötanschlüsse sowohl bei der Federleiste als auch bei der Stiftleiste (Bild 5.2) zwar in zwei Reihen liegen, jedoch gewissermaßen „auf Lücke“, so daß sich in Wirklichkeit ein einreihiges Rastermaß von 2,5 mm ergibt. Sie benötigen also nicht unbedingt eine geätzte Leiterplatte mit kompliziert umeinander herumgeführten Bahnen, Sie können auch eine Experimentierplatte verwenden.

Überdies sind sowohl Feder- als auch Stiftleisten sehr robust. Sie überstehen unbeschadet auch den rauen Schulbetrieb.

Die Anschlußbelegung (Bild 5.3) gilt selbstverständlich für sämtliche Baugruppen.

Hinweise zum Aufbau

Beim Kauf der Federleisten ist unbedingt darauf zu achten, daß diese für **Printmontage** gefertigt wurden, es gibt sie nämlich auch für Drahtanschluß (Bild 5.2). Diese Leisten sind ein gängiger Artikel, der eigentlich in jedem Elektronikladen zu haben ist. Wenn sie aber doch einmal ausgegangen sein sollten, hüten Sie sich vor Ratschlägen wie „Drücken Sie die Lötflächen platt, dann geht's auch.“! Es geht nicht! Entweder Leisten für Printmontage oder keine! Wozu gibt es schließlich die Konkurrenz!

Bild 5.4 zeigt die Kupferseite der Busplatte, Bild 5.5 die Bestückungsseite.

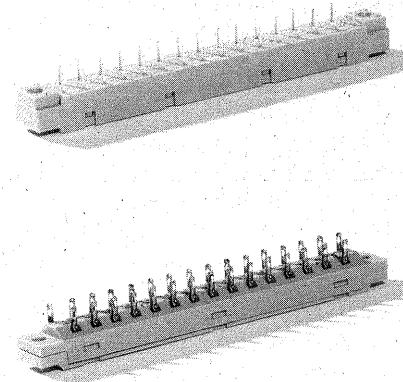


Bild 5.2 Federleiste a) für Printmontage, b) für Drahtanschluß, c) Stiftleiste

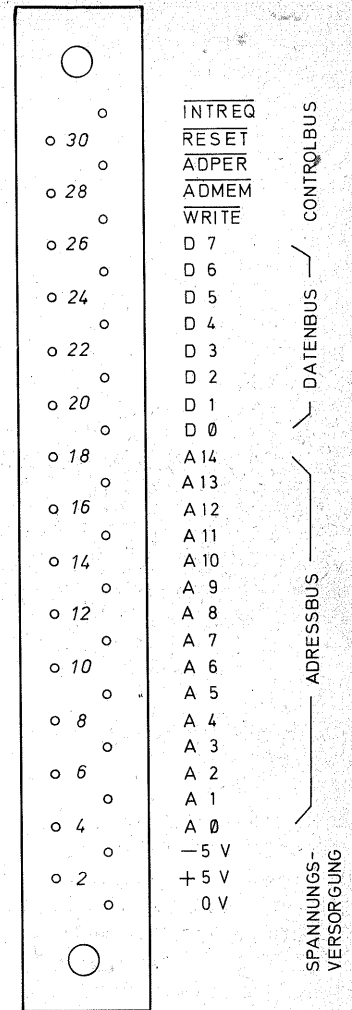


Bild 5.3 Anschlußbelegung der Busplatte

Wenn überhaupt eine Platte aus glasfaserverstärktem Epoxydmaterial bestehen sollte, dann diese, denn sie ist die einzige, die mechanisch durch das Einstecken oder Herausziehen der Stiftleisten wirklich belastet wird. Das Einsetzen der Federleisten erfordert etwas Feingefühl und Geduld.

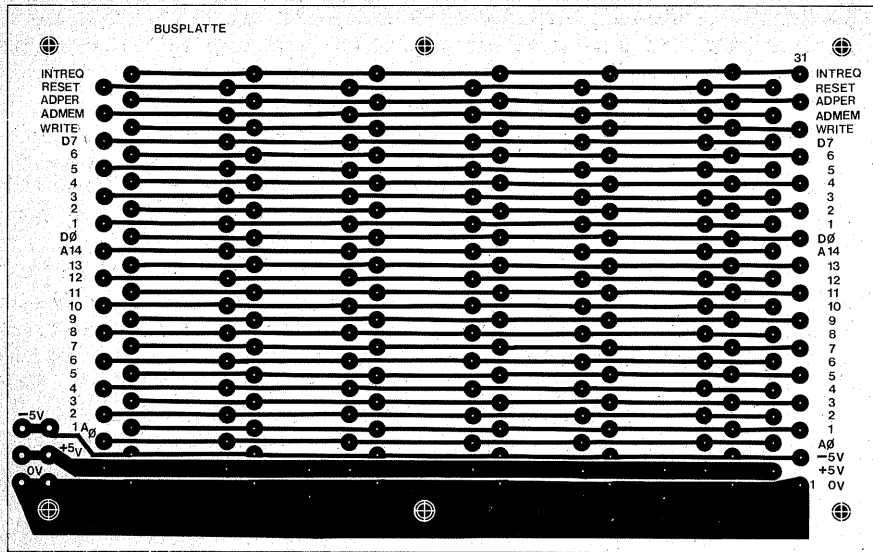


Bild 5.4 Kupferseite der Busplatte

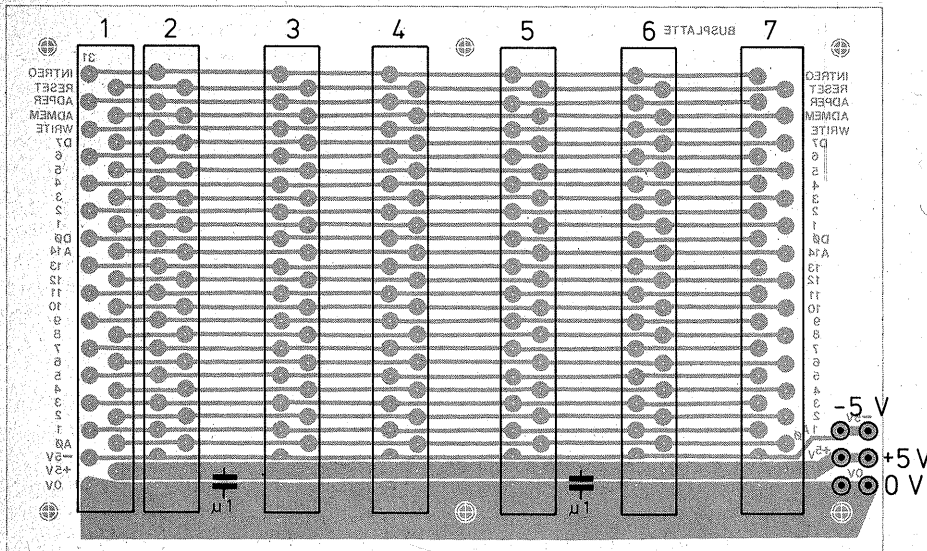


Bild 5.5 Bestückungsseite der Busplatte

Die auf Seite 22 (Bild 1.13) vorgestellte Pinzette mit den abgewinkelten Spitzen ist eine ideale Hilfe.

Zuerst sind die Lötstifte möglichst genau in ihren beiden Reihen auszurichten. Beim Einsetzen der Federleiste beginnen Sie am besten mit einer Reihe, z. B. den ungeradzahligen Lötstiften. Wenn diese Reihe durchgesteckt ist, folgt die zweite. Es ist unbedingt zu kontrollieren, ob auch sämtliche Lötstifte auf der Kupferseite durchschauen. Nur zu leicht knickt einer der dünnen Stifte um und kommt an der Lötseite nicht an. Dann ist der zugehörige Buchsenanschluß taub.

Es sei Ihnen gegönnt, bei diesem Hinweis nachsichtig zu lächeln; so etwas kann Ihnen nicht passieren. Es ist aber passiert und nicht nur ein einziger Mal; merkwürdige Funktionsausfälle hatten keinen anderen Grund, als daß ein Stift (oder gar mehrere!) sich beim Einsetzen der Federleiste umgelegt hatte und gar nicht an die Kupferbahn angelötet war. Ein bißchen Pingeligkeit kann Ihnen stundenlange Fehlersuche ersparen, denn wer rechnet im Ernstfall schon mit solch einem Fehler! Es ist leicht, sich beim Einsetzen der Federleiste etwas Zeit zur genauen Sichtkontrolle zu nehmen; eine Leiste auszulöten und hinterher wieder einzulöten, macht Ihnen viel mehr Mühe.

Die Federleiste muß **fest** auf der Leiterplatte aufliegen; es ist jedoch nicht nötig, daß Sie sie anschrauben. Wenn Sie sie aber anschrauben, dann vor dem Löten.

Zum Einstecken der Stiftleiste ist eine ziemlich große Kraft erforderlich. Liegt die Federleiste fest auf der Platine auf, so fängt diese die Kraft ab. Liegt sie jedoch nicht ganz dicht auf, so wird die Kraft auf die Lötstellen,

d. h. auf die Kupferbahnen übertragen, und wenn man Pech hat, reißen sie dann von der Platte ab, denn sie sind ja nur aufgeklebt.

Löten Sie zuerst einen Stift aus etwa der Mitte der Reihe an; verflüssigen Sie das Zinn nochmals, und drücken Sie nun mit einem Holzstab, z. B. dem Griff eines Tuschepinsels, dicht neben der Lötstelle **kräftig** auf die Leiterplatte (Bild 5.6). Wenn die Platte auch nur etwas gebogen ist, werden Sie merken, daß der Lötstift noch ein kleines Stückchen weiter aus dem Bohrloch herausrückt.

In gleicher Weise löten Sie auch an den Enden der Federleiste je zwei Anschlußstifte stramm an. Nun sitzt die Federleiste fest, und Sie können die übrigen Stifte anlöten.

Hinweise für den Aufbau auf einer Experimentierplatte

Die Leiterbahnen 1 (0 V) und 2 (+5 V) führen vergleichsweise starken Strom. Deswegen sind sie auf der Platine so breit wie möglich gehalten (Bild 5.4). Auf einer Experimentierplatte (Bild 5.9) müssen diese Bahnen verstärkt werden; dazu löten Sie einen 0,6 bis 0,8 mm starken (versilberten) Schaltdraht auf die beiden Leiterbahnen (Bild 5.8). Wenn Sie die Federleiste, wie oben beschrieben, in der Mitte und an den Enden anheften, so benutzen Sie nicht die Anschlüsse 1 und 2, sondern 3 und 4. Nach dem Einsetzen der Lötnägel in die Bahnen 1, 2 und 3 (versetzt anordnen, weil sie sonst zu dicht nebeneinanderstehen) und dem Anlöten der übrigen Anschlüsse legen Sie den gerecten und daher vollkommen geraden Schaltdraht dicht neben die Anschlußstifte der Federleiste auf die Leiterbahn, löten ihn **zusammen mit dem Anschlußstift** erst bei einer mitt-

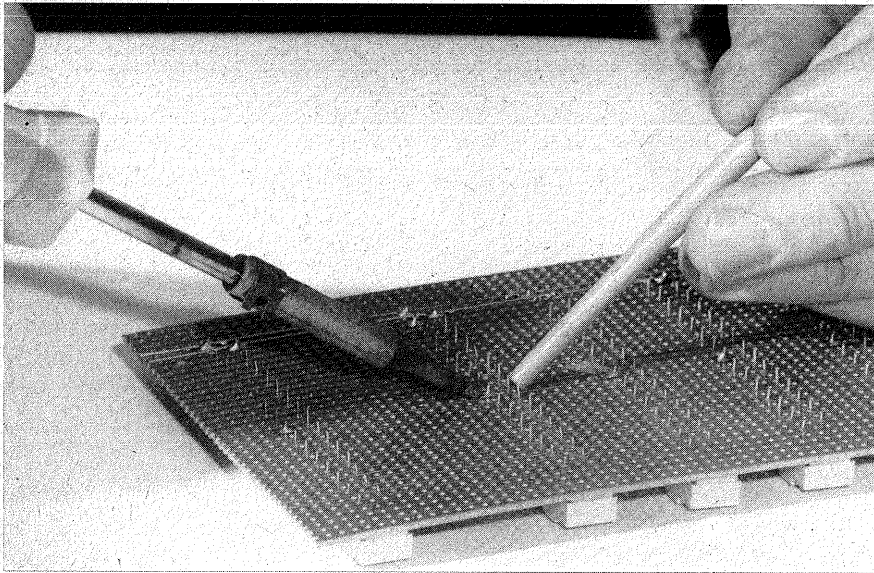


Bild 5.6 Andrücken der Leiterplatte

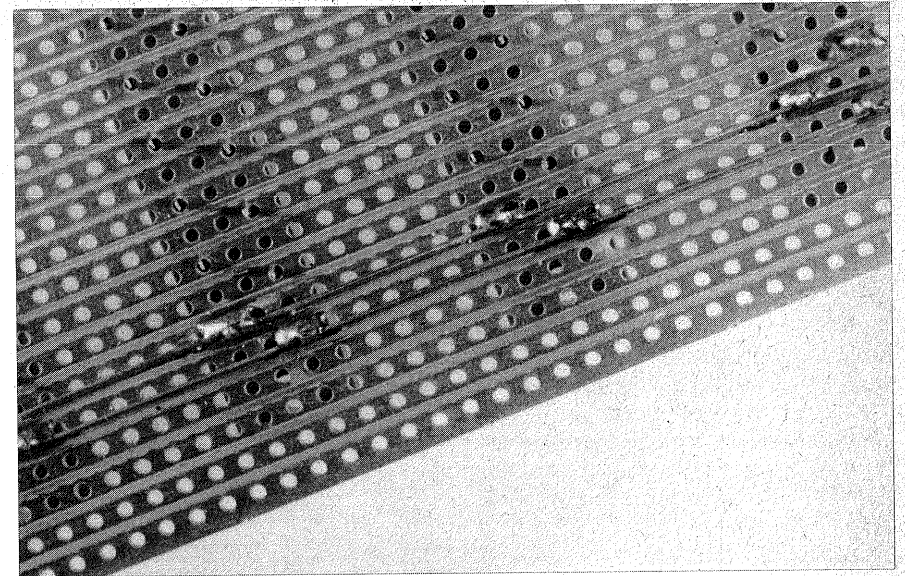


Bild 5.8 Drahtverstärkung der Leiterbahnen einer Experimentierplatte

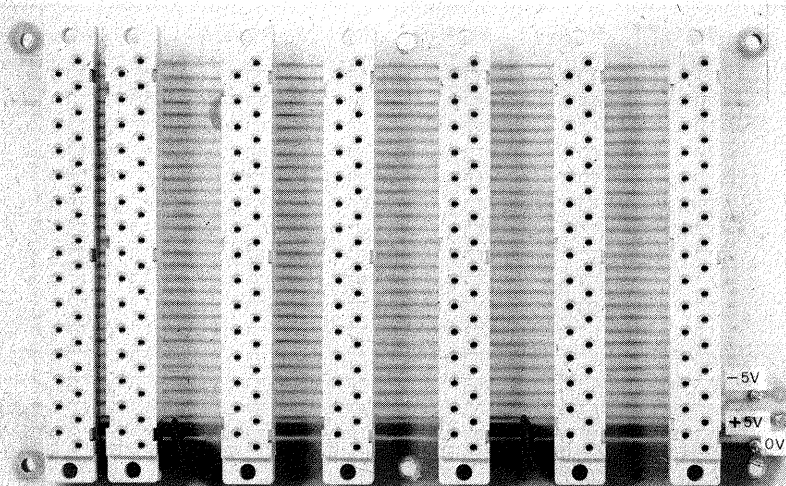


Bild 5.7 Die fertige Busplatte auf der
Printplatte

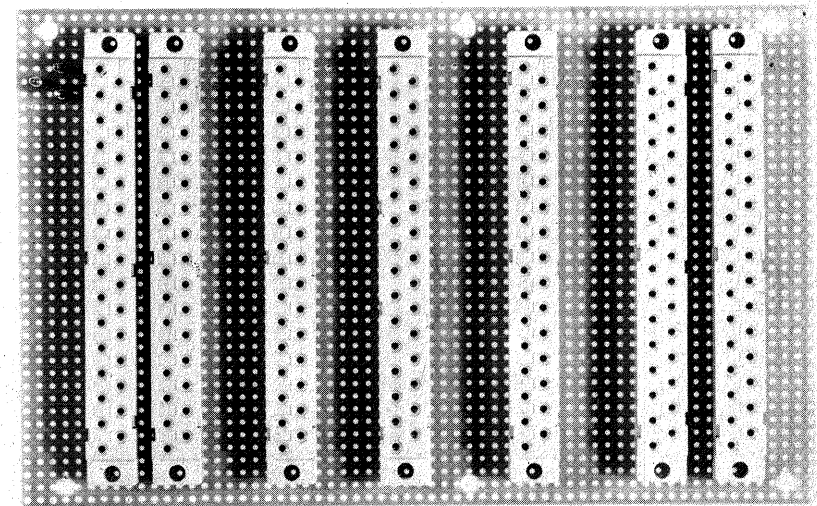
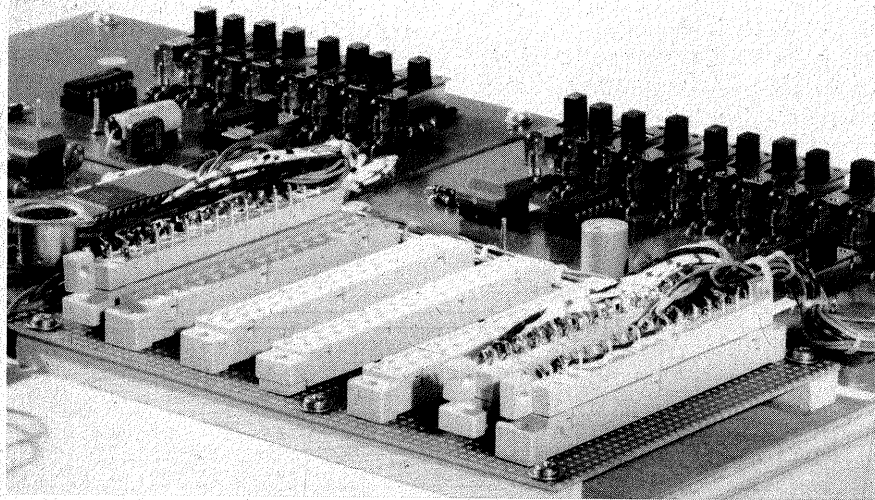


Bild 5.9 Die fertige Busplatte auf einer
Experimentierkarte

leren Leiste (z. B. Nr. 4) fest, dann an den Enden, einschließlich der Lötnägel, zum Schluß an den übrigen Federleisten. Diese Reihenfolge empfiehlt sich deshalb, damit die Wärmespannungen möglichst gering bleiben. Durch die Lötwärme dehnt sich der Draht (Sie werden es beim Anlöten an die letzten Federleisten gut bemerken können), und nach dem Erkalten zieht er sich zusammen. Wenn Sie ihn nach der Reihe der Federleisten anlöten, ist er für den erkalteten Zustand geringfügig zu kurz und zieht die Experimentierplatte krumm.

Die Experimentierplatte ist durch ihre zahlreichen Bohrungen sehr weich und für die mechanische Belastung fast zu schwach. Montieren Sie sie daher auf dünnen Holzleisten statt auf Distanzröllchen (Bild 5.10). Die Federleisten liegen dann mit ihren Enden auf den Leisten, und die Experimentierplatte braucht die Steckkraft nicht auszuhalten. Sie erreichen dadurch eine überraschende Stabilität Ihrer Busplatte.



Die Prüfung der Busplatte

Unterziehen Sie die Platte nach dem Löten einer genauen Sichtkontrolle (Lupe!). Etwaige Kurzschlüsse durch überschüssiges Lötzinn werden Sie schon auf diese Weise finden. Sie beseitigen diese am besten mit Entlötlitze (siehe Seite 23).

Danach testen Sie die Platte mit dem Ohmmeter (Vielfachinstrument, Meßbereich $R \times 1k$). Klemmen Sie mit kurzen Prüfkabeln zwei Stecknadeln an die Meßleitungen; stecken Sie eine davon in Buchse 1 der ersten Leiste, und tippen Sie mit der zweiten Nadel Buchse 1 der folgenden Leiste an. Das Ohmmeter muß Durchgang (0Ω) anzeigen, andernfalls ist eine der beiden Buchsen nicht richtig an-

Bild 5.10 Die Montage der Busplatte auf Leisten

gelötet. Fehlermöglichkeiten sind eine kalte Lötstelle, ein umgeknickter Anschlußstift oder eine Verschiebung der Federleiste um eine Leiterbahn (auf der Experimentierplatte leicht möglich). Anschließend prüfen Sie die Buchsen 1 aller Federleisten auf Durchgang. In gleicher Weise verfahren Sie mit den Buchsen 2 bis 31. Vergessen Sie nicht, auch die Lötnägel in die Prüfung einzubeziehen.

Eine weitere Fehlermöglichkeit sind Kurzschlüsse zwischen zwei benachbarten Leiterbahnen. Auch hier zeigt das Ohmmeter an, was dem Auge eventuell bei der Sichtprüfung entgangen ist:

Stecken Sie eine Nadel (eine Meßleitung) des Ohmmeters in Buchse 2 einer beliebigen Leiste, und tippen Sie mit der anderen Nadel nacheinander die benachbarten Buchsen 1 und 3 an. Das Ohmmeter muß $\infty\Omega$ (kein Zeigerausschlag) anzeigen. Wenn der Zeiger doch ausschlägt, liegt ein Kurzschluß vor und ist zu beseitigen. Nun prüfen Sie auf gleiche Weise von der Buchse 4 aus die Nachbarn 3 und 5 usw., bis Sie ganz sicher sind, daß sämtliche 31 Leiterbahnen voneinander isoliert sind.

Nun ist die Busplatte fertig und kann auf der Grundplatte montiert werden, dazu auch das Netzteil, sofern dies noch nicht geschehen ist (siehe Seite 25, Bild 1.18).

Schließen Sie jetzt die drei Leitungen vom Netzteil an – mit isolierten Steckschuhen (siehe Seite 77), und messen Sie die Spannungen an den Buchsen 1, 2 und 3. Ein falscher Anschluß hat später katastrophale Folgen.

1. Minusanschluß des Vielfachinstruments an Buchse 1, Plusanschluß an Buchse 2, Sie müssen +5 V messen.

2. Plusanschluß des Vielfachinstruments an Buchse 1, Minusanschluß an Buchse 3; Sie müssen –5 V messen.

3. Nochmalige Kontrolle: Plusanschluß des Vielfachinstruments an Buchse 2, Minusanschluß an Buchse 3; Sie müssen eine Spannung von 10 V messen (der Spannungsunterschied von +5 V nach –5 V beträgt 10 V).

Wenn alle Spannungen richtig auf der Busplatte vorhanden sind, fassen Sie diese Leitungen nicht mehr an.

Ersparnismöglichkeit

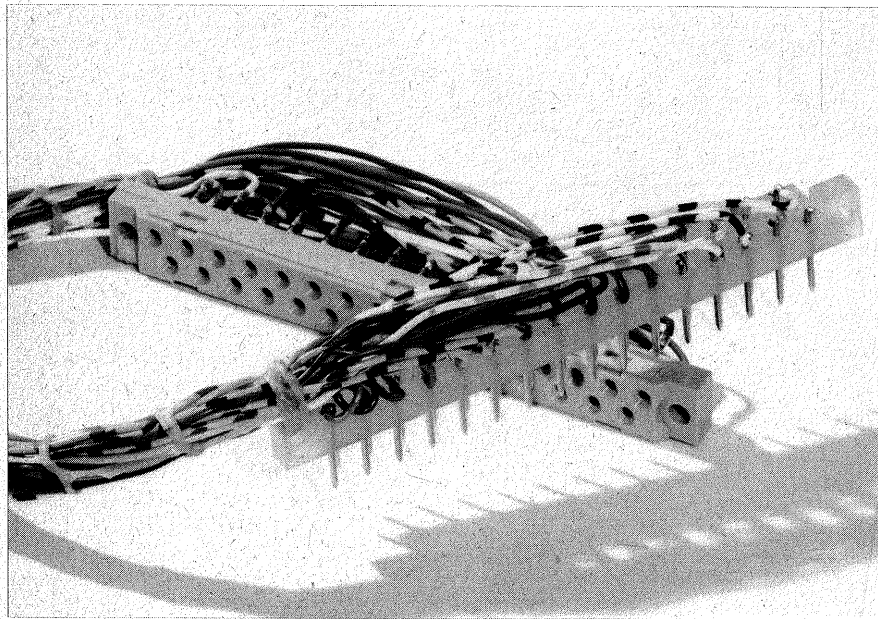
Abschließend noch ein Wort zu den Kosten: Die sieben Feder- und Stiftleisten sind nicht billig. Sie kosten zusammen gut 40 DM (1983). Wenn Sie Geld sparen wollen oder müssen, können Sie – zumindest eine lange Zeit – auf den Komfort der Steckleisten verzichten und die Baugruppen festverdrahtet anschließen.

Als Busplatte nehmen Sie dann eine Experimentierplatte und löten die Anschlußdrähte zwischen die vorgesehenen Steckplätze. Wenn Sie sich später einmal doch die Steckleisten anschaffen können, brauchen Sie nur die Anschlußdrähte dicht über der Platte abzuschneiden (Auslöten ist eine Quelle für viele Kurzschlüsse) und die Federleisten einzulöten. Zum Verdrahten nehmen Sie dünne, weiche Litzen (siehe Seite 16, Bild 1.2), damit sich beim Bewegen der Baugruppen keine starken Biegekräfte auf die Leiterbahnen übertragen.

Ein nützliches Hilfsmittel – der Busadapter

Für Experimente, bei der Fehlersuche und Reparaturen usw., empfiehlt sich ein Adapter. Der besteht aus einer Stiftleiste, an die über ca. 15 bis 20 cm lange weiche Litzen eine Federleiste angelötet ist (Bild 5.11). Er ist also eine bewegliche Verlängerung zur Busplatte. Wer öfter an Baugruppen messen oder etwas ändern muß, z. B. Leiter von Schülerarbeitsgruppen, sollte sich einen Adapter bauen. Über die flexiblen Adapterleitungen kann man mit der zu untersuchenden Platte bequem hantieren.

Bild 5.11 Der fertige Busadapter



Als Federleiste empfiehlt sich für diesen Zweck eine Ausführung für Drahtanschluß. Bei der Stiftleiste schneiden Sie die Lötanschlüsse dicht hinter der Biegung ab (Bild 5.12).



Bild 5.12 Kürzen der Lötanschlüsse der Stiftleiste

Die folgenden Hinweise zum Anlöten und Befestigen des Kabelbaums gelten auch für die übrigen Baugruppen Dateneingabe und -anzeige (4), CPU (3), Adreßeingabe und -anzeige (5) sowie Tastatur & Anzeige (9):

Löten Sie zuerst etwa 30 cm lange Litzen an die Federleiste, und notieren Sie sich die Zuordnung von Buchsen und Farbmarkierungen (Bild 1.3). Biegen Sie die Litzen an das Schraubenloch neben der Buchse 31, binden Sie den Kabelbaum dort fest und schnüren ihn auf die gewünschte Länge von ca. 15 cm weiter. Schneiden Sie das Bindegarn aber noch

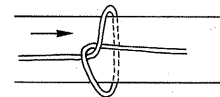


Bild 5.13 Einfacher Knoten für den Kabelbaum

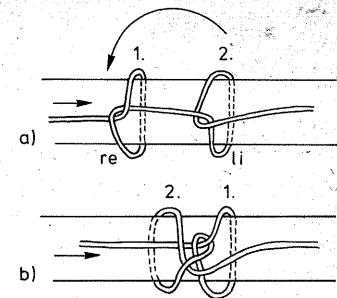
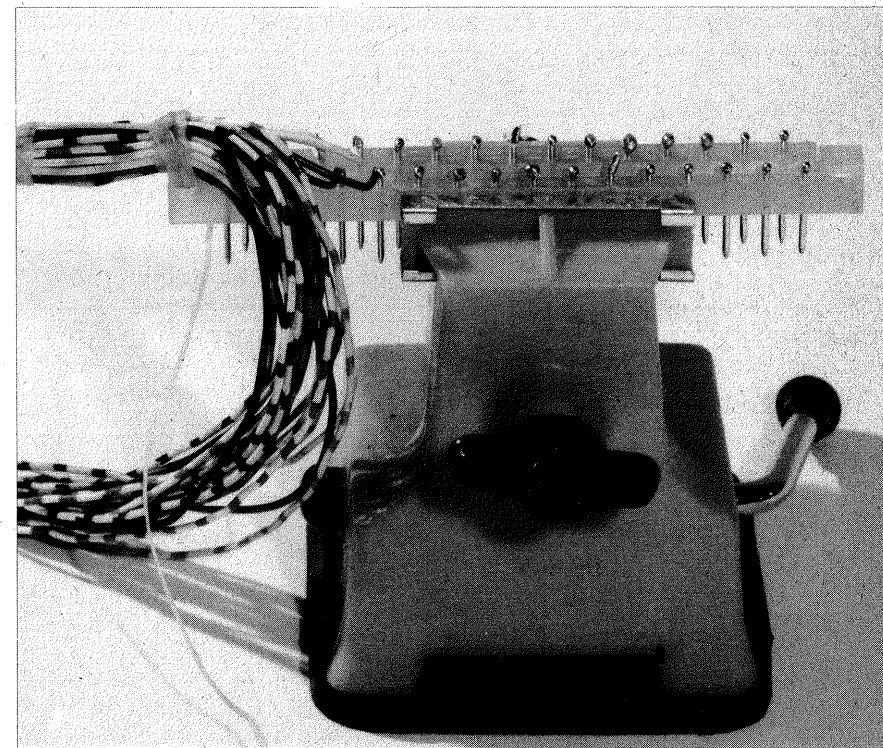


Bild 5.14 Doppelter Knoten für den Kabelbaum. Knoten 1 wird rechts herum geschlagen und festgezogen. Danach wird Knoten 2 links herum geschlagen (a), vor den Knoten 1 geschoben und erst dann festgezogen (b)

Bild 5.15 Anlöten und Anbinden eines Kabelbaums an die Stiftleiste



nicht ab, denn mit dem Ende soll der Kabelbaum ja noch an der Stiftleiste befestigt werden.

Es genügt, wenn Sie den Kabelbaum mit einem einfachen Knoten binden (Bild 5.13). Besser ist freilich der „doppelte“ Knoten, weil er sich auch bei vielem Bewegen des Kabelbaums nicht löst (Bild 5.14).

So weit entspricht der Arbeitsgang dem Anlöten der Litzen an die oben genannten Baugruppen.

Nach dem Kürzen der Lötanschlüsse der Stiftleiste (Bild 5.12) klemmen Sie die Leiste fest, z. B. in einem kleinen Schraubstock (Bild 5.15). Dann löten Sie von links (Stift 1) nach rechts die ersten Leitungen an, bis etwa Stift 6 oder 7. Jetzt binden Sie das gesamte Drahtbündel an dem Schraubenloch neben Stift 1 fest. Achten Sie darauf, daß die ersten Leitungen nicht stramm zur Anbindestelle führen, damit sie ein gewisses Spiel haben, falls an ihnen einmal kräftig gezogen werden sollte.

Nun löten Sie die übrigen Drähte an. Schneiden Sie jeden Draht eineinhalb Stiftabstände „hinter“ dem Stift ab, an den Sie ihn anlöten wollen. Führen Sie z. B. den Draht, den Sie an Stift 20 löten wollen, an diesem Stift vorbei und schneiden ihn zwischen den Stiften 21 und 22 ab. Dann isolieren Sie ihn kurz (1 bis 2 mm) ab, verzinnen ihn und löten ihn an den Stift. Abschließend prüfen Sie die richtige Zuordnung von Buchsen und Stiften mit dem Ohmmeter.

Kapitel 6

Die Dateneingabe und -anzeige (Baugruppe 3)

Einem jeden Computer müssen Befehle und Daten in „seiner Sprache“ angeboten werden, also als ein Muster von Nullen und Einsen. Das ist auch bei den Computern nicht anders, die über eine Schreibmaschinentastatur verfügen und denen alle Informationen über eine höhere Programmiersprache oder als Klartext angeboten werden. Alle Eingaben müssen dem Computer, vom Taschenrechner bis zum Großcomputer, in Muster von Nullen und Einsen übersetzt werden. Umgekehrt kann er die Ergebnisse seiner Arbeit auch nur als Muster von Nullen und Einsen von sich geben, und diese müssen wiederum in unsere gewohnten Codes, z. B. Buchstaben oder Ziffern, umgesetzt werden. Diese Übersetzungsarbeit erfordert oft einen gewaltigen Aufwand.

Das Dualsystem

Eine Ordnung in die Ketten aus Nullen und Einsen ergibt sich durch das **duale Zahlensystem** (duo, lat. zwei), das *Gottfried Wilhelm Leibniz*

(1646–1716) entwickelte, nicht zuletzt deswegen, um sich selbst das Rechnen, vornehmlich das Dividieren, zu erleichtern. Das Dualsystem unterscheidet sich von unserem gewohnten dekadischen Zahlensystem (déka, griech. zehn) nur durch die Basiszahl, d. h. die Anzahl der Möglichkeiten, die eine Stelle annehmen kann.

Am einfachsten finden Sie sich wohl durch einen Vergleich mit dem dekadischen System in das Dualsystem hinein. Jede Stelle einer Zahl hat einen bestimmten Wert, so unterscheiden wir von rechts nach links gelesen Einer, Zehner, Hunderter, Tausender usw. Die Zahl „4712“ bedeutet bei genauer Betrachtung

$$\begin{array}{cccc} 4 & 7 & 1 & 2 \\ 4 \cdot 10^3 & + 7 \cdot 10^2 & + 1 \cdot 10^1 & + 2 \cdot 10^0 \\ 4 \cdot 1000 & + 7 \cdot 100 & + 1 \cdot 10 & + 2 \cdot 1 \end{array}$$

Beim Lesen der Zahl werden die Stelleninhalte addiert. Der Stelleninhalt ergibt sich aus der Multiplikation des Stellenwerts mit der darin stehenden Ziffer.

Die Anzahl der möglichen Inhalte, die eine bestimmte Stelle haben kann, ist die **Basiszahl** (B) des jeweiligen Zahlensystems. In unserem gewohn-

ten Zehnersystem kann jede Stelle 10 unterschiedliche Inhalte haben, nämlich die Ziffern 0 ... 9. Daher ist die Basiszahl in diesem Falle „10“.

Die Stellenwerte sind die Potenzen dieser Basiszahl. Sie beginnen bei B^0 – in unserem gewohnten Zehnersystem also 10^0 . Jede Zahl hoch Null ist gleich 1. Daher stehen – sofern wir nur ganze Zahlen betrachten – am Anfang jedes Zahlensystems „Einer“. Von rechts nach links steigen die Potenzen kontinuierlich an: $10^1 = 10$ (Zehnerstelle); $10^2 = 10 \cdot 10 = 100$ (Hunderterstelle); $10^3 = 10 \cdot 10 \cdot 10 = 1000$ (Tausenderstelle) usw.

Genauso ist das Dualsystem aufgebaut, lediglich mit dem Unterschied, daß die Basiszahl „2“ ist, denn es gibt genau zwei Möglichkeiten, eine Stelle zu belegen: mit „0“ oder mit „1“. Wegen der anderen Basiszahl ergeben sich andere Stellenwerte:

$$\begin{aligned} 2^0 &= 1 \\ 2^1 &= 2 \\ 2^2 &= 2 \cdot 2 = 4 \\ 2^3 &= 2 \cdot 2 \cdot 2 = 8 \\ 2^4 &= 2 \cdot 2 \cdot 2 \cdot 2 = 16 \\ 2^5 &= 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 = 32 \\ 2^6 &= 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 = 64 \\ 2^7 &= 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 = 128 \end{aligned}$$

Die folgende Tafel (6.1) gibt die Gegenüberstellung von dualen Zahlen und dekadischen Zahlen bis 2^3 an. Soweit muß man die Zahlen unbedingt auswendig wissen, ebenso ihre hexadezimalen Äquivalente (siehe Hexadezimalsystem).

Beim **Decodieren** einer Dualzahl in unsere normalen Zehnerzahlen werden die Stelleninhalte addiert; ein Stelleninhalt kann nur „0“ oder der Stellenwert sein, weil nur die Ziffern „0“ oder „1“ als Faktoren in den Stellen möglich sind.

Stellenwert	2^3	2^2	2^1	2^0	dekad. Zahl	Hex-Zahl
	0	0	0	0	0	0
	0	0	0	1	1	1
	0	0	1	0	2	2
	0	0	1	1	3	3
	0	1	0	0	4	4
	0	1	0	1	5	5
	0	1	1	0	6	6
	0	1	1	1	7	7
	1	0	0	0	8	8
	1	0	0	1	9	9
	1	0	1	0	10	A
	1	0	1	1	11	B
	1	1	0	0	12	C
	1	1	0	1	13	D
	1	1	1	0	14	E
	1	1	1	1	15	F

Bild 6.1 Gegenüberstellung der Dualzahlen bis 2^3 und der ihnen entsprechenden dekadischen Zahlen und der Hexadezimalzahlen

Beispiel:

$$\begin{array}{cccccccc} 2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 128 & + & 16 & + & 8 & + & 2 & + & 1 = 155 \end{array}$$

Zum Umsetzen einer dekadischen Zahl in eine Dualzahl gibt es verschiedene Verfahren. Das einfachste ist das wiederholte Dividieren durch 2; es macht die Potenzierung der Basiszahl 2 gleichsam rückgängig. Beim Dividieren entsteht ein Quotient („Ergebnis“) und ein „Rest“. Der Rest kann 0 sein (bei einer geraden Zahl geht die Teilung durch 2 auf) oder 1 (ungerade Zahl). Man beginnt damit, daß man die umzusetzende Zahl durch 2 dividiert; den Rest (0 oder 1) schreibt man in die rangniedrigste Stelle, den entstandenen Quotienten dividiert man wieder durch 2, schreibt

den Rest in die nächsthöhere Stelle, dividiert den eben entstandenen Quotienten wieder, schreibt den Rest auf usw., so lange, bis eine weitere Division nicht mehr möglich ist, weil die Zahl 0 erreicht ist.

Beispiel: Die Zahl 75 soll in eine Dualzahl umgesetzt werden:

Division	Quotient	Rest
75:2	37	1
37:2	18	1
18:2	9	0
9:2	4	1
4:2	2	0
2:2	1	0
1:2	0	1

Bild 6.2 Umrechnung dekadischer Zahlen in Dualzahlen

Die hexadezimale (sedezimale) Schreibweise

Der Datenbus unseres und sehr vieler anderer Mikroprozessoren besteht aus acht Leitungen. Diese Leitungen realisieren die Stellen 2^0 bis 2^7 . Der „Name“ der Datenleitung gibt dabei die Zweierpotenz der Stelle an; D0 entspricht also der Stelle 2^0 , D1 der Stelle 2^1 , ... D7 der Stelle 2^7 . Eine Stelle für sich allein heißt **Bit** (binary digit, engl. Zweierzahl). Die acht Stellen zusammen ergeben 1 **Byte**. Was immer man auf dem Datenbus betrachtet, Befehle, Daten usw., es ist ein Muster aus acht Nullen oder Einsen. Diese Kette ist unhandlich; beim Eingeben oder Lesen, beim Notieren usw. ist sie sehr fehlerträchtig. Um 1 Byte besser handhaben zu können, ist es üblich, das Byte in 2 **Nibbles**

(nibble, engl. kleiner Bissen) von je 4 Bit zu unterteilen. Das Nibble wird als Zahl betrachtet; es kann 16 verschiedene Inhalte haben (Tabelle 6.1). Zum Beschreiben der Inhalte reichen die zehn Ziffern 0 bis 9 nicht aus, daher behilft man sich mit den Großbuchstaben A bis F. Der dekadische Inhalt

- 10 wird mit A bezeichnet,
- 11 mit B
- 12 mit C
- 13 mit D
- 14 mit E und
- 15 mit F (siehe Tabelle 6.1).

Um 1 Byte zu notieren und später auch mit der Tastatur einzugeben oder mit den Siebensegmentanzeigen zu lesen, genügen dann zwei Stellen:

$$\begin{array}{r} 1001 \ 0111 \\ \hline A \quad 7 \end{array}$$

Auf diese Weise ergibt sich ein Zahlensystem mit der Basiszahl „16“. Daher heißt es **Hexadezimalsystem** (hexadecim, griech./lat. sechzehn) oder **Sedezimalsystem** (sedecim, lat. sechzehn). Der Ausdruck Hexadezimalsystem darf Sie aber nicht zu dem Schluß verleiten, der Prozessor rechnet in diesem System, er rechnet dual. Die hexadezimale **Schreibweise** macht uns nur die Kolonnen aus Nullen und Einsen übersichtlicher.

Die rechte (rangniedrigste) Stelle enthält „Einer“ ($16^0 = 1$), die zweite Stelle von rechts „Sechzehner“ ($16^1 = 16$). Bei Adreßangaben werden 2 Bytes benötigt; auf diese Weise entstehen ein 3. und ein 4. Nibble. Die dritte Stelle von rechts enthält „Zweihundertsechundfünfziger“ ($16^2 = 256$), die vierte Stelle von rechts „Viertausendsechundneunziger“ ($16^3 = 4096$).

Beim Umrechnen in unser gewohntes Zehnersystem werden wieder die

Stelleninhalte addiert. Diesmal ist aber der Stellenwert mit der darin stehenden Ziffer vorher zu multiplizieren, genauso wie beim Zehnersystem. Das Beispiel A7 bedeutet:

$$A \quad 7 \\ 10 \cdot 16 + 1 \cdot 7 = 167$$

Drei- oder vierstellige Hexadezimalzahlen werden Sie nicht umzurechnen brauchen. Die Arbeit überlassen Sie ruhig Ihrem Computer (siehe Seite 274). Ein einziges Byte umzurechnen, ist nicht schwer, denn das 1×16 beherrschen Sie ja schon seit der Grundschulzeit. Trotzdem sei Ihnen die folgende Tafel als Hilfe gegeben:

Hex-Ziffer	16^3	16^2	16^1	16^0
0	0	0	0	0
1	4096	256	16	1
2	8192	512	32	2
3	12288	768	48	3
4	16384	1024	64	4
5	20480	1280	80	5
6	24576	1536	96	6
7	28672	1792	112	7
8	32768	2048	128	8
9	36864	2304	144	9
A	40960	2560	160	10
B	45056	2816	176	11
C	49152	3072	192	12
D	53248	3328	208	13
E	57344	3584	224	14
F	61440	3840	240	15

Bild 6.3 Dekadischer Inhalt der Hexadezimalstellen

Wollen Sie dagegen eine dekadische Zahl in das Hexadezimalsystem umsetzen, so können Sie das in Tafel 6.2 dargestellte Verfahren anwenden, nur mit dem Unterschied, daß Sie statt durch 2 durch 16 dividieren.

Die Unterscheidung der Zahlensysteme

Inzwischen haben wir es mit drei Zahlensystemen zu tun. Zur Unterscheidung erhalten die Zahlen daher ihre Basiszahl als Index, „2“ für das Dualsystem, „10“ für das Zehnersystem und „16“ für das Hexadezimalsystem. Wie wichtig diese Unterscheidung ist, zeigt folgende Gegenüberstellung:

- 11_2 bedeutet „drei“
- 11_{10} bedeutet „elf“
- 11_{16} bedeutet „siebzehn“.

Beim Lesen von Zahlen geht man, wenn man ganz korrekt sein will, in der Aufzählung der Stellen von rechts nach links vor. Reste dieser Korrektheit haben sich auch in der deutschen Sprechweise des Zehnersystems noch erhalten, denn wir lesen 23_{10} als „dreiundzwanzig“. Bei höheren Zahlen gilt diese Regel in der Umgangssprache freilich nicht mehr; sie ist auch nicht nötig, weil wir den Stellenwert mitnennen, z.B. lesen wir 8523_{10} als „achttausendfünfhundertdreiundzwanzig“.

Bei anderen Zahlensystemen gibt es diese Sprachhilfe nicht. Daher liest man 1011_2 als „eins-eins-null-eins“. Die Reihenfolge der gelesenen Ziffern gibt ihren Stellenwert an. Soweit die Theorie, die Praxis sieht dagegen anders aus. Niemand sollte sich zu dem Unsinn verleiten lassen, 10_2 als „zehn“ oder 15_{16} als „fünfzehn“ zu lesen.

Der gewohnten Leserichtung von links nach rechts zuliebe weicht man aber von der oben dargestellten Leseweise ab. Da man es in der Mikroprozessortechnik wegen der vorhandenen Leitungen des Datenbusses immer mit einer genau definierten Stellenzahl zu tun hat und der Stellenwert

somit festliegt, kann man es sich leisten, die Stellen von links nach rechts aufzuzählen: 0011_2 heißt dann „null-null-eins-eins-null-eins-null-eins“.

Auch Hexadezimalzahlen, kurz Hex-Zahlen genannt, werden üblicherweise von links nach rechts stellenweise gelesen:

- 12_{16} als „eins-zwei“
- $A7_{16}$ als „A-sieben“

Die Rechenregeln

Unser Mikroprozessor arbeitet grundsätzlich mit allen acht Stellen des Datenbusses (siehe Seite 95). Bei kleinen Zahlen sind die vorangehenden Stellen gleich 0. In den folgenden Rechenbeispielen wurden die vorangehenden Nullen zugunsten der besseren Übersicht weggelassen.

Die Rechenregeln des Dualsystems unterscheiden sich grundsätzlich nicht von denen des dekadischen Systems – es ist alles nur viel einfacher.

Die Addition

Bei der Addition zweier Zahlen gibt es nur folgende Möglichkeiten:

- $0+0=0$
- $0+1=1$
- $1+0=1$
- $1+1=10$
- $1+1+1=11$

Die letztgenannte Möglichkeit kommt dann in Frage, wenn ein Übertrag berücksichtigt werden soll. Die schriftliche Addition gleicht der Ihnen bekannten Form:

	dekadisch	dual
	27	11011
	+ 14	+ 1110
Überträge	1	1111
	41	101001

Sind mehrere Zahlen zu addieren, so werden sie einzeln nacheinander (seriell) addiert.

Die Subtraktion

Die Regeln für die (schriftliche) Subtraktion gelten so, wie sie Ihnen aus dem dekadischen System bekannt sind, gleichgültig, ob Sie nach dem Ergänzungsverfahren oder dem Umtauschverfahren (unzutreffend als „Borge-Verfahren“ bezeichnet) rechnen.

Beispiel:

	dekadisch	dual
	61	111101
	- 19	- 10011
Überträge	1	1
	42	101010

Sollen mehrere Zahlen subtrahiert werden, so werden sie einzeln nacheinander (seriell) subtrahiert.

Dieses Beispiel erweckt den Eindruck, als könne ein Digitalrechner (echt) subtrahieren. Zu den Selbstverständlichkeiten im Befehlssatz eines Mikroprozessors gehört ein Subtraktionsbefehl. In Wirklichkeit können die Rechner aber nicht „echt“ subtrahieren, sie können nur addieren. Den Effekt der Subtraktion erreicht man aber auch durch die Addition des Komplements (comple, lat. auffüllen; gemeint ist das Ergänzen der Stellen bis zu ihrem höchstmöglichen Inhalt).

Beispiel:

$$8-3=5$$

„-3“ ist technisch nicht zu realisieren.

ren. Der Rest, der von der „3“ bis zur nächsten vollen Stelle (also 10) fehlt, ist „7“. Dieser Rest wird addiert. Dabei ergibt sich ein Stellenübertrag, in diesem Fall ein Zehner. Wenn man den fortfallen läßt – z. B. weil für ihn keine Stelle vorhanden ist, kommt das der Subtraktion dieser Stelle gleich, in unserem Beispiel also der Subtraktion von „10“. Man hat also „7“ addiert und „10“ fortfallen lassen (subtrahiert). Der Effekt ist, daß real „3“ wie gewünscht subtrahiert wurden:

$$\begin{array}{r} 8+7-10=5 \\ -3 \end{array}$$

In dem obigen Beispiel wurden aber zwei Rechenschritte zusammengefaßt; bei einer mehrstelligen Zahl wäre es auch sofort aufgefallen: In Wirklichkeit kann man einen Stelleninhalt nicht bis „10“ ergänzen, sondern nur bis „9“. Die Auffüllung einer Stelle bis zu ihrem höchstmöglichen Inhalt ist das „Stellenkomplement“. Dazu muß man noch „1“ addieren, um die nächsthöhere Stelle und die „Nullen“ in allen übrigen Stellen zu erreichen. Die oben addierte „7“ besteht aus dem **Stellenkomplement „6“** (von 3 bis 9 fehlt 6) **plus 1**.

Bei mehrstelligen Zahlen wird zu jeder Stelle das Stellenkomplement gebildet, anschließend wird „1“ addiert:

$$\begin{array}{r} 657-134= \\ \quad 865 \text{ (Stellenkomplement zu 134)} \\ + 1 \\ \hline 866 \end{array}$$

$$657+866 (-1000) = (1)523$$

Bei Dualzahlen ist das Komplement leicht herzustellen, denn man braucht nur die Nullen gegen Einsen oder umgekehrt auszutauschen; „1“ ist der höchste Stelleninhalt; von 1 bis 1 fehlt 0; von 0 bis 1 fehlt 1.

dekadisch	dual		
8	1000	1000	
-3	-0011	+1100	Zweier-
5	0101	+ 1	komplement
		(1)0101	

Die Summe aus dem einfachen Stellenkomplement plus 1 ist das Zweierkomplement. Das Zweierkomplement wird in der Datentechnik allgemein zur Darstellung negativer Zahlen gebraucht (siehe unten „arithmetische Arbeitsweise“).

Wenn man eine größere Zahl von einer kleineren subtrahiert, ist das Ergebnis eine negative Zahl, z. B.

$$3-8=-5$$

Bei der Subtraktion mittels des Zweierkomplements tritt in einem solchen Fall **kein Übertrag** auf, der zu ignorieren wäre, und das Ergebnis der Rechnung erscheint in seinem Zweierkomplement:

dekadisch	dual	
3	0011	
-8	+0111	
-5	+ 1	Zweierkomplement
	111	Überträge
	1011	$\hat{=} -0101$

Wenn man das Ergebnis in seiner „richtigen“ Gestalt haben möchte, ist erst sein Bitmuster umzukehren (zu invertieren), und dann ist „1“ zu addieren. Was hier umständlich erscheinen mag, ist für den Computer kein Problem.

Woran merkt man aber, ob ein Ergebnis positiv oder negativ ist? Bei einem positiven Ergebnis entsteht ein Überlauf, der unterdrückt werden muß, bei einem negativen Ergebnis entsteht dieser Überlauf dagegen nicht. Mit diesem Überlauf kann man ein Flip-Flop setzen oder nicht setzen lassen und je nachdem, ob es gesetzt wurde, die zusätzliche Bildung des Zweier-

komplements vornehmen lassen oder nicht. Der Prozessor kann also selbst über die Art des Ergebnisses entscheiden.

Der „Merker“ für den Vorzeichenwechsel durch Überlauf ist bei unserem Mikroprozessor das OVERFLOW-Bit (Überlaufbit) im Programm-Status-Wort (siehe Seite 114). Vergleichbare Anzeigen (Flag, engl. Flagge) sind in jedem Allzweckmikroprozessor zu finden.

Eine weitere Möglichkeit, negative Zahlen zu erkennen bzw. zu verarbeiten, ergibt sich aus den verschiedenen Betriebsarten eines Mikroprozessors.

Wie bereits oben erwähnt, verfügt der 2650 über acht Datenleitungen, D0 bis D7, welche die Stellen 2^0 bis 2^7 repräsentieren.

In der **logischen Betriebsart** (logical mode, siehe Seite 99) werden alle acht Stellen als positive Dualzahl interpretiert. 1 Byte erfaßt damit den Zahlenbereich von 0 bis 255 (00_{16} - FF_{16}).

In der **arithmetischen Betriebsart** (arithmetic mode, siehe Seite 99) dient das werthöchste Bit D7 als Vorzeichen:

D7=0 bedeutet: Der Inhalt der Bits D0 bis D6 ist eine positive Zahl.

D7=1 bedeutet: Der Inhalt der Bits D0 bis D6 ist eine **negative Zahl in Zweierkomplementdarstellung**.

Für den eigentlichen Betrag einer Zahl bleiben nur noch sieben Stellen, nämlich die Bits D0 bis D6. Es können daher Zahlen von 0 bis +127 ($D7=0$; 00_{16} bis $7F_{16}$) sowie von -1 bis -128 ($D7=1$; $FF_{16} \hat{=} -1$ bis $80_{16} \hat{=} -128$) dargestellt werden.

Beispiel: Das Bitmuster 1011 1100 heißt

in „logischer Arbeitsweise“: +188
in „arithmetischer Arbeitsweise“: -68

Die Multiplikation

Die Multiplikation ist besonders leicht, weil das gesamte „kleine Einmaleins“ aus nur drei Aufgaben besteht:

$$\begin{array}{l} 0 \cdot 0 = 0 \\ 0 \cdot 1 = 0 \\ 1 \cdot 1 = 1 \end{array}$$

Multipliziert wird wie bekannt:

$$\begin{array}{r} 3 \cdot 5 = 15 \quad \text{Multiplikand Multiplikator} \\ \quad \quad \quad 111 \cdot 101 \\ \quad \quad \quad \underline{111} \\ \quad \quad \quad 000 \\ \quad \quad \quad \quad 111 \\ \text{Überträge} \quad \underline{11} \\ \quad \quad \quad \quad \quad 100011 \end{array}$$

Taucht im Multiplikator eine „1“ auf, wird der Multiplikand abgebildet und um die Stelle des Multiplikators nach rechts verschoben addiert.

Taucht im Multiplikator eine „0“ auf, so wird die Zwischensumme zwar um die Stelle der „0“ verschoben, es braucht aber nichts addiert zu werden. Das Multiplizieren ist also ein schrittweises, stellenverschobenes Addieren des Multiplikanden.

Die Division

Dualzahlen werden wie Zehnerzahlen dividiert. Der Einfachheit halber seien die hierbei auftretenden Subtraktionen als „echte“ Subtraktionen notiert. Das ist insofern realistisch, als die Prozessoren einen Subtraktionsbefehl kennen und somit die Bildung des Zweierkomplements nicht sichtbar wird. In Wirklichkeit läuft aber bei jeder Subtraktion die Bildung des Zweierkomplements mit anschließender Addition und Unterdrückung des Übertrags ab:

dekadisch
45:5=9

dual
101101:101=1001
101
01
0
10
0
101
0

Schaltungstechnik

Die Erzeugung der Nullen und Einsen

Die L- bzw. H-Pegel, die den Nullen und Einsen entsprechen, lassen sich sehr leicht mit je einem kleinen Schiebeshalter erzeugen. Die Schaltung nach Bild 6.4a ist der einfachste Weg: Die Datenleitung D wird entweder an 0 V oder an VCC geschaltet. Dadurch ergeben sich auf ihr die L- oder H-Pegel. Die Schalter brauchen nicht entprellt zu werden, weil man ja eine gewisse Zeit braucht, um nach dem Einstellen des Datenworts einen Taster auf der CPU-Platte zu bedienen, und so lange prellen selbst die schlechtesten Schalter nicht.

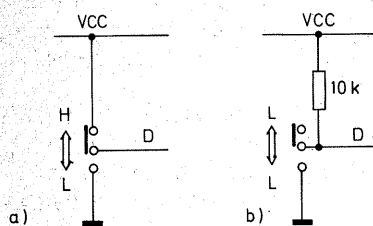


Bild 6.4 Erzeugung von H- und L-Pegeln:
a) durch einen UM-Schalter, b) durch einen Widerstand mit einem EIN/AUS-Schalter

Die Schaltung hat aber einen Nachteil, der unangenehme Folgen nach sich ziehen könnte: Die Datenleitung ist nicht kurzschlußfest. Wenn sie den

jeweiligen Gegenpol der Versorgungsspannung berührt, im Falle L die Leitung VCC, im Falle H die Masse, so gibt es in der Spannungsversorgung des Geräts einen Kurzschluß. Abhilfe schafft die Schaltung nach Bild 6.4b. Der H-Pegel wird über einen Widerstand von 10 kΩ erzeugt. Der Widerstand darf so groß sein, weil in die TTL-Eingänge kein nennenswerter Eingangsstrom fließt (siehe Seite 64). Wenn der Schalter offen ist, liegt die Datenleitung über den Widerstand auf H; beim Schließen des Schalters wird die Datenleitung auf L geschaltet. Über den Widerstand fließt ein Strom von ca. 0,5 mA – der fällt nicht ins Gewicht.

Abschließend sei angemerkt, daß für jede Datenleitung je ein Widerstand und ein Schalter vorhanden sein müssen. Der Schalter braucht kein UM-Schalter zu sein, es genügt ein einfacher Schließer.

Die Anzeige der Nullen und Einsen

Zur Anzeige, ob eine 0 oder eine 1 eingestellt ist, reicht eigentlich schon die Schalterstellung aus. Besser ist jedoch, den Pegel mit einer Leuchtdiode anzuzeigen. Diese LED soll zugleich auch die Signale anzeigen, die der Prozessor abgibt. Die LED-Anzeige ist eine im Vergleich zu den Bildschirmanzeigen sehr einfache Form der Signaldarstellung. Sie zeigt dafür aber das wirkliche Geschehen auf dem Datenbus. Und was eine einfache Anzeige mittels einer LED oder eines Glühlämpchens wirklich leistet, offenbart am besten ein Blick in die Geschichte der Elektrotechnik und Signalübermittlung.

Eine der ersten elektrisch betriebenen Signalübermittlungsanlagen war die des Spaniers *Francisco Salvá*, deren

längste – etwa 50 km lang – Madrid und Aranjuez während der Jahre 1796–1798 verband. Salvá wollte 22 Buchstaben des Alphabets übermitteln. Für jeden der 22 Buchstaben verwendete er eine Doppelleitung. Als Stromquelle diente ihm eine besonders leistungsstarke Elektriziermaschine. Nun brauchte er nur noch 22 Anzeigen. Mangels technischer Mittel bediente er sich einer drastischen Methode: Er gab 22 Männern je eine Doppelleitung in die Hand und teilte jedem von ihnen einen Buchstaben zu. Wer einen Stromschlag erhielt, mußte den ihm zugeordneten Buchstaben ausrufen. Wie hätten sich wohl Salvás Männer über Leuchtdioden gefreut!

Weder die Schalter mit den Widerständen noch die Prozessorausgänge können ausreichend Strom abgeben, um eine LED zum Aufleuchten zu bringen. Daher wird an jede Datenleitung eine Verstärkerstufe in Form eines Inverters angeschlossen (Bild 6.5). Ein (Darlington-)Transistor würde die Aufgabe ebensogut erfüllen. Die Schaltung wäre aber wegen der großen Anzahl der Anzeigen etwas umständlich aufzubauen.

Es mag überraschend wirken, daß zur Anzeige eines H-Pegels ausgerechnet

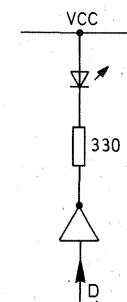


Bild 6.5 Anzeigeverstärker mit einem Inverter

ein Inverter benutzt wird. Der Inverter arbeitet im Prinzip wie ein Transistor in Emitterschaltung, also wie ein Schalter, der durch H-Signal eingeschaltet wird. Wenn sein Eingang L ist, ist sein Ausgang H. Zwischen den Anschlüssen der LED steht keine nennenswerte Spannung – die LED leuchtet nicht und zeigt damit L ($\cong 0$) an. Ein H-Pegel am Eingang bewirkt, daß dessen Ausgang auf L geschaltet wird. Zwischen den Anschlüssen der LED einschließlich ihres Vorwiderstandes steht nun die Differenz zwischen VCC und (nahe) 0 V; die LED leuchtet und zeigt H ($\cong 1$) an.

Als Anzeigeverstärker sind Inverter mit Open-Collector-Ausgang (Bild 2.29b) besser geeignet als die mit dem normalen TTL-Gegentaktausgang. Daher wurde der Typ 74LS05 (Anschlußbelegung, Bild 2.9) gewählt. Wenn Sie aber andere Typen mit gleicher Anschlußbelegung in der Bastelkiste haben (7406, 7407, 7416, 7417 einschließlich der LS- oder S-Typen), können Sie auch diese verwenden.

Nur IC1 (Bild 6.7) muß ein Open-Collector-IC sein, weil der Inverter N1 (an der Leitung WRITE/READ) mit dem Inverter N7 der CPU-Platte (Bild 7.12, ebenfalls an der Leitung WRITE/READ) ein Wired-NOR bildet (Bild 2.30).

Der LED-Vorwiderstand ist mit 330 Ω recht hoch gewählt. Die LEDs brauchen ja nicht sehr hell zu leuchten. Es genügt, wenn ihr Schaltzustand gut sichtbar wird. So können Sie viel Strom sparen, und Sie belasten Ihr Netzteil nicht so stark. Der LED-Strom beträgt etwa 10 mA statt ca. 18 mA beim üblichen Vorwiderstand von 180 Ω. Bei der großen Anzahl der LEDs macht sich dieses schon bemerkbar.

Wenn Ihre Bastelkiste es hergibt, lei-

sten Sie sich superhelle LEDs. Sie strahlen bei einem Strom von 20 mA mit mehr als 70 mcd gegenüber etwa 4 bis 6 mcd normaler LEDs (z. B. die Typen CQX 24, V 310 P, HLMP 3750). Dann können Sie den Vorwiderstand auf 1 k Ω erhöhen und benötigen nur ca. 3,2 mA pro LED.

Wenn die LEDs einen von einem Schalter bestimmten Zustand anzeigen sollen, kommt es nur auf die Schalterstellungen an. Während dieser Zeit müssen aber die Ausgänge der anderen Bausteine (Prozessor, Speicher) im hochohmigen Zustand sein (siehe Seite 63).

Soll die LED dagegen einen Pegel anzeigen, der vom Prozessor kommt, muß der zugehörige Schalter **offen** sein, weil er sonst einen H-Pegel des Prozessors kurzschließt. Der Prozessor übersteht das schon eine Weile, denn seine Ausgänge sind in gewissem Maße kurzschlußfest, nur Daten können Sie dann nicht lesen. Zum Lesen müssen Sie daher alle Schalter auf „1“ schieben. Das ist freilich etwas mühsam, doch wenn man alle Datenleitungen über Dioden entkoppelt, kann man diesen Effekt mit einem einzigen Schalter, dem Datenschalter Da in Bild 6.7, erreichen.

Wenn **Da** in Stellung **WRITE** steht, ist die Sammelleitung der Dioden auf L geschaltet, daher können die Datenleitungen über ihre Schalter auf L oder H gesetzt werden. Diese Schalterstellung dient dazu, Daten auf den Datenbus zu geben und in den Prozessor oder Speicher einzuschreiben, daher die Bezeichnung **WRITE** (to write, engl. schreiben). Falls in dieser Schalterstellung Daten vom Prozessor kommen, setzt sich der jeweilige L-Pegel durch: Kommt vom Prozessor ein H und trifft auf einen geschlossenen Schalter (\cong L), wird das

H von ihm kurzgeschlossen. Kommt vom Prozessor ein L und trifft auf einen geöffneten Schalter (also H über 10 k Ω und den Invertereingang), zieht der Prozessor dieses hochohmige H auf L.

Der Datenschalter Da erzeugt aber auch noch ein Steuersignal auf der Leitung **WRITE/READ**: Der Eingang des Inverters N1 wird auf H gesetzt, denn ein offener TTL-Eingang verhält sich ja so, als hätte er H, so daß die Leitung **WRITE/READ** auf L gezogen wird. Dieses Signal wird später gebraucht, um den Speicher zur Aufnahme von Daten („einschreiben“) zu öffnen.

Wenn **Da** in Stellung **DAFLOT** (data float, engl. die Daten schweben) steht, sind alle Invertereingänge gewissermaßen von den Schaltern abgetrennt, denn die Schalter können sie jetzt ja nicht mehr auf L schalten. Die Leitungen werden daher nur noch von

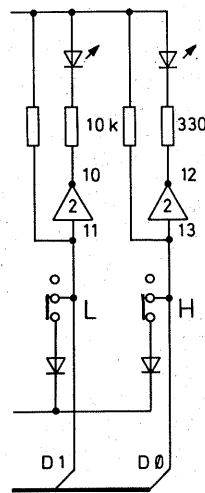


Bild 6.6 Entkopplung der zusammengefaßten Datenleitungen mit Dioden

den Datenleitungen des Prozessors oder des Speichers beeinflusst. Die Dioden an den Datenleitungen sind zur Entkopplung unentbehrlich. Bild 6.6 zeigt die zwei Datenleitungen D0 und D1. Wenn auf D0 ein H steht (also eine positive Spannung gegen Masse), so ist dafür die Diode von D1 in Sperrichtung gepolt; somit kann das H von D0 nicht an D1 gelangen. Umgekehrt wäre es natürlich genauso: Ein H von D1 könnte nicht auf D0 gelangen, weil dann die Diode von D0 in Sperrichtung gepolt dazwischen liegt.

Das L-Signal, das auf der Datenleitung D1 liegt, kann diese Leitung nicht „verlassen“, weil die Diode der Datenleitung D1 dafür in Sperrichtung gepolt ist.

Was hier an zwei Datenleitungen dargestellt ist, gilt natürlich für sämtliche Datenleitungen.

Zugleich wird in der Schalterstellung **DAFLOT** der Eingang des Inverters N1 auf L geschaltet. Dadurch erhält die Leitung **WRITE/READ** ein H-Signal, das später gebraucht wird, um den Speicher zum **Lesen** seiner Daten (to read, engl. lesen) zu öffnen.

Die Behandlung der Leitung **WRITE/READ** zeigt etwas für die Digitaltechnik Typisches: **Damit ein Signal aktiv wird, muß es nicht unbedingt H sein.** Ob es mit H oder L aktiv ist, hängt vom jeweiligen Baustein ab. Um anzugeben, mit welchem Pegel eine Funktion aktiviert wird, gibt man ihr ein Vorzeichen mit: Der Negationsstrich über der Benennung eines Ein- oder Ausgangs (**WRITE**) gibt an, daß er mit L aktiviert wird. Ist nur der Name angegeben (z. B. **READ**, ohne Negationszeichen), bedeutet dies, daß ein H-Pegel das aktive oder aktivierende Signal ist.

Bild 6.7 zeigt den vollständigen

Stromlaufplan. Die Baugruppe enthält neben den Datenanzeigen auch noch LEDs für die wichtigsten Steuersignale: **WRITE**, **ADMEM** (Adressen **Memory**, engl. Speicher) und **ADPER** (Adressen der Peripherie). Die Negationsstriche geben an, daß die Signale mit L aktiv sind:

Ist die LED **WRITE** dunkel (L), können Daten eingegeben (geschrieben) werden, leuchtet die LED **WRITE** (H), so können Daten nur gelesen werden.

Leuchtet die LED **ADMEM** (H), so sind die Adressen des Speichers gesperrt, d. h. der Speicher ist elektronisch abgetrennt. Ist die LED **ADMEM** dagegen dunkel (L), so ist der Speicher adressiert und steht für Les- oder Schreiboperationen offen.

Analog verhält es sich mit der LED **ADPER**: Leuchtet sie (H), so sind alle Peripherien abgetrennt, ist sie dunkel (L), so steht die adressierte Peripherieeinrichtung zur Aufnahme oder Abgabe von Daten offen.

Abschließend noch ein Wort zu den beiden Kondensatoren an der Plusleitung. Der Elko muß nicht unbedingt eine Kapazität von genau 220 μ F haben. Er soll nur den (induktiven) Widerstand der Leitungen zur Spannungsversorgung kompensieren. Er darf wesentlich geringer sein (bis minimal 47 μ F) oder auch größer. Sie dürfen sich in diesem Fall nach Ihren Vorräten richten. Der Kondensator 0,1 μ F sollte unbedingt ein keramischer Scheibenkondensator sein, denn diese Bauform hat einen sehr geringen induktiven Widerstand und ist daher geeignet, HF-Reste, die durch den Impulsbetrieb auf der Plusleitung entstehen, kurzzuschließen.

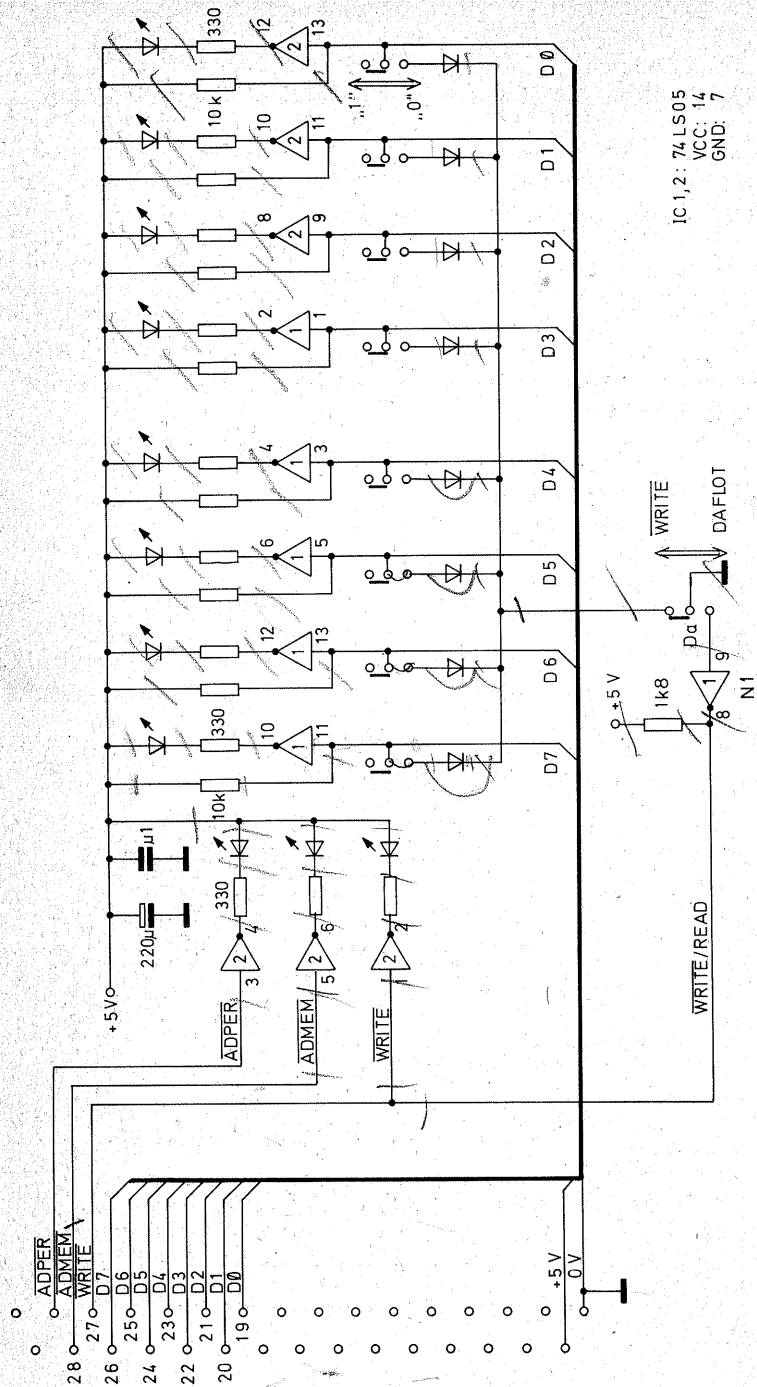


Bild 6.7 Stromlaufplan der Dateneingabe und -anzeige

Hinweise zum Aufbau

Bild 6.9 zeigt die Kupferseite der Platine, Bild 6.10 die Bestückung der Oberseite und Bild 6.11 die zusätzlichen Drahtbrücken auf der Kupferseite.

Sie beginnen die Bestückung am besten mit den kleinen Bauelementen, den Dioden und den Widerständen.

Die Montage der Schiebeschalter ist die einzige etwas schwierigere Arbeit. Stecken Sie zuerst 0,5 bis 0,6 mm starke Drähte durch die Lötösen der Schalter, und löten Sie sie zwischen den Lötösen an (Bild 6.12). Dann biegen Sie sie um und schneiden sie stufenweise ab (Bild 6.13). So lassen sich die Drähte leichter in die Platine einfädeln.

Löten Sie zuerst nur je einen der mittleren Anschlußdrähte an die Kupferbahn an; dann können Sie die Schalter ausrichten und die restlichen Drähte anlöten. Achten Sie darauf,

- | | |
|----|---|
| 1 | Leiterplatte |
| 1 | 31polige Stiftleiste Baureihe GdsW |
| 2 | Lötangel RTM 1,3 mit Steckhülse |
| 2 | Fassung DIL 14 |
| 9 | kleiner Schiebeschalter |
| 8 | Silizium-Allzweckdiode, z.B. 1N4148 |
| 2 | 74LS05 |
| 11 | LED |
| 11 | Widerstand 330 Ω, 0,125 W |
| 1 | Widerstand 1,8 kΩ, 0,125 W |
| 8 | Widerstand 10 kΩ, 0,125 W |
| 1 | Elektrolytkondensator 47 bis 470 µF/16V |
| 1 | keramischer Scheibenkondensator 0,1 µF |

Bild 6.8 Stückliste für die Dateneingabe und -anzeige

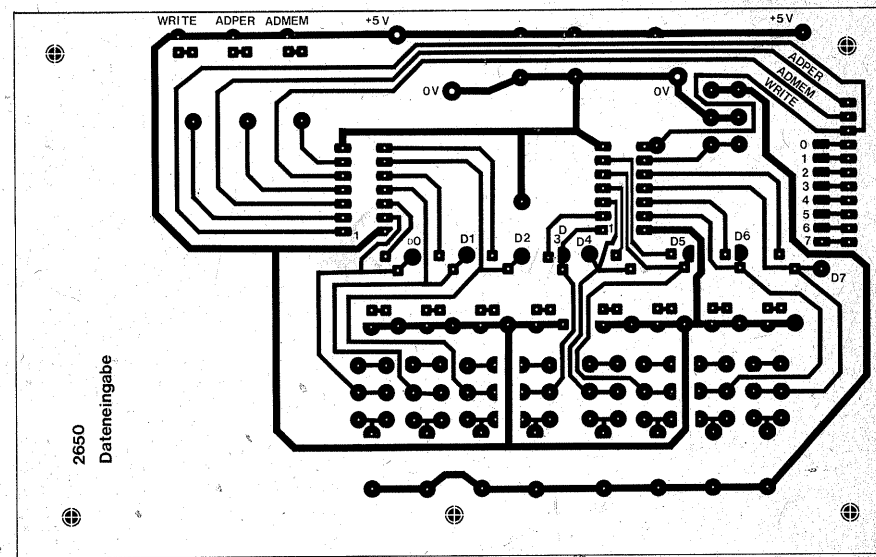


Bild 6.9 Leiterplatte für die Dateneingabe und -anzeige

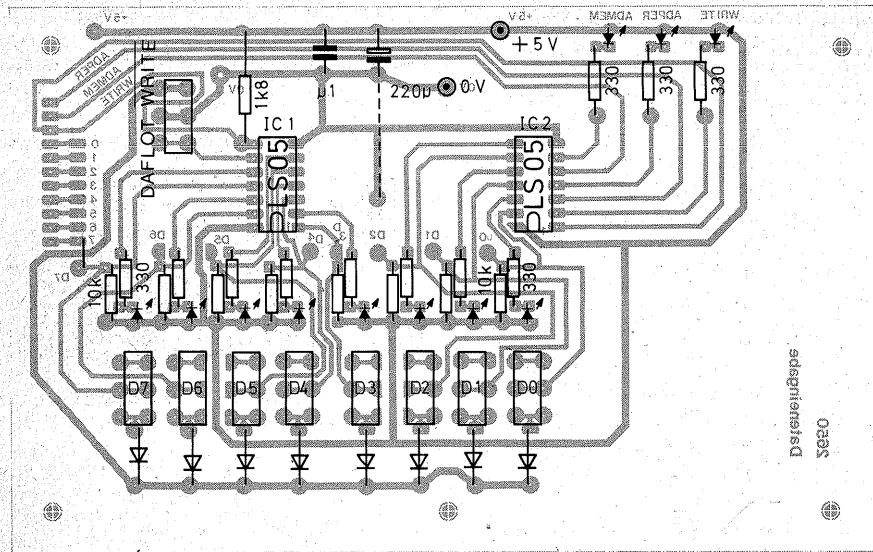


Bild 6.10 Bestückungsplan für die Dateneingabe und -anzeige

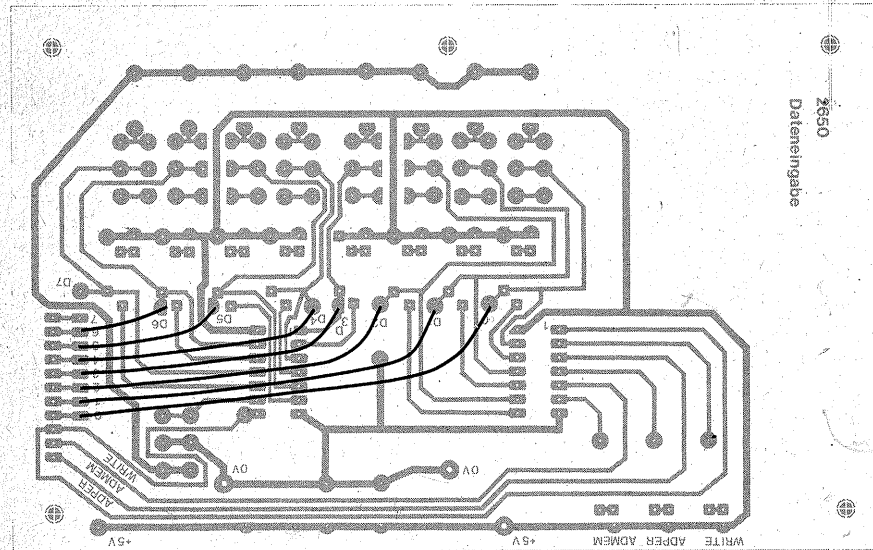


Bild 6.11 Die Drahtbrücken auf der Kupferseite der Dateneingabe und -anzeige

daß die Schalter fest auf der Platine aufsitzen (Bild 1.1).

Das Einsetzen der LEDs: Die Anschlußdrähte der LEDs verdicken sich etwa 4,5 mm vor dem Gehäuse so stark, daß sie nicht mehr durch die 1-mm-Bohrungen passen. Setzen Sie die LEDs fest auf ihre verdickten Anschlußenden, dann sind sie alle gleich hoch. Achten Sie beim Einsetzen auf die richtige Polung. Die abgeflachte Seite bzw. der kurze Draht ist die Kathode. Löten Sie die LEDs erst am Anodendraht fest, richten Sie dann die Reihe der LEDs aus und löten danach die Kathoden an.

Prüfen Sie, ob die LEDs den Einbau überstanden haben.

Schließen Sie eine Flachbatterie mit dem Pluspol an den Stift +5V an, und tippen Sie mit dem Minuspol (Prüfkabel und Stecknadel) die den LEDs abgewandten Enden der 330-Ω-Widerstände an. Dann muß jeweils die betreffende LED aufleuchten. Leuchtet sie nicht auf, so gibt es verschiedene Fehlermöglichkeiten:

- Die LED ist falsch gepolt;
- kalte Lötstelle(n) an der LED oder ihrem Vorwiderstand;
- Widerstände 330 Ω/10 kΩ vertauscht;
- die LED hat die Lötwärme nicht überstanden, was aber sehr selten vorkommt.

Statt die Enden der Widerstände anzutippen, können Sie auch nach dem Einlöten der IC-Fassungen diesen Test an den Anschlüssen für die Inverterausgänge (2, 4, 6, 8, 10, 12) durchführen.

Bestücken Sie die Platine erst weiter, wenn dieser Test positiv verlaufen ist. Vorher hat es keinen Sinn, weiterzuarbeiten.

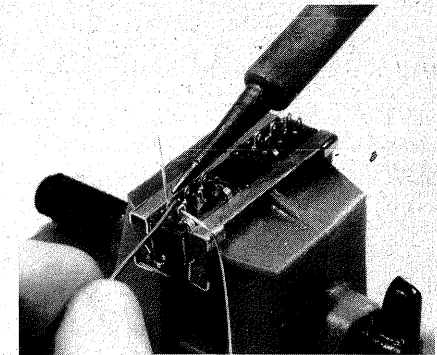


Bild 6.12 Vorbereiten der Schiebeshalter zur Montage auf der Leiterplatte

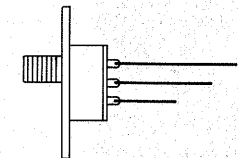


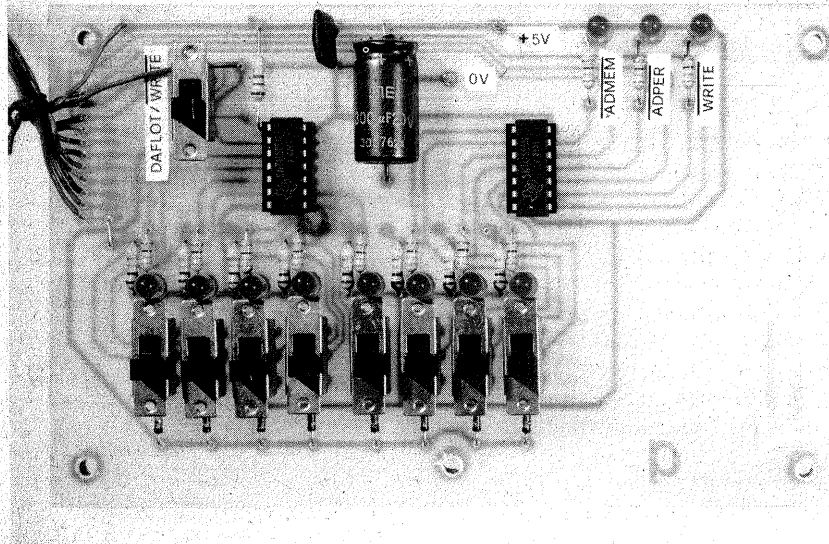
Bild 6.13 Stufenweise gekürzte Anschlußdrähte der Schalter

Die Folge der übrigen Bestückung ist gleichgültig. Die Drähte auf der Kupferseite (Bild 6.11) löten Sie zuletzt an. Dabei richten Sie sich nach der Beschriftung: 0 an D0, 1 an D1 usw. Schließen Sie nun die 13 ca. 30 cm langen Anschlußleitungen für die Stiftleiste an, und notieren Sie sich die Farbmarkierungen.

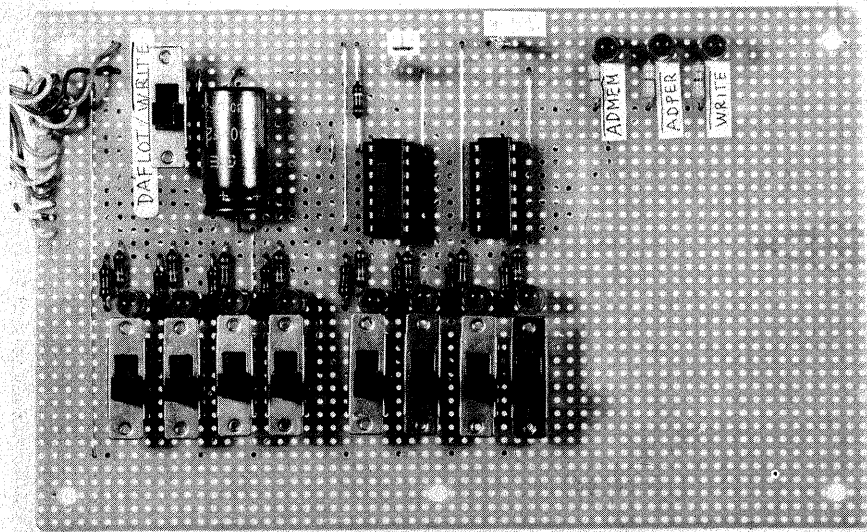
Bohren Sie am linken Rand der Platine oben ein Loch mit etwa 2,5 mm Ø und binden die Leitungen dort fest. Setzen Sie die Stiftleiste in Platz 2 der Busplatte (Bild 5.5) und löten die Leitungen gemäß Kapitel 5.5 an.

Hinweise für den Aufbau einer Experimentierplatte

1. Bild 6.15 zeigt die Anordnung der Bauelemente auf einer Experimentierplatte. Sie unterscheidet sich nicht



Oben:
Bild 6.14 Die fertige Dateneingabe und
-anzeige



Unten:
Bild 6.15 Aufbau der Dateneingabe und
-anzeige auf einer Experimentierplatte. Der
Großteil der Verdrahtung befindet sich auf der
Unterseite

wesentlich von der Anordnung auf der Platine (Bild 6.14).

2. Setzen Sie zuerst als Orientierungspunkte die Schalter und die IC-Fassungen ein.

3. Verlegen Sie die Drahtbrücken für die Spannungsversorgung der ICs und der LEDs auf der Isolierseite. Schließen Sie eine Batterie an die Stifte +5V und 0V an, und messen Sie, ob Spannung an den IC-Stiften 7 bzw. 14 sowie an der Sammelleitung für die LEDs und die 10-k Ω -Widerstände richtig ankommt. Löten Sie die LEDs und die Widerstände ein, und lassen Sie den oben angegebenen Zwischentest unter keinen Umständen aus.

4. Die acht Schaltdioden löten Sie am besten unterhalb der Experimentierplatte auf die Leiterbahnen. Als Sammelschiene für die Kathoden dient die sechste Leiterbahn vom vorderen Rand aus gezählt.

5. Beim Verdrahten der Daten- und Anzeigeleitungen löten Sie zuerst eine „Doppelleitung“ (Bilder 1.26 und 1.27) an den Invertereingang (z. B. Anschluß 11 von IC1 für D7), führen das eine Ende an den 10-k Ω -Widerstand, das andere an den Mittelkontakt des Schalters und löten es dort wieder als Doppelleitung an. Das freie Ende führen Sie an den linken Rand der Experimentierplatte, wo Sie den Kabelbaum anlöten. Achten Sie darauf, daß in einer solchen Verbindungsfolge die Farbcodierung der Litze gleich bleibt. Nehmen Sie für jede Datenleitungsfolge eine andere Farbcodierung. So bleibt das Drahtgewirr überschaubar.

Prüfung des Dateneingabe- und -anzeigebausteins

1. Stellen Sie den Datenschalter Da auf DAFLOT. Alle LEDs müssen leuchten.

2. Stellen Sie den Datenschalter auf WRITE. Die LED WRITE muß verlöschen. Die LED ADMEM und ADPER leuchten auch in dieser Schalterstellung. Die Daten-LEDs leuchten oder sind dunkel, je nach der Stellung ihrer zugehörigen Schalter.

3. Stellen Sie alle Eingabeschalter auf „0“ (zum Platinenrand schieben). Alle LEDs müssen erlöschen. Prüfen Sie nun der Reihe nach, ob sie sich einzeln ein- oder ausschalten lassen, indem Sie den zugehörigen Eingabeschalter von „0“ auf „1“ und wieder auf „0“ stellen. **Fehlermöglichkeiten:** Wenn eine LED bei Schalterstellung „1“ nicht aufleuchtet, kann eine Unterbrechung im Leitungszug vorliegen. Aber auch ein Kurzschluß mit einer benachbarten Datenleitung verhindert das Aufleuchten der LED, weil Sie alle anderen Eingabeschalter ja auf „0“ stehen haben.

Leuchten bei Betätigung eines Eingabeschalters zwei LEDs auf, so liegt mit Sicherheit ein Kurzschluß zwischen den betroffenen Datenleitungszügen vor, der sich meist leicht finden läßt, weil sich die Fehlersuche auf wenige Punkte beschränkt.

4. Wenn alle Daten-LEDs richtig reagieren, messen Sie auf der Busplatte nach, ob alle Eingabesignale an den richtigen Buchsen ankommen. Schließen Sie das Minuskabel Ihres Multimeters an den Stift 0V an, tasten mit dem Pluskabel, u. U. mit Hilfe einer

Nadel, die Buchse 19 einer beliebigen Federleiste ab und schalten den Eingabeschalter D0 von „0“ auf „1“ und wieder auf „0“. Ihr Vielfachinstrument muß Ihnen in der Reihenfolge die Spannungssprünge L (ca. 0,6 V, wegen der Schaltdioden) – H – L anzeigen.

Achten Sie besonders darauf, daß die **Zuordnung der Datenleitungen stimmt**. Bei Schülerarbeiten sind mehrfach Vertauschungen der Reihenfolge (D7 an der Stelle von D0, D6 an der Stelle von D1 usw.) vorgekommen. Mit so vertauschten Datenleitungen kann sich der Prozessor nur „unlogisch“ verhalten.

5. Prüfen Sie die Leitungen WRITE, ADMEM und ADPER. Schalten Sie den Datenschalter Da wieder auf DAFLOT. Verbinden Sie Buchse 27 einer Federleiste mit 0 V, muß die LED WRITE erlöschen. Wenn Sie dasselbe mit Buchse 28 tun, muß die LED ADMEM erlöschen und bei Buchse 29 die LED ADPER:

Übung von Eingaben

Sie sollten jetzt schon mit diesem Baustein spielen und Eingaben üben. Einerseits gewöhnen Sie sich an die Handhabung, andererseits können Sie die Dual- bzw. Hexadezimalzahlen schon einüben (Tabelle 6.1). Sie werden sie dringend brauchen.

Geben Sie z. B. 04_{16} ein (0000 0100). Das ist der erste Ladebefehl, mit dem Sie später die CPU-Platte ausprobieren. Er bewirkt, daß der Prozessor das nächste Byte, das auf dem Datenbus erscheint, in seinen internen Speicher aufnimmt.

$F0_{16}$ (1111 0000) ist ein Schreibbefehl. Er veranlaßt den Prozessor, den In-

halt seines internen Speichers wieder auf den Datenbus zu geben. Auch diesen Befehl werden Sie sofort benötigen.

Stellen Sie sich beliebige Bitmuster ein und versuchen Sie, diese hexadezimal oder dekadisch zu lesen, z. B.:

$10001\ 1011_2 \cong 9B_{16} \cong 155_{10}$ usw.

Wie glücklich Sie mit Ihrem Computer werden, hängt nicht zuletzt davon ab, wie vertraut Sie mit ihm sind. Außerdem werden Sie schon nach kurzer Übung merken, daß die Beherrschung der Dual- und Hexadezimalzahlen keine besonders große Leistung ist. An den nächsten Baustein, die CPU-Platte, sollten Sie aber erst herangehen, wenn Ihnen der Datenbaustein keine unlösbaren Rätsel mehr aufgeben kann, wenn also die Sprache des Computers für Sie keine Fremdsprache mehr ist.

Kapitel 7

Die CPU-Platte (Baugruppe 4)

Das wichtigste Bauelement auf der CPU, die **central processing unit** (to process, engl. bearbeiten; unit, engl. Einheit) ist der Mikroprozessor. „Mikro“ ist er nur in seinen Abmessungen und im Stromverbrauch, jedenfalls im Vergleich zu seinen Vorgängern. Mit seinen rund 7000 Transistoren, den 576 Bits für Festwertspeicher, 250 Bits für Register und ca. 900 Gatterfunktionen ist er alles andere als „mikro“. Hineinschauen, um nachzusehen, wie alles funktioniert, nützt wohl nicht viel, auch wenn einen der Blick durch das Mikroskop das Stauen lehrt (Bild 7.1).

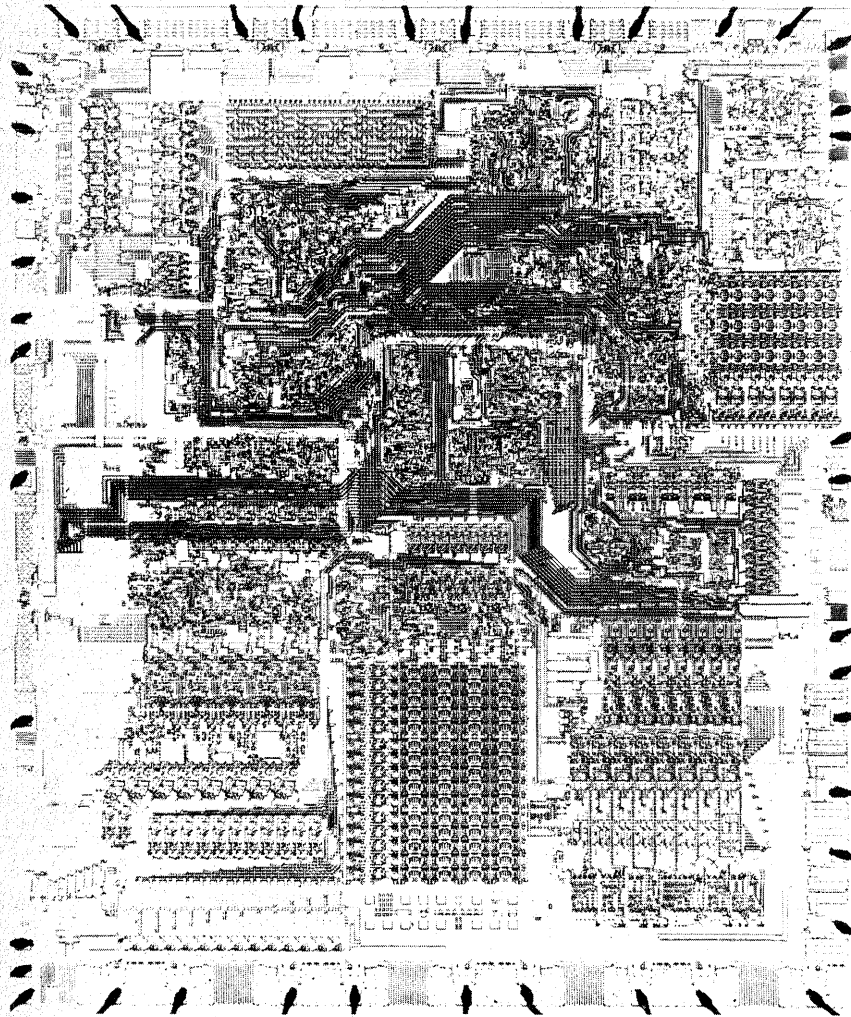
Um mit dem Mikroprozessor sinnvoll umzugehen, brauchen Sie nicht über jeden Transistor oder jedes Gatter Bescheid zu wissen. Es gibt aber einige Strukturelemente, die Sie kennen müssen, weil diese in Ihrer späteren Programmierarbeit eine wichtige Rolle spielen. Die folgenden Angaben sind dem VALVO-Handbuch zum Mikroprozessor 2650 A entnommen.

Die Struktur des 2650 A

Bild 7.2 zeigt Ihnen die Grobstruktur des Mikroprozessors 2650 A. Die Blockschaltung reduziert die Übersicht auf das Wichtigste und auch für andere Prozessoren Typische. Die Pfeile an den Leitungen geben an, ob es sich bei ihnen um Ein- oder Ausgänge des Mikroprozessors handelt.

Das Rechenwerk (ALU)

Den Kern des Mikroprozessors bildet die **ALU** (arithmetic logical unit), das eigentliche Rechenwerk. Diese Einheit führt alle datenbezogenen Befehle durch. Es sind Lade- und Speicherbefehle, die arithmetischen Operationen der Addition und der Subtraktion, die logischen Verknüpfungen UND, ODER, EXOR, Vergleichsoperationen, Rotationsbefehle (Verschiebungen eines Datenworts nach rechts oder links), Inkrementierungen (Addition + 1; incrementum, lat. Zuwachs) und Dekrementierungen (Verringerung um 1; decrementum, lat. Abnahme) eines Datenwortes. Durch die ALU werden auch das Carry-Bit (carry, engl. Übertrag), das Overflow-Bit (overflow, engl. Überlauf; siehe



Seite 116, Subtraktion), das Interdigit-Carry-Bit (Zwischenübertrag zwischen dem unteren und dem oberen Nibble eines Datenworts) und die beiden Condition-Code-Bits (Bedingungsmarken) im Programm-Status-Wort (PSW, siehe Seite 114) berücksichtigt bzw. gesetzt.

Bild 7.1 Mikroskopaufnahme des Mikroprozessors 2650. Der Kristall mißt nur etwa $4,5 \times 4 \text{ mm}^2$ (nach VALVO-Unterlagen)

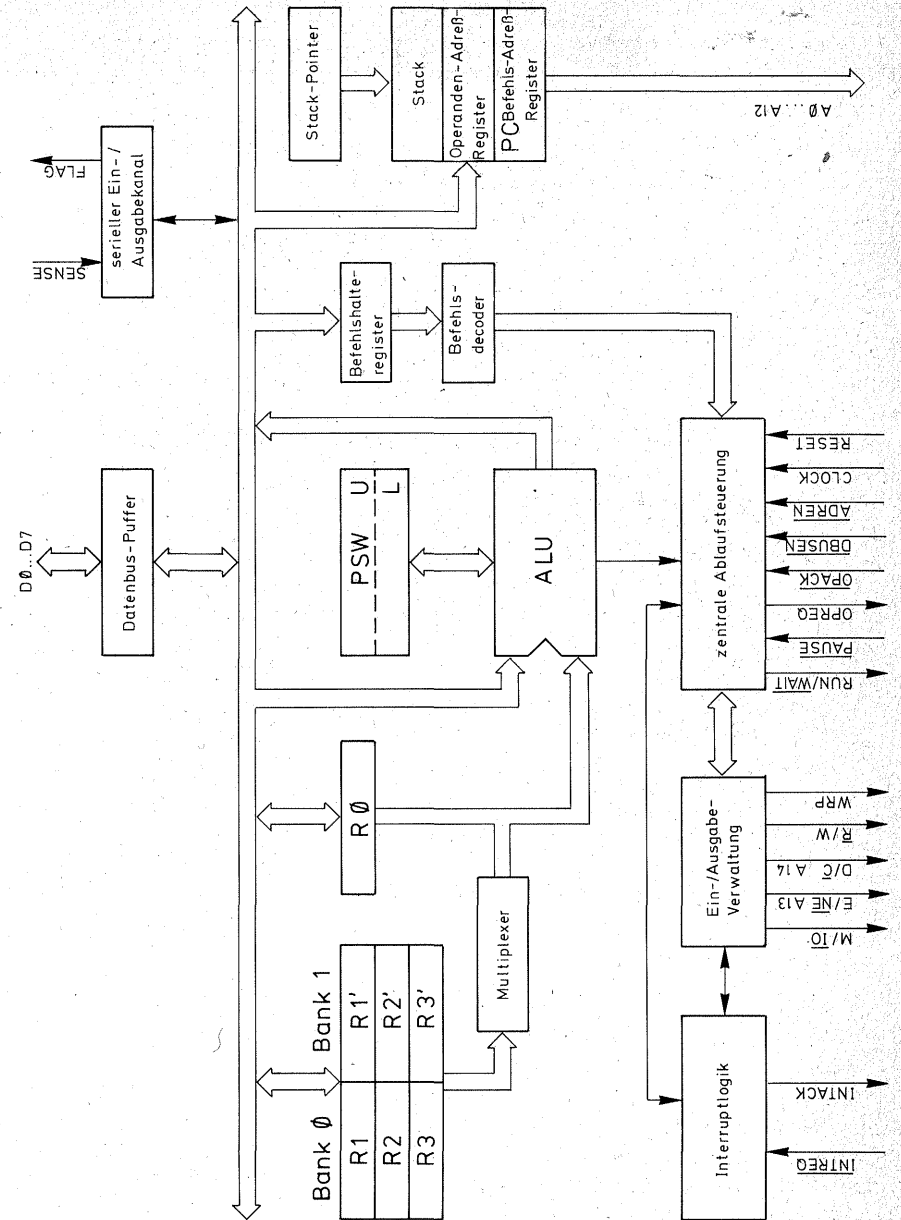


Bild 7.2 Die Grobstruktur des Mikroprozessors 2650

Das Programm-Status-Wort (PSW)

Das Programm-Status-Wort ist ein spezielles 16-Bit-Register im Mikroprozessor, das wesentlich zur Flexibilität des 2650 beiträgt. Es wird in ein oberes (Program Status Upper, PSU) und ein unteres (Program Status Lower, PSL) 8-Bit-Register unterteilt. Die Bits des Programm-Status-Worts können geprüft, geladen, gespeichert, gesetzt oder gelöscht werden. Nur das SENSE-Bit kann nicht intern gesetzt oder gelöscht werden, weil der SENSE-Eingang direkt mit dem Anschluß 1 des Mikroprozessors verbunden und nur von außen beeinflussbar ist.

Das PSW hält den Zustand (lat. status) nach bestimmten Operationen fest und definiert damit die Ausgangssituation für die nächste Operation. Daher kann es die Art, wie ein Befehl ausgeführt werden soll, be-

stimmen. Da der Programmierer aber auch Einfluß auf das PSW nehmen kann, ist er ebenso imstande, die Art zu beeinflussen, wie ein Befehl ausgeführt wird, ob z. B. ein Übertrag berücksichtigt, ein Programmsprung ausgeführt werden soll oder nicht.

Die folgende Aufzählung der Bits wird Sie am Anfang vielleicht verwirren, weil Ihnen deren Wirkung erst bei Ihren Programmierübungen sichtbar werden kann. Sie brauchen sie daher nicht sofort auswendig zu lernen. Für den Anfang mag es genügen, daß Sie wissen, wo Sie die Bedeutung der Bits im PSW finden, Sie also wissen, daß sich das PSW an der Befehlsausführung beteiligt und daß sich bei Überraschungen das Nachsehen im PSW lohnt.

Sie können die Bits des PSW mit den Vorzeichen vergleichen, die man vor die Noten eines Musikstücks setzt, um anzudeuten, ob ein Ton erhöht

oder erniedrigt werden soll. Eine Melodie kann auch überraschende Wendungen nehmen, wenn man beim Singen oder Spielen die Vorzeichen nicht berücksichtigt.

Die einzelnen Bits (Bild 7.3) haben folgende Bedeutung:

Im PSU:

Bit 7: Das SENSE-Bit (S) stellt einen 1-Bit-Eingang in den Mikroprozessor dar. Es gibt den logischen Zustand des SENSE-Eingangs wieder. Das SENSE-Bit kann durch das Programm abgefragt werden. Wenn man das z. B. achtmal nacheinander macht und die so in den Prozessor gelangten Daten immer einen Schritt weiterrückt, hat man ein ganzes Datenwort eingelesen. Nach diesem Prinzip können z. B. Daten, die auf einer Tonbandkassette gespeichert sind, in den Prozessor gelangen.

Bit 6: Das FLAG-Bit (F) ist durch den 1-Bit-Ausgang eines Auffang-Flip-Flops gegeben, welches durch das Programm gesetzt oder gelöscht werden kann. Auf diese Weise kann man Daten nacheinander (seriell) Bit für Bit aus dem Prozessor herausgeben, z. B. auf eine Tonbandkassette.

Bit 5: Das INTERRUPT-INHIBIT-Bit (II) wird bei der Annahme eines Interrupts (lat. Unterbrechung; gemeint ist der Sprung in ein Unterprogramm) automatisch auf „1“ gesetzt. Ein spezieller Rücksprungbefehl löscht dieses Bit wieder. Ist das Interrupt-Inhibit-Bit gesetzt, wird die Annahme von weiteren Interrupts verhindert (to inhibit, engl. verhindern). Durch bestimmte Programm-Status-Wort-Befehle kann dieses Bit zusätzlich manipuliert werden.

Bit 4 und 3: Nicht benutzt.

Bit 2, 1, 0: Der STACK POINTER (SP) „zeigt“ auf jene Adresse des

Speicherplatzes im Stack (siehe Seite 119), die die zuletzt gespeicherte Rücksprungadresse enthält. Der Stackpointer ist als 3-Bit-Dualzähler (Zähler von $2^0 - 2^2 \cong 0 - 7_{10}$) organisiert. Er wird bei Unterprogramm-sprüngen automatisch inkrementiert und bei den Rücksprüngen automatisch dekrementiert (siehe Seite 119).

Im PSL:

Bit 7, 6: Der CONDITION-CODE (CC) besteht aus 2 Bits, die immer dann durch den Prozessor verändert werden, wenn eines der sieben Arbeitsregister (siehe Seite 117) durch einen Befehl geladen oder verändert wird. Ebenso zeigt der Condition-Code auch das Ergebnis eines jeden Vergleichsbefehls (compare) an. Die folgende Tabelle zeigt das Setzen des Condition-Codes, nachdem die Daten in einem der sieben Arbeitsregister verändert wurden. Die Daten in den Registern werden als 8-Bit-Zweierkomplementzahl interpretiert.

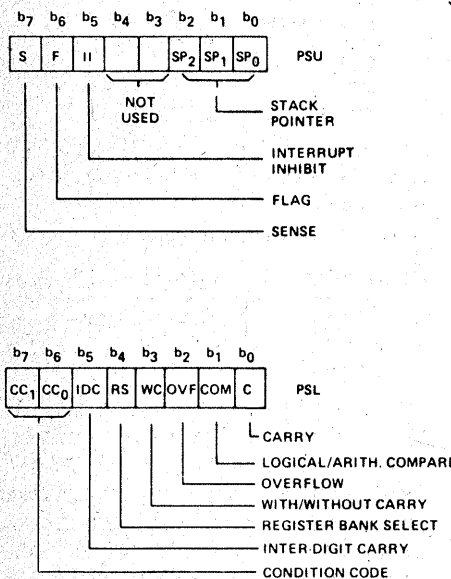
Registerinhalt	CC 1	CC 0
positiv	0	1
null	0	0
negativ	1	0

Die folgende Tabelle zeigt das Setzen des Condition-Codes bei Vergleichsoperationen:

Vergleich der Inhalte von

Register/Speicher	R0/RX	CC 1	CC 0
RX > Speicher	R0 > RX	0	1
RX = Speicher	R0 = RX	0	0
RX < Speicher	R0 < RX	1	0

Ist das Bit 1 COM = 1 (logical mode), so werden die Daten als positive 8-Bit-Zahl interpretiert. Ist COM = 0 (arithmetic mode), so werden die Daten als 8-Bit-Zweierkomplementzahl interpretiert, wobei das Bit D7 als Vorzeichen gilt (siehe Seite 98).



PROGRAM STATUS WORD (PSW)

Bild 7.3 Die Belegung des Programm-Status-Worts (nach VALVO-Unterlagen)

Der Inhalt der Condition-Code-Bits bildet, bei Übereinstimmung mit bestimmten Bits (v-Bits) im Befehlscode, bei Sprungbefehlen die Basis für bedingte Sprünge.

Normalerweise wird der Condition-Code niemals auf $11_2 \triangleq 3_{16}$ gesetzt. Durch die Befehle LPSL oder PPSL kann man ihn jedoch vorübergehend auf 11_2 setzen.

Bit 5: Dieses Bit enthält das INTERDIGIT-CARRY (IDC). Bei der BCD-Arithmetik ist es manchmal notwendig, einen Übertrag von Bit 3 auf Bit 4 zu bekommen. BCD heißt binär codiertes Dezimalsystem. Zur Darstellung einer dekadischen Stelle werden 4 Bits = 1 Nibble benötigt. In einem Byte können also zwei dekadische Stellen (Digit) untergebracht werden, zwischen denen ein Übertrag, eben das Interdigit-Carry, vorkommen kann. Es kann sich sowohl bei der Addition wie bei Subtraktion ergeben, aber auch aufgrund der Rotierbefehle RRR und RRL.

Bit 4: Das REGISTER-BANK-SELECT-Bit (RS) gibt an, welche der beiden Registerbänke verwendet wird (siehe Seite 117).

RS=0
bedeutet Bank 0 mit den Registern R 0, R 1, R 2 und R 3;

RS=1
bedeutet Bank 1 mit den Registern R 0, R 1', R 2' und R 3'.

Das Register R 0 wird also unabhängig von RS benutzt.

Bit 3: Das WITH/WITHOUT CARRY-Bit (WC; with, engl. mit; without, engl. ohne) steuert die Ausführung von Additions-, Subtraktions- und Rotierbefehlen. Das Rotieren kommt einer Multiplikation oder Division gleich. Wird eine Zahl z.B. um eine Stelle nach links verschoben, so ist das praktisch eine Multiplikation mit

2. Die Verschiebung einer Zahl um eine Stelle nach rechts ist eine Division durch 2.

Setzt man WC=0, so wird bei der Durchführung eines Additions-, Subtraktions- oder Rotierbefehls ein vorhandenes Carry bzw. Interdigit-Carry nicht berücksichtigt (without carry). Setzt man WC=1, so wird bei der Addition das Carry-Bit automatisch zum Ergebnis addiert oder bei einer Subtraktion automatisch vom Ergebnis subtrahiert (with carry). Unabhängig vom WC-Bit werden jedoch sowohl das Carry-Bit wie auch das Interdigit-Carry-Bit beim Abarbeiten der oben genannten drei Befehle immer automatisch berechnet.

Setzt man bei einer Rotate-Operation WC=0, dann werden lediglich die 8 Bits des angesprochenen Registers verschoben.

Setzt man WC=1, dann werden folgende Bits des PSL in die Rotate-Operation miteinbezogen:

- das Carry-Bit
- das Overflow-Bit
- das Interdigit-Carry-Bit.

Das Carry-Bit wird nun als 9. Bit in die Verschiebung miteinbezogen. Das OVF-Bit wird jedesmal gesetzt, wenn sich das Bit 7 (Vorzeichenbit) des Registers ändert. Das Interdigit-Carry-Bit enthält den aktuellen Wert von Bit 5.

Bit 2: Das OVERFLOW-Bit (OVF) wird während des Abarbeitens von Additionen oder Subtraktionen berechnet. Es wird gesetzt, wenn beide Operanden das gleiche, das Ergebnis aber das entgegengesetzte, Vorzeichen haben, wenn also ein Zweierkomplement-Überlauf auftritt. Operanden mit unterschiedlichen Vorzeichen können keinen Überlauf verursachen.

Beispiel:

$$\begin{array}{r} 124 \quad 0111\ 1100 \\ +64 \quad 0100\ 0000 \\ \hline 188 \quad 1011\ 1100 \quad (\triangleq -68) \end{array}$$

In Zweierkomplement-Darstellung würde diese Zahl fälschlicherweise als -68 interpretiert werden. Das Overflow-Bit wird gesetzt.

Das OVF-Bit wird auch gesetzt, wenn sich bei einem Rotierbefehl mit WC=1 das Bit 7 (Vorzeichen) ändert.

Bit 1: Das COMPARE-Bit (COM) bestimmt die Art des Vergleichs beim Abarbeiten von Compare-Befehlen (to compare, engl. vergleichen):

Ist COM=1
(logical mode), werden die Daten als positive 8-Bit-Zahl interpretiert.

Ist COM=0
(arithmetic mode), werden die Daten als 8-Bit-Zweierkomplementzahl interpretiert (D7=Vorzeichen).

Bit 0: Das CARRY-Bit (C, carry, engl. Übertrag) speichert beim Abarbeiten von Additions-, Subtraktions- oder Rotationsbefehlen den Übertrag aus dem Bit 7 des benutzten Arbeitsregisters.

Die Arbeitsregister R0, R1, R2, R3 und R1', R2', R3'

Alle Arbeitsvorgänge der ALU laufen über ein sogenanntes Arbeitsregister. Die Notwendigkeit geht aus Bild 7.2 hervor: Die ALU hat zwei Eingänge und einen Ausgang. Die zwei Eingänge sind nötig, weil die ALU ja im allgemeinen zwei Datenwörter (Bytes) miteinander arithmetisch oder logisch verknüpfen soll. Die beiden Datenwörter müssen der ALU gleichzeitig zur Verfügung stehen. Das wäre über zwei Datenbusse möglich, wenn auch sehr aufwendig. Denken Sie daran, daß der Mikroprozessor auch so schon 40 Anschlüsse braucht, und

daß dann auch noch zwei Datenbusse dazukämen, einer für den zweiten Eingang, ein anderer für den Ausgang - und diese Datenbusse müßten auch noch verwaltet werden.

Da Geschwindigkeit bekanntlich aber keine Hexerei ist, wird die Arbeit mit Hilfe schneller Zwischenspeicher, eben der Arbeitsregister, nacheinander erledigt. Zu diesem Zweck sind alle Mikroprozessoren mit mindestens einem, in der Regel aber mit einer ganzen Reihe von Arbeitsregistern ausgestattet. „Schnell“ heißt in diesem Zusammenhang, daß die ALU **direkt** auf diese Register zugreifen kann.

Die Register sind 8 Bit (=1 Byte) breit, so daß sie jeweils ein ganzes Datenwort aufnehmen können.

Der Arbeitsvorgang läuft im Prinzip so ab:

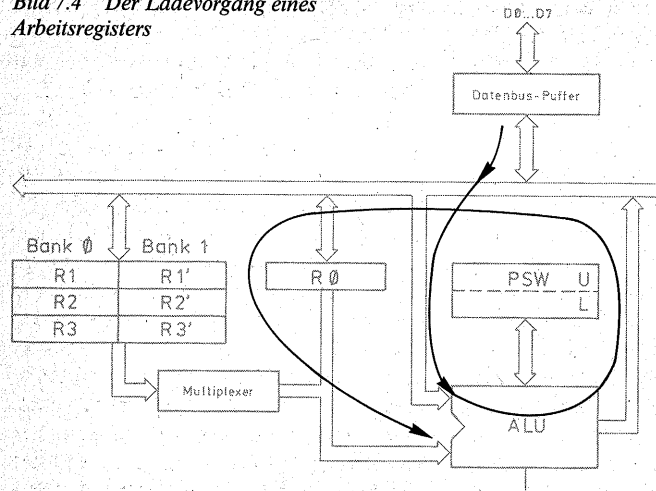
Das erste Datenwort wird auf den Datenbus gegeben, gelangt über den ersten (in Bild 7.4 oberen) Eingang in die ALU und wird durch sie über den internen Datenbus in ein Arbeitsregister geschoben. Dieser Vorgang heißt „laden“ (siehe Seite 118). Damit steht das 1. Byte für den zweiten Eingang (in Bild 7.4 unteren) der ALU zur Verfügung. Der Multiplexer, ein elektronischer UM-Schalter, entscheidet durch einen entsprechenden Befehl, in welches Register 1 Byte geladen wird und aus welchem 1 Byte an den Eingang der ALU gelangt.

Nun kann auf dem Datenbus das 2. Byte folgen, über den ersten Eingang in die ALU gelangen und dort mit dem 1. Byte verknüpft werden.

Das Ergebnis wird wieder in ein Arbeitsregister geschoben und steht dort zum Lesen oder zur weiteren Bearbeitung zur Verfügung.

Von den Arbeitsregistern nimmt eines in der Regel eine bevorzugte Stellung

Bild 7.4 Der Ladevorgang eines Arbeitsregisters



ein. Dieses heißt allgemein **Akkumulator**, (lat. Sammler), kurz **Akku**. In unserem Prozessor ist R0 der Akku. Bisweilen werden aber alle internen Arbeitsregister Akkumulatoren genannt.

Die übrigen sechs Register des 2650 sind auf zwei „Bänke“ verteilt. Die Bank 0 enthält die Register R1, R2 und R3, die Bank 1 enthält die Register R1', R2' und R3'.

Darüber, welche Registerbank benutzt wird, entscheidet das Bit 4 im **unteren** Programm-Status-Wort (PSL). Ist es auf 0 gesetzt, wird die Bank 0 benutzt, steht es auf 1, dann wird die Bank 1 benutzt. Diese Umschaltmöglichkeit bringt programmiertechnisch allerlei Vorteile, weil die Befehle für beide Bänke gleich sind, hat aber den Nachteil, daß der Programmierer immer nur eine Bank gleichzeitig benutzen kann. Der Akku (R0) steht unabhängig von der Wahl der Registerbank immer zur Verfügung.

Das Befehlsadreibregister (IAR)

Sehr wichtig für jede Programmierarbeit ist das Befehlsadreibregister IAR (instruction adress register). Es ist mehr als nur ein Register. Dieser Schaltungsblock enthält einen Dualzähler, den sogenannten **Programmzähler (PC, program counter)**.

Damit hat es folgende Bewandnis:

Die Befehle und Daten eines Programms sind im Speicher wie in fortlaufend nummerierten Schubladen untergebracht. Am Anfang steht der Programmzähler auf 0000_{16} . Der Prozessor holt sich („liest“) aus der Speicherstelle mit der Adresse 0000_{16} das 1. Byte und verarbeitet es. Dabei rückt der Programmzähler von 0000_{16} auf 0001_{16} , der Prozessor liest das nächste Byte aus dieser Adresse und verarbeitet es. Währenddessen rückt der Programmzähler von 0001_{16} auf 0002_{16} . Der Prozessor liest nun aus der Speicheradresse 0002_{16} die nächste Information, und so geht es Schritt um Schritt weiter. **Der Programmzähler adressiert immer die Speicherstelle,**

aus der sich der Prozessor das nächste Byte holen wird. Der Programmzähler steuert also die Reihenfolge der Programmschritte.

Wenn der Programmzähler wirklich bloß ein Zähler wäre, der von vorn beginnend nur Schritt um Schritt weiterrücken könnte, so könnten Programme nur linear abgearbeitet werden. Daher folgt auf den Zähler ein Register, das vom Zähler oder durch einen Befehl mit der nächsten zu lesenden Adresse geladen wird. Das bedeutet, daß die nächste zu lesende Adresse nicht die in der Zahlenreihe folgende sein muß. Das Besondere an dem Zähler ist, daß er durch die neue Adresse voreingestellt wird und von der neuen Adresse an weiterzählt. Auf diese Weise kann der Programmierer Sprünge und Schleifen (Wiederholungen von Programmteilen) einbauen.

Der Programmzähler zählt unmittelbar von 2^0 bis 2^{12} (genauer $2^{13} - 1$), das entspricht einem Umfang von 8 Kilobyte, 8 K. Ein K (Großbuchstabe) ist $2^{10} = 1024_{10}$ und nicht zu verwechseln mit k (Kleinbuchstabe = mal 1000). Die Adreßleitungen A0 bis A12 adressieren einen Bereich von 0 bis 8191_{10} .

Außerdem stehen noch die Adreßleitungen A13 (E/\overline{NE}) und A14 (D/\overline{C}) zur Verfügung, mit denen der Adreßbereich auf 32 K vervierfacht werden kann (siehe auch Speicherorganisation Seite 150). Und wer damit nicht auskommt, kann durch eine Zusatzlogik den Adressierbereich darüber hinaus erweitern. Doch seien Sie unbequem, Sie werden sich wundern, wie lange Sie brauchen, um auch nur ein einziges „K“ zu adressieren.

Der Stack

Der Stack (engl. Stapel) ist ein Speicher mit acht Registern zu je 15 Bits, 15 Bits deswegen, weil eine vollständige Adresse mit 15 Bits, nämlich A0 bis A14, angegeben werden muß. Dieser Stapelspeicher ist ein LIFO-Speicher (last in, first out; engl. was zuletzt eingegeben wurde, wird zuerst ausgegeben). Er nimmt die Rückkehradressen bei Unterprogrammen auf. Wenn ein Programm durch Einschleichen eines Unterprogramms unterbrochen wird, teilt der Stack nach Abarbeiten des Unterprogramms dem Befehlsadreibregister mit, an welcher Stelle das Hauptprogramm weiterläuft. Der Stack kann sich acht Adressen merken, d.h. daß acht Unterprogramme ineinander verschachtelt werden können. Das Hauptprogramm enthält ein Unterprogramm, das Unterprogramm enthält wieder ein „Unter-Unterprogramm“ usw., und das acht Ebenen tief. Wer diese Möglichkeiten voll ausnutzen will, muß schon ein bißchen um die Ecke denken können. Die technischen Möglichkeiten sind jedenfalls vorhanden.

Der Stack hat als eigenständiger Speicher auch einen eigenen Programmzähler, den **Stackpointer**. Der braucht, weil er nur acht Speicherstellen zu verwalten hat, auch nur von 0_{10} bis 7_{10} zählen zu können und kommt daher mit den Stellen 2^0 bis 2^2 aus. Der gegenwärtige Zählerstand, der zugleich die Schachtebene angibt, ist im **oberen** Programm-Status-Wort (PSU) in den Bits 0 (2^0), 1 (2^1) und 2 (2^2) enthalten und kann dort abgefragt werden.

Das Operandenadreibregister (OAR)

In diesem Register steht die Adresse

des zweiten Operanden einer Operation bzw. es werden Zwischenergebnisse bei der Berechnung von effektiven Adressen abgespeichert.

Die Ablaufsteuerung

Der übrige Zweig der Struktur in Bild 7.2, nämlich das Befehlshalterregister, der Befehlsdecoder, die zentrale Ablaufsteuerung, die Ein-/Ausgabeverwaltung und die Interruptlogik dient der Koordination aller Abläufe. Im **Befehlshalterregister** wird jeder eingehende Befehl bis zur Beendigung des Befehls festgehalten, denn auf dem Datenbus erscheint er ja nur für kurze Zeit. Im **Befehlsdecoder**, der im wesentlichen aus einem Festwertspeicher besteht, wird er in eine Anzahl von Steuersignalen umgesetzt, die in der **zentralen Ablaufsteuerung** mit den eingehenden oder abgegebenen Steuersignalen das Gesamtgeschehen beeinflussen, wozu auch die **Ein-/Ausgabeverwaltung** gehört (z. B. die Bestimmung der Zeiten, wann Daten angenommen oder abgegeben werden) sowie die **Interruptlogik**, die bestimmt, ob eine geforderte Programmunterbrechung (Interrupt) angenommen werden kann oder nicht. Was da intern geschieht, dürften wahrscheinlich nur die Schaltungsentwickler ganz genau wissen. Für Sie ist es nicht so wichtig, denn Sie dürfen sich darauf verlassen, daß die Koordination funktioniert. Einfluß auf das Geschehen nehmen oder es auswerten können Sie über die Steuerleitungen, die Sie unten im Bild 7.2 finden. Die Pfeile geben an, ob es sich jeweils um einen Ein- oder einen Ausgang handelt.

Die Anschlüsse des 2650

Bild 7.5 zeigt die Anschlußbelegung des 2650. Die Pfeile geben wieder die Richtung des Signalfusses an.

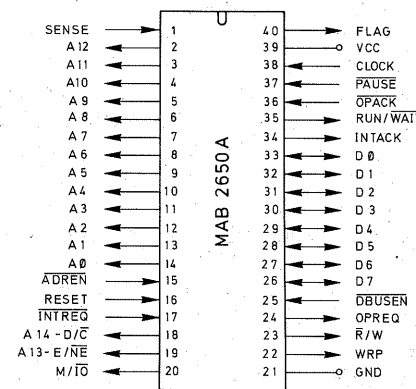


Bild 7.5 Die Anschlußbelegung des Mikroprozessors 2650

Die Spannungsversorgung

Die Versorgungsspannung von $5V \pm 5\%$ wird an Pin 21, GND (Minuspol) und Pin 39, VCC (Pluspol) angeschlossen. Die Spannung muß gut stabilisiert und mit Abblockkondensatoren gegen Störimpulse geschützt sein. Die Leistungsaufnahme beträgt maximal 525 mW.

Der Datenbus (Anschlüsse 26 bis 33)

Die Numerierung der Datenbusleitungen D0 bis D7 gibt deren Stellenwert im Dualsystem an (siehe Seite 95). Der Datenbus stellt einen bidirektionalen 8-Bit-Datenweg zwischen Speichern, Ein-/Ausgabeeinheiten und dem Mikroprozessor dar. Die Richtung des Datenflusses auf dem Datenbus wird durch den Zu-

stand der \bar{R}/W -Leitung angegeben. Bei Schreiboperationen (Write, Store) legt der Prozessor ein Datenbyte auf den Datenbus. Bei Leseoperationen (Read, Load) erwartet der Prozessor auf dem Datenbus ein Datenbyte von peripheren Bausteinen.

FLAG (Anschluß 40) und SENSE (Anschluß 1)

Beim 2650 gibt es die Möglichkeit, serielle Daten – jeweils 1 Bit nacheinander – direkt ein- bzw. auszugeben, ohne Zwischenschaltung eines Registers und ohne Steuerleitungen decodieren zu müssen.

Das FLAG-Bit im Programm-Status-Wort (Bit 6 im PSU) ist direkt mit Pin 40 des Mikroprozessors verbunden. Der FLAG-Ausgang repräsentiert immer den Wert des FLAG-Bits. Das SENSE-Bit ist die komplementäre Funktion zum FLAG-Bit und stellt einen 1-Bit-Eingang in den Mikroprozessor dar. Der SENSE-Eingang (Pin 1) führt direkt zum SENSE-Bit im oberen Programm-Status-Wort (Bit 7 im PSU), wo es immer den Wert des externen Signals wiedergibt.

Der Adreßbus (Anschlüsse 2 bis 14 für A0 bis A12, 19 für A13 und 18 für A14)

Die Numerierung der Leitungen gibt ihren Rang im Dualsystem an. Der Adreßbus stellt einen 15-Bit-Datenweg aus dem Mikroprozessor dar, der vorwiegend für die Adressierung von Speicherstellen bei Speicheroperationen verwendet wird. Die Adressen am Adreßbus sind während der aktiven Phase von OPREQ gültig, so daß kein zusätzliches Adreßregister benötigt wird.

Bei Extended-I/O-Operationen (I/O für input/output, engl. Ein-/Ausgabe) stellen die 8 niedrigerwertigen

Bits des Adreßbusses das 2. Byte des Befehls, die sogenannte I/O-Adresse, dar.

Die 13 niedrigerwertigen Leitungen A0 bis A12 übertragen nur Adreßinformationen. Die verbleibenden 2 höchstwertigen Bits (A13, A14) repräsentieren bei I/O-Operationen die Leitungen D/C bzw. E/NE. Bei Speicheroperationen enthalten sie die sogenannte Page-Adresse (page, engl. Seite; siehe Seite 149, Speicherorganisation).

Die Signale am Adreßbus sind aktiv H.

Die Steuerleitungen

RESET (Anschluß 16)

Durch das RESET-Signal (aktiv H) wird der Prozessor in einen definierten Anfangszustand gebracht. Nach dem Einschalten der Betriebsspannung und nach der Aktivierung des RESET-Signals wird das Interrupt-Inhibit-Bit gelöscht und der Programmzähler auf 0000_{16} gestellt. Das RESET-Signal darf asynchron zum Takt sein und muß mindestens drei Taktperioden lang anliegen. Wird das RESET-Signal als Startsignal verwendet, so wird nach dessen Rückkehr auf L der Befehl des ersten Speicherplatzes (Adresse 0000_{16}) ausgeführt. Wenn RESET und eine Interruptanforderung gleichzeitig vorliegen, wird der Interrupt nach Wegfall des RESET-Signals durchgeführt.

CLOCK (Anschluß 38)

Das CLOCK-Signal (Taktsignal) koordiniert sämtliche Vorgänge des Mikroprozessors. Mit der ansteigenden Flanke dieses Signals beginnt die Befehlsausführung. Ein Befehl besteht aus mehreren Prozessorzyklen („Maschinenzyklen“), von denen jeder drei

Takte lang ist. Befehle benötigen zwei, drei oder vier Prozessorzyklen, bei indirekter Adressierung kommen noch zwei weitere dazu.

Die H-Zeit und die L-Zeit des CLOCK-Signals muß jeweils mindestens 400 ns (beim 2650 A-1 250 ns) betragen. Eine CLOCK-Periode muß daher mindestens 800 ns bzw. 500 ns dauern. Damit liegt die höchste CLOCK-Frequenz mit 1,25 MHz, bzw. 2 MHz beim 2650 A-1, fest. Der 2650 kommt mit einem einfachen CLOCK-Signal aus. Das ist durchaus nicht selbstverständlich. Andere Prozessoren benötigen z.B. 2-Phasen-Taktsignale, was die Schaltungstechnik wieder erschwert. Der wesentliche Vorteil des 2650 gegenüber anderen Mikroprozessoren ist aber, daß die CLOCK-Zeiten beliebig lang sein dürfen, daß man sich also in Ruhe jede Veränderung, die sich nach einem CLOCK-Impuls oder währenddessen ergibt, ansehen kann.

PAUSE (Anschluß 37)

Mit Hilfe des PAUSE-Signals kann man ein laufendes Programm für eine bestimmte Zeit anhalten. Durch ein L-Signal an der PAUSE-Leitung, die normalerweise auf H liegt, beendet der Prozessor den laufenden Befehl und geht in den WAIT-Zustand über (to wait, engl. warten). Mit „Befehl“ ist ein ganzer Befehl gemeint. Bei 2-Byte- oder 3-Byte-Befehlen werden sämtliche zum Befehl gehörenden Bytes abgearbeitet.

Sobald am PAUSE-Eingang wieder H anliegt, wird die Abarbeitung des Programms beim nächstfolgenden Befehl fortgesetzt.

Wird die PAUSE-Leitung während der Dauer eines Befehls vorübergehend aktiv (L) und wiederum H, so erfolgt keine Unterbrechung.

Bei gleichzeitigem Auftreten des PAUSE-Signals und der Interruptanforderung INTREQ geht der Prozessor zuerst in den WAIT-Zustand über. Erst nach Wegnahme des PAUSE-Signals wird der Interrupt angenommen. Ein Interrupt wird auch dann angenommen, wenn er während der WAIT-Phase eintrifft.

RUN/WAIT (Anschluß 35)

Der RUN/WAIT-Ausgang (to run, engl. laufen) gibt den Betriebszustand des Prozessors an. Der Mikroprozessor kann durch den HALT-Befehl oder durch Anlegen eines L-Signals in die PAUSE-Leitung in den WAIT-Zustand gebracht werden. Hierbei geht der RUN/WAIT-Ausgang auf L. In allen anderen Fällen ist RUN/WAIT auf H.

Der Wait-Zustand kann durch das RESET-Signal oder durch einen Interrupt aufgehoben werden.

INTREQ (Anschluß 17)

Der INTREQ-Eingang (interrupt-request, engl. Interruptanforderung) gestattet peripheren Einheiten, die Programmabarbeitung zu unterbrechen. Bei einem L an der INTREQ-Leitung beendet der Prozessor den gerade in Ausführung befindlichen Befehl. Dann lädt er den Befehl ZBSR (Zero Branch to Subroutine, Relative) in das Befehlsregister, setzt das Interrupt-Inhibit-Bit (II) im oberen Programm-Status-Wort (PSU) und aktiviert die Signale INTACK und OPREQ. Wenn die periphere Einheit, die den Interrupt ausgelöst hat, das Signal INTACK empfangen hat, muß sie das Anforderungssignal INTREQ auf H setzen (aktiv L). Gleichzeitig muß 1 Datenbyte (der Interruptvektor) an den Datenbus gelegt werden. Der Prozessor über-

nimmt dieses Byte mit den Steuersignalen einer I/O-Operation, es entspricht dem 2. Byte des relativen Sprungbefehls (ZBSR). Bei indirekter Adressierung kann somit der gesamte Speicherbereich (32 Kbyte) angesprochen werden. Wird von der Peripherie kein Interruptvektor an den Datenbus gelegt, so wird automatisch der Vektor 00₁₆ genommen.

Da nach Annahme des Interrupts sofort das Interrupt-Inhibit-Bit (II) im Programm-Status-Wort gesetzt wird, bleibt der Prozessor so lange für weitere Interrupts gesperrt, bis das II-Bit gelöscht wird. Dies kann entweder durch Befehle geschehen, die das Programm-Status-Wort ändern, oder über einen „Return and Enable“-Befehl. Wird das II-Bit erst während einer Interruptanforderung gelöscht, so wird der Interrupt trotzdem angenommen.

INTACK (Anschluß 34)

Das INTACK-Signal (interrupt acknowledge, engl. Interruptquittung) ist die Antwort des Prozessors auf die Interruptanforderung von außen. Hat der Prozessor den Interrupt angenommen, sendet er das Signal INTACK und das Signal OPREQ aus. Die periphere Einheit empfängt INTACK und muß nun gleichzeitig die Interruptanforderung INTREQ auf H setzen (aktiv L). Das INTACK-Signal wird gelöscht, sobald der Prozessor das OPACK-Signal von der externen Einheit empfangen hat.

M/IO (Anschluß 20)

Der M/IO-Ausgang (M von Memory, engl. Speicher; IO von input/output, engl. Ein-/Ausgabe) gibt an, ob die Schreib- bzw. Leseoperationen zwischen Speicher und Mikroprozessor (M aktiv H) oder zwischen Input/

Output-Einheiten und Mikroprozessor (IO aktiv L) erfolgen soll. Während der aktiven Phase von OPREQ bleibt M/IO unverändert.

OPREQ (Anschluß 24)

Das OPREQ-Signal (operation request, engl. Arbeitsaufforderung) koordiniert die Kontrollsignale für alle externen Operationen. Die Signale M/IO, R/W, E/NE, D/C und INTACK beschreiben die Art der externen Operationen, aber nur, solange das OPREQ-Signal aktiv ist. Auch die Signale am Daten- und Adreßbus sind nur dann gültig, wenn auch OPREQ anliegt. Solange der OPREQ-Ausgang auf H liegt, ändern sich mit einer Ausnahme, dem Write-Pulse (WRP), die Ausgangssignale nicht. Das OPREQ-Signal ist so lange aktiv, bis die Operation von einer externen Einheit angenommen ist und diese mit dem Signal OPACK geantwortet hat (Hand-Shake-Verfahren). Zwischen den beiden Signalen OPREQ und OPACK ruht der processorinterne Betrieb.

OPACK (Anschluß 36)

Das OPACK-Signal (operation acknowledge, engl. Quittierung der Operation) ist die „Hand-Shake“-Antwort (hand-shake, engl. Händedruck, Quittung) auf das OPREQ-Signal und muß vom Speicher oder einer Input/Output-Einheit erzeugt werden. Mit dem Signal OPACK zeigt eine Einheit, z. B. ein Drucker, dem Prozessor an, daß sie den vorangegangenen Vorgang abgeschlossen hat und nun für die nächste Operation bereit ist. In der Zeit zwischen der Aussendung von OPREQ und dem ersten Taktimpuls nach der Anlieferung von OPACK ruht die Verarbeitung im

Prozessor. Dieses System gestattet die asynchrone Arbeitsweise mit externen Einheiten (Hand-Shake-Verfahren). Diese Möglichkeit ist deswegen so wichtig, weil viele externe Einheiten viel langsamer arbeiten als der Prozessor, z. B. Stellmotoren, Drucker, Fahrstühle, Werkzeugmaschinen usw.

Kann sichergestellt werden, daß alle peripheren Einheiten eines Systems genügend schnell sind, d. h. daß das $\overline{\text{OPACK}}$ -Signal spätestens 600 ns nach OPREQ eintreffen würde, dann kann die $\overline{\text{OPACK}}$ -Leitung dauernd auf L gelegt werden. Das ist z. B. dann der Fall, wenn der Prozessor nur eine Anzeigeeinheit zu bedienen hat.

$\overline{\text{R/W}}$ (Anschluß 23)

Der $\overline{\text{R/W}}$ -Ausgang (read/write, engl. lesen/schreiben) gibt die Richtung der Datenübertragung auf dem Datenbus an. Soll die Information von der Peripherie in den Mikroprozessor übertragen werden, (Read), so ist $\overline{\text{R/W}} = \text{L}$. Soll die Information vom Mikroprozessor zur Peripherie gelangen (Write), so ist $\overline{\text{R/W}} = \text{H}$. Während der aktiven Phase von OPREQ wird $\overline{\text{R/W}}$ nicht verändert.

WRP (Anschluß 22)

Der WRP -Ausgang (write pulse, engl. Schreibpuls) liefert einen Synchronisierimpuls, der (nur bei Schreiboperationen) in der Mitte von OPREQ vom Prozessor ausgesendet wird. Bei allen anderen Operationen ist das Signal H (Bild 7.10).

$\overline{\text{DBUSEN}}$ (Anschluß 25)

Durch H am $\overline{\text{DBUSEN}}$ -Eingang (data bus enable, engl. Datenbusfreigabe) können alle Datenbusleitungen in den hochohmigen (Tri-State) Zustand gebracht werden (siehe Seite 64).

Durch L an diesem Eingang werden die Datenleitungen für den Datenfluß freigegeben.

$\overline{\text{ADREN}}$ (Anschluß 15)

Durch H am $\overline{\text{ADREN}}$ -Eingang (address enable, engl. Adressenfreigabe) können die Adreßleitungen A0 bis A12 in den hochohmigen (Tri-State) Zustand gebracht werden. Durch L an diesem Eingang wird der Stand des Programmzählers auf den Adreßbus gegeben.

Die Adreßleitungen A13 und A14, welche mit $\text{E}/\overline{\text{NE}}$ bzw. $\text{D}/\overline{\text{C}}$ im Multiplex-Betrieb arbeiten, werden durch $\overline{\text{ADREN}}$ nicht beeinflusst.

$\text{D}/\overline{\text{C}}$ (Anschluß 18)

Durch den $\text{D}/\overline{\text{C}}$ -Ausgang wird zwischen den zwei Möglichkeiten von „non extended I/O“-Befehlen, nämlich „Data“ ($\text{D}/\overline{\text{C}} = \text{H}$) und „Control“ ($\text{D}/\overline{\text{C}} = \text{L}$), unterschieden.

$\text{D}/\overline{\text{C}}$ ist gültig und hat die obige Bedeutung, wenn OPREQ aktiv ist sowie $\text{M}/\overline{\text{IO}}$ auf $\overline{\text{IO}}$ und $\text{E}/\overline{\text{NE}}$ auf $\overline{\text{NE}}$ stehen. Steht hingegen $\text{M}/\overline{\text{IO}}$ auf M , so repräsentiert die $\text{D}/\overline{\text{C}}$ -Leitung, so lange OPREQ aktiv ist, das Adreß-Bit 14.

$\text{E}/\overline{\text{NE}}$ (Anschluß 19)

Durch den $\text{E}/\overline{\text{NE}}$ -Ausgang (extended/not extended, engl. erweitert/nicht erweitert) wird zwischen 1-Byte- und 2-Byte-I/O-Befehlen unterschieden.

Ist $\text{E}/\overline{\text{NE}} = \text{L}$, handelt es sich um 1-Byte-I/O-Befehle (REDD, WRD, REDC, WRTC), und der Adreßbus enthält keine gültige Information.

Ist $\text{E}/\overline{\text{NE}} = \text{H}$, handelt es sich um 2-Byte-I/O-Befehle (REDE, WRTE). Die acht niedrigerwertigen Bits des Adreßbusses bestimmen die Adresse des I/O-Ports.

$\text{E}/\overline{\text{NE}}$ ist gültig und hat die obige Bedeutung, wenn OPREQ aktiv ist und $\text{M}/\overline{\text{IO}}$ auf $\overline{\text{IO}}$ steht.

Liegt hingegen $\text{M}/\overline{\text{IO}}$ auf M , so repräsentiert die $\text{E}/\overline{\text{NE}}$ -Leitung das Adreß-Bit 13, solange OPREQ aktiv ist.

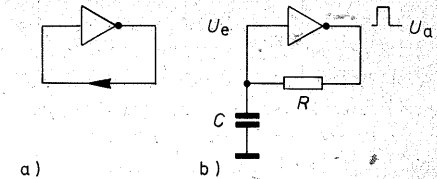


Bild 7.6 a) Rückgekoppelter Inverter, b) rückgekoppelter Inverter mit RC-Glied, zur Herabsetzung der Schwingfrequenz

Schaltungstechnik

Allein kann der Mikroprozessor noch nicht mit seinen Partnern „Speicher“ und „Peripherie“ zusammenarbeiten. Er benötigt einige Hilfen zu seiner eigenen Steuerung sowie zur Steuerung der ihn umgebenden Einheiten. Es ist durchaus keine Schwäche, daß diese wenigen Gatter, die im Vergleich zu der großen Anzahl, die der Prozessor enthält, geradezu ein Nichts sind, noch hinzukommen müssen. Gerade die Möglichkeit der Außenbeschaltung macht den Prozessor zum Aufbau unterschiedlicher Systeme sehr flexibel.

Der CLOCK-Oszillator

Unentbehrlich ist in jedem Fall ein Taktsignal. Dieses wird der Einfachheit halber von einem RC-Generator erzeugt. Bild 7.6a zeigt das Prinzip mit einem rückgekoppelten Inverter: Wenn man den Ausgang eines invertierenden Schaltgliedes – es darf auch ein NAND- oder NOR-Gatter sein – auf den Eingang rückkoppelt, gerät der Inverter gewissermaßen in einen Verhaltenskonflikt.

Angenommen, am Inverterausgang steht ein H, das auf den Eingang zurückgeleitet wird, so zwingt es den Ausgang auf L. Dieses L erscheint durch die Rückkopplung wieder am Eingang und zwingt den Ausgang er-

neut auf H, und so geht es in unaufrührlichem Wechsel weiter. Der rückgekoppelte Inverter ist zu einem **Oszillator** (lat. Schwinger) geworden.

Wie schnell sich der Signalwechsel am Ausgang vollzieht, d. h. wie hoch die Oszillatorfrequenz ist, hängt von der Signalverzögerungszeit im Gatter ab, also von der Zeit, die der Ausgang braucht, um auf eine Veränderung am Eingang zu reagieren. Sie liegt beim 74(LS)14 in der Größenordnung von 20 ns. Ihr entspricht eine Oszillatorfrequenz von ca. 20 bis 25 MHz.

Dieser einfache Oszillator funktioniert übrigens nur, wenn der Inverter einen genügend hohen Verstärkungsfaktor hat. Mit einem normalen TTL-Inverter (z. B. 7404) geht es oft nicht. Wenn man aber mehrere hintereinanderschaltet, und zwar eine ungerade Anzahl (siehe Seite 47), z. B. drei, bekommt man den Oszillator auch zum Schwingen. Inverter mit einem Schmitt-Trigger-Eingang (siehe Seite 53) erfüllen die Verstärkungsbedingung. Daher wurde für den CLOCK-Oszillator der Baustein 74LS14 ausgewählt.

Durch ein RC-Glied (Bild 7.6b) läßt sich die Oszillatorfrequenz beliebig herabsetzen. Angenommen, der Ausgang hat H-Signal, dann lädt sich C über R auf. Sobald die Kondensatorspannung die Schaltschwelle V_{T+} (ca.

1,6 V) erreicht hat, schaltet der Inverter um, und der Ausgang fällt auf L. Nun entlädt sich C über R und den Ausgang. Wenn die Ladespannung von C die Schaltschwelle V_{T-} (ca. 0,8 V) erreicht hat, springt der Ausgang wieder auf H (siehe Hystereseschleife des 74(LS)14 in Bild 2.19b). Bild 7.7 zeigt die Spannungsverläufe am Ausgang und am Eingang des Inverters. Auch diese Gegenüberstellung macht deutlich, daß dieser Ein-Inverter-Oszillator nur mit sehr hoher Spannungsverstärkung oder einem Schmitt-Trigger zu verwirklichen ist, weil der Ausgang blitzschnell von einem Zustand in den anderen springen muß.

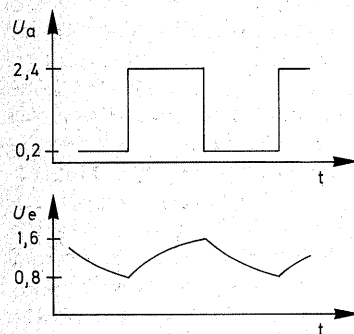


Bild 7.7 Ausgangsspannung (U_a) und Eingangsspannung (U_e) eines Schmitt-Trigger-RC-Oszillators

Der Rückkopplungswiderstand darf nicht beliebig groß sein; bei Standard-TTL-Bausteinen darf er wegen des vergleichsweise hohen L-Eingangstroms $1\text{ k}\Omega$ nicht überschreiten, bei LS-Bausteinen nicht den vierfachen Wert davon. Der Kapazität des Kondensators sind keine Grenzen gesetzt. Die Schwingfrequenz richtet sich nach der Zeitkonstanten des RC-

Gliedes und läßt sich annähernd nach der Formel

$$f \approx \frac{0,7}{R \cdot C} \quad [R \text{ in } \Omega; C \text{ in F}]$$

berechnen. Sie hängt auch weitgehend vom Abstand der Schaltschwellen ab, und der ist, fertigungstechnisch bedingt, weiten Toleranzen unterworfen.

Sehr hohe Anforderungen darf man an die Frequenzstabilität dieses Oszillators nicht stellen. Als Zeitnormal für eine Uhr, die über lange Zeit genau gehen soll, reicht seine Stabilität nicht aus, für den „normalen“ Prozessorbetrieb zum Spielen, Messen, Steuern, Regeln usw. aber allemal. C5 sollte ein guter Folienkondensator sein, R5 (und möglichst auch R6) ein Metallfilmwiderstand.

Bild 7.8 enthält die komplette Oszillatorschaltung. Durch den UM-Schalter S2 lassen sich zwei Frequenzen einstellen, ca. 2 MHz und ca. 3 Hz. Da das Tastverhältnis dieses RC-Oszillators sehr unausgewogen sein kann, wird das Oszillatorsignal durch N9 in garantiert saubere Rechtecksignale geformt und von FF1 durch zwei geteilt. So entstehen die CLOCK-Frequenzen 1 MHz bzw. 1,5 Hz mit genau gleich langen H- bzw. L-Zeiten. Dadurch kann die höchstmögliche CLOCK-Frequenz von 1,25 MHz ausgenutzt werden, denn es ist ja sichergestellt, daß H- bzw. L-Zeit je 400 ns dauern. Bei einem unausgewogenen Tastverhältnis müßte man die CLOCK-Frequenz so erniedrigen, daß die kürzere der beiden Zeiten mindestens 400 ns dauert.

Für die hohe Frequenz enthält der Oszillator einen Trimmerwiderstand, damit die CLOCK-Frequenz auf 1 MHz abzugleichen ist.

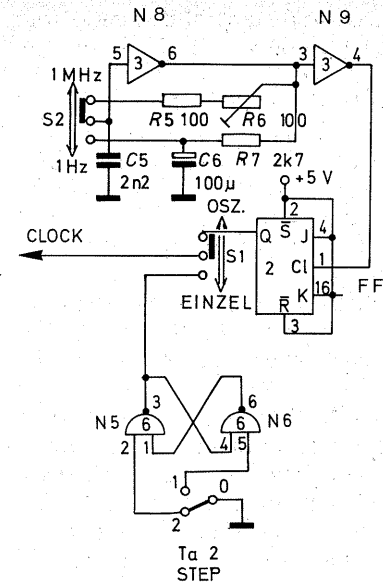


Bild 7.8 Die Oszillatorschaltung der CPU-Einheit

Für die niedrige Taktfrequenz, die es ermöglichen soll, dem Mikroprozessor bei der Arbeit zuzusehen, braucht man eine hohe Zeitkonstante. Die Verwendung eines LS-Bausteins erlaubt es, für R7 einen Widerstand mit einem hohen Ohmwert einzusetzen. Mit $R7 = 2,2\text{ k}\Omega$ schwingt der Oszillator sicher an. Wenn Ihnen die Frequenz zu hoch ist, weil Sie alles lieber etwas langsamer hätten, können Sie R7 vergrößern ($2,7\text{ k}\Omega$; $3,3\text{ k}\Omega$). Manche Bausteine schwingen noch mit $R7 = 3,9\text{ k}\Omega$ sicher an, andere aber nicht mehr mit $R7 = 2,7\text{ k}\Omega$. Sie können aber auch C6 vergrößern, um den Oszillator langsamer schwingen zu lassen.

Es hieße, die guten Eigenschaften des 2650 schlecht auszunutzen, wenn man nicht eine Einzeltakteingabe vorsähe. Dazu dient der Taster Ta 2

(STEP, engl. Schritt) mit dem Entprell-Flip-Flop aus N5 und N6 (siehe Seite 56). Mit dem Schalter S1 kann man zwischen Oszillator-CLOCK oder Einzeltakt umschalten. Die Einzeltakteingabe ermöglicht es, den Prozessor mitten in einem Maschinenzklus, ja mitten in einer CLOCK-Periode, festzuhalten.

Die Decodierung der Steuersignale ADMEM und ADPER

Das Steuersignal $\overline{\text{ADMEM}}$ (Adressen Memory) dient dazu, Speicherbausteine zu aktivieren, das Steuersignal ADPER (Adressen der Peripherie) soll Peripheriebausteine aktivieren (siehe Seite 103). Die meisten dieser Bausteine erfordern ein aktiv-L-Signal, weswegen diese Signale auf der CPU-Platte aktiv-L decodiert werden. Mit dem Ausgang $M/\overline{\text{IO}}$ (Anschluß 20) zeigt der Prozessor an, ob er sich an den Speicher (Memory) oder an die Peripherie (Input Output) wendet. Ist $M/\overline{\text{IO}} = H$, zeigt er auf den Speicher, ist $M/\overline{\text{IO}} = L$, zeigt er auf die Peripherie. Beide Signale sind aber nur gültig, wenn OPREQ (Anschluß 24) = H ist. Daher müssen $M/\overline{\text{IO}}$ und OPREQ durch UND verknüpft werden (Bild 7.9).

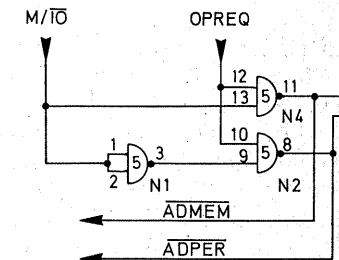


Bild 7.9 Die Decodierung der Steuersignale $\overline{\text{ADMEM}}$ und ADPER

Die Decodierung von $\overline{\text{ADMEM}}$: N4 verknüpft $\overline{\text{OPREQ}}$ UND M (aktiv H). Da die Verknüpfung durch ein NAND-Gatter geschieht, steht am Ausgang von N4 gleich die invertierte Form des Speichersignals, also $\overline{\text{ADMEM}}$.

Die Decodierung von $\overline{\text{ADPER}}$: Der Prozessor zeigt durch L auf dem Ausgang $\overline{\text{M}/\overline{\text{IO}}}$ an, daß er sich an die Peripherie wenden will. Zur $\overline{\text{UND}}$ -Verknüpfung muß das L des $\overline{\text{IO}}$ in ein H invertiert werden, das geschieht durch N1. Nach der NAND-Verknüpfung durch N2 steht $\overline{\text{ADPER}}$ zur Verfügung.

$\overline{\text{ADMEM}}$ und $\overline{\text{ADPER}}$ sind gegeneinander verriegelt: Wenn $\overline{\text{M}/\overline{\text{IO}}} = \text{H}$, ist N2 gesperrt, weil N1 dann ein L an N2 liefert.

Wenn $\overline{\text{M}/\overline{\text{IO}}} = \text{L}$ ist, ist N4 gesperrt, und N2 ist durch N1, das H am Ausgang hat, geöffnet.

Die Erzeugung von $\overline{\text{WRITE}}$

Der Prozessor liefert das Schreib-/Lesesignal als $\overline{\text{R}/\text{W}}$ am Anschluß 23 (Bild 7.12). Es wird aber als $\overline{\text{READ}}/\overline{\text{WRITE}}$, also invertiert, benötigt. Daher folgt auf den Ausgang $\overline{\text{R}/\text{W}}$ ein Inverter (N7). Die „verkehrte“ Signalbelegung am Prozessor wurde von den Schaltungsentwicklern sicherlich wohl bedacht: Das fan out am Prozessor muß – bedingt durch die Technologie – sehr klein sein (1 TTL Last, siehe Seite 64). Da diese Steuerleitung aber eine Reihe von Bausteinen zu treiben hat, muß sie gepuffert werden. Invertierende Puffer (Inverter, NAND- oder NOR-Gatter) mit größerem fan out stehen bei jedem Schaltungsaufbau zur Verfügung. Daher ist das gepufferte Signal dann das benötigte „richtige“.

Da das $\overline{\text{WRITE}}$ -Signal auch auf der

Dateneingabe- und -anzeige (Baugruppe 3) erzeugt werden soll und die Signalleitungen zusammenschaltet werden sollen, muß N7 einen Open-Collector-Ausgang haben (siehe Seiten 61 und 62).

Das $\overline{\text{RESET}}$ -Signal

Der Prozessor braucht am Anschluß 16 für den Betrieb ein Dauer-L, das nur zum Rückstellen für die Dauer von drei $\overline{\text{CLOCK}}$ -Perioden auf H geht (Bild 7.12). Der Eingang des vorgeschalteten Inverters liegt bei geöffnetem $\overline{\text{RESET}}$ -Taster Ta 1 über R1 an H. Der Ausgang des Inverters ist deswegen L, und der Prozessor kann laufen. Beim Betätigen des Tasters Ta 1 wird der Eingang des Inverters auf L gezogen, der Ausgang springt auf H, und dadurch erhält der Prozessor sein $\overline{\text{RESET}}$ -Signal (aktiv H). Auf dem Control-Bus ist die $\overline{\text{RESET}}$ -Leitung aber in Übereinstimmung mit den anderen Steuerleitungen aktiv L.

R1 bildet mit C3 ein sogenanntes Auto- $\overline{\text{RESET}}$ (autós, griech. eigen-, selbst-): Wenn die Betriebsspannung eingeschaltet wird, ist C3 zunächst leer und zieht daher den Eingang des Inverters auf L. Der Prozessor erhält dadurch ein $\overline{\text{RESET}}$ -Signal, das sich beim Einschalten der Betriebsspannung von selbst einstellt. C3 lädt sich über R1 auf, und wenn die Ladespannung die Schaltschwelle des Inverters erreicht hat, erkennt sein Eingang H, der Ausgang schaltet auf L, und der Prozessor beginnt seine Arbeit definiert bei der Adresse 0000_{16} . Dieses Auto- $\overline{\text{RESET}}$ funktioniert nur bei der schnellen $\overline{\text{CLOCK}}$ -Frequenz von 1 MHz, weil die Zeitkonstante aus R1/C3 nur etwa 0,1 s beträgt. Sollte das Auto- $\overline{\text{RESET}}$ auch bei der 1-Hz-

$\overline{\text{CLOCK}}$ -Frequenz funktionieren, so müßte sie mindestens 3 s betragen; C3 wäre dann auf 330 μF zu vergrößern. Wenn in einer Baugruppe verschiedene Inverter zur Verfügung stehen, „normale“ TTL, o.c.-TTL oder Schmitt-Trigger, dann wird man für einen solchen Umschaltvorgang einen Schmitt-Trigger-Inverter bevorzugen.

Das $\overline{\text{PAUSE}}$ -Signal

Der Anschluß $\overline{\text{PAUSE}}$ liegt über R4, bei offenem Schalter S4, auf H (Stellung $\overline{\text{RUN}}$ in Bild 7.12). Damit ist der Prozessor für die Arbeit frei. Wenn S4 geschlossen ist und damit $\overline{\text{PAUSE}}$ auf L zieht, arbeitet der Prozessor den gerade laufenden Befehl ab und geht in den $\overline{\text{WAIT}}$ -Zustand über. Aus diesem kann er nur durch ein $\overline{\text{RESET}}$ oder eine Interruptanforderung (L an $\overline{\text{INTREQ}}$, Anschluß 17) erlöst werden.

Das $\overline{\text{INTREQ}}$ -Signal

Für den Betrieb muß die Leitung $\overline{\text{INTREQ}}$ (Anschluß 17) auf H stehen. Sie ist daher über R2 (Bild 7.12) mit VCC verbunden. Da die Interruptanforderung von außen kommen muß, ist die $\overline{\text{INTREQ}}$ -Leitung sowohl auf die Busplatte als auch an die 7polige Zusatzbuchse auf der Platine geführt. R2 ist mit 10 k Ω recht hoch gewählt, damit die Leitung gegebenenfalls auch von Schaltgliedern mit geringem fan out auf L gezogen werden kann.

Das $\overline{\text{OPACK}}$ -Signal

Der Prozessor arbeitet erst dann weiter, wenn nach dem Eintreffen des $\overline{\text{OPREQ}}$ -Signals ein L am $\overline{\text{OPACK}}$ -Eingang (Anschluß 36 in Bild 7.12) erscheint. Wenn keine „langsamen“

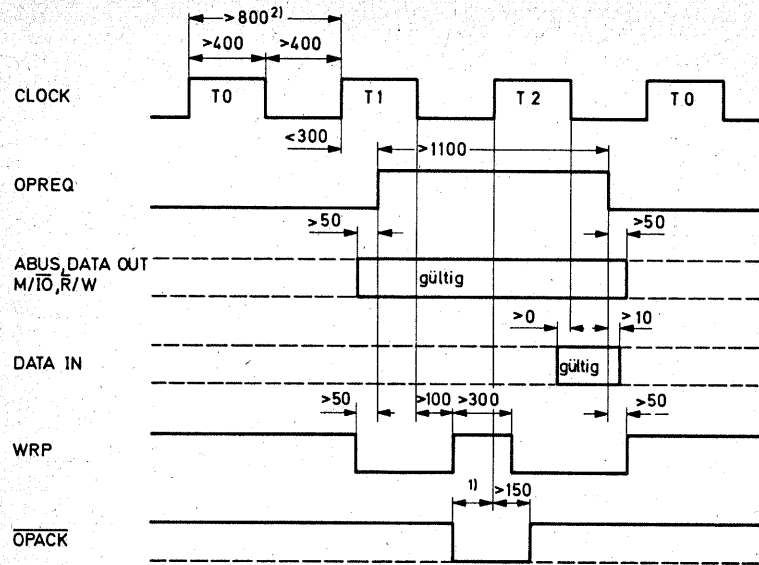
Peripheriegeräte angeschlossen sind, so daß es nicht das Problem gibt, ob ein Drucker, eine Abfüll- oder Werkzeugmaschine dem Prozessor folgen kann, darf $\overline{\text{OPACK}}$ auf Dauer-L geschaltet sein. Dies ist mit dem Schalter S3 in Stellung $\overline{\text{LAUF}}$ möglich (Bild 7.12).

Der Ausgang des angeschlossenen NAND-Gatters liegt auf L, weil die offenen TTL-Eingänge sich so verhalten, als hätten sie H. Sie bilden den Eingang $\overline{\text{STOP}}$. Schaltet eine Peripherieeinheit diesen Eingang auf L, so geht der Ausgang auf H, und das $\overline{\text{OPACK}}$ -Signal wird inaktiv – der Prozessor „wartet“. Über den Eingang $\overline{\text{STOP}}$ kann der Prozessor also mit peripheren Einheiten im Quitungsverfahren („hand-shake“) zusammenarbeiten.

Die Einzelschrittsteuerung

Ein besonders langsamer Partner des Computers dürfen z. B. Sie sein, wenn Sie alle Signale in Ruhe betrachten wollen. Die Einzeltakteingabe ermöglicht Ihnen schon ein beliebig langsames Arbeitstempo. Dabei müssen Sie aber die Taktimpulse, die Sie eingeben, genau mitzählen. Für manche Betrachtungen ist die Einzeltakteingabe sehr vorteilhaft, insgesamt ist das dauernde Abzählen der Taktimpulse doch sehr mühsam. Bequemer haben Sie es, wenn Sie dem Prozessor beibringen, von selbst nach einem Maschinenzyklus anzuhalten. Mit dem $\overline{\text{OPACK}}$ -Signal ist das gar nicht so schwer (Bild 7.10).

Zur Einzelschrittsteuerung schieben Sie den Schalter S3 in Stellung $\overline{\text{SCHRITT}}$ (Bild 7.11). Dadurch erhält der Prozessor sein $\overline{\text{OPACK}}$ -Signal von $\overline{\text{Q}}$ des Flip-Flops FF2.



- 1) >100 ns, kürzere Zeiten lösen Wartetakt aus.
- 2) Flankensteilheit aller Eingangssignale <20 ns (zwischen 0,8V und 2,0V), Messpegel 1,5 V.

Bild 7.10 Impulsiagramm für Lese- und Schreibzyklus des Mikroprozessors 2650 (alle Zeiten in ns; nach VALVO-Unterlagen)

Unten: Bild 7.11 Die Einzelschrittsteuerung der CPU

Mit dem Taster Ta 2 (STEP) können sie entprellte Pulse erzeugen. Im Ruhezustand steht am Ausgang von N5 ein H. Beim Drücken von Ta 2 fällt der Ausgang auf L. C7 differenziert diesen Spannungssprung. Sein Ladestrom erzeugt an R8 einen sehr kurzzeitigen Spannungsabfall, so daß der S-Eingang von FF2 kurzzeitig auf L gezogen wird, die Folge: FF2 wird gesetzt, Q ist also auf L, der Prozessor erhält sein OPACK-Signal und kann arbeiten.

Er arbeitet die Taktzeiten (T0), T1 und T2 ab. So lange ist entweder ADMEM oder ADPER aktiv. Kurz bevor der Prozessor aber wieder bei T0 angekommen ist, ist OPREQ = L, d.h. sowohl ADMEM als auch ADPER

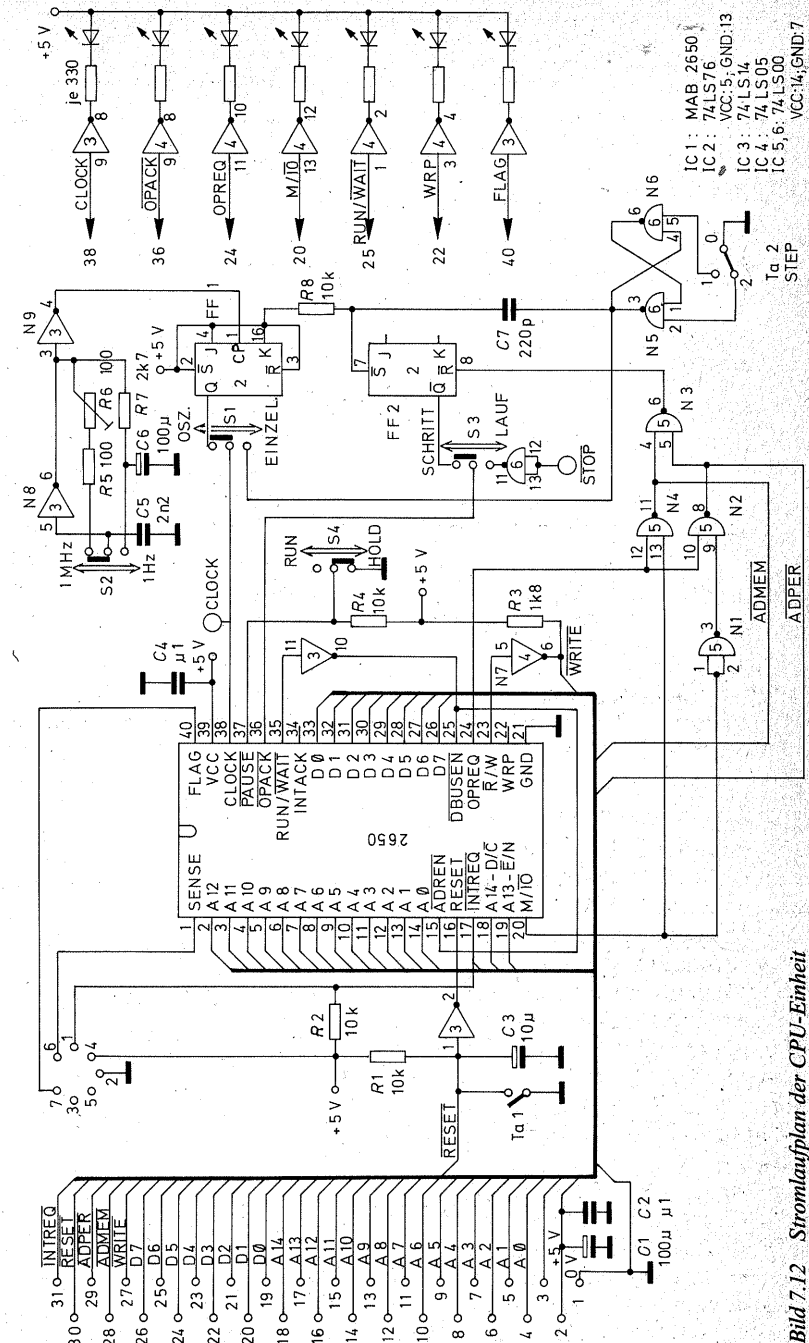
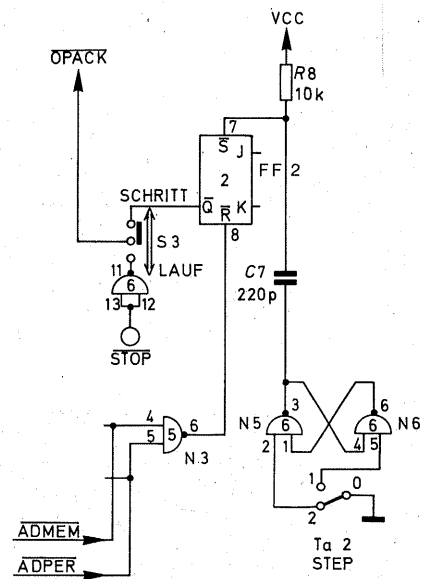


Bild 7.12 Stromlaufplan der CPU-Einheit

sind H (zur Erinnerung: Ein einziges L an einem Eingang zwingt den Ausgang eines NAND-Gatters auf H, siehe Seite 49). Dadurch ist die UND-Bedingung für N3 erfüllt. Dessen Ausgang springt auf L und setzt das FF2 zurück ($Q = H$). Nun gibt es kein OPACK-Signal mehr, und der Prozessor wartet, bis Sie wieder durch einen Druck auf STEP das FF2 setzen und das OPREQ-Signal mit OPACK quittieren.

Während der Wartezeit innerhalb von T1 liegen die Adressen und Daten gültig auf den Bussen und können gelesen werden.

Die Signale ADREN und DBUSEN

Die Anschlüsse ADREN (15) und DBUSEN (25) dienen dazu, die Tri-State-Ausgänge der Busse zu öffnen oder hochohmig zu machen. Für den Normalbetrieb käme man damit aus, beide Eingänge auf L zu schalten. Es ist aber doch sehr nützlich, wenn man den Prozessor elektronisch von den Bussen abtrennen kann. Das ist z. B. notwendig, wenn man den Speicher laden will.

Deshalb sind die beiden Eingänge zusammengefaßt und werden von einem Inverter gesteuert (Bild 7.12). Dieser wiederum erhält sein Eingangssignal vom RUN/WAIT-Ausgang.

Wenn der Prozessor läuft, ist RUN/WAIT = H, also liegen ADREN und DBUSEN auf L und sind freigegeben.

Will man die Busleitungen in den hochohmigen Zustand bringen, um den Speicher zu laden, so läßt man den Prozessor in den WAIT-Zustand laufen (Schalter S4 auf HOLD). Die Leitung RUN/WAIT wird dadurch L, der Ausgang des Inverters schaltet

auf H, und so werden die Busleitungen in den hochohmigen Zustand versetzt.

Das Erreichen des WAIT-Zustandes erkennen Sie daran, daß die LED RUN/WAIT erlischt und daß in den Schalterstellungen DAFLOT (Dateneingabe und -anzeige) und ADFLOT (Adreßeingabe und -anzeige) alle Daten- und Adreß-LEDs aufleuchten.

1	Leiterplatte
1	31polige Stiftleiste, Baureihe GdsW
4	Lötinsel RTM, 1,3 mit Steckhülse
2	Shadow-Digitaster „Mini“, Typ REK, 1 X UM
4	kleiner Schiebeschalter
4	Fassung DIL 14
1	Fassung DIL 16
(1)	Fassung DIL 30
1	Fassung DIL 40
1	7polige Diodenbuchse für Printmontage o.ä. Steckverbinder
7	LED
2	74LS00
1	74LS05
1	74LS14
1	74LS76
1	MAB 2650 A
1	Widerstand 100 Ω, 0,125 W
7	Widerstand 330 Ω, 0,125 W
1	Widerstand 1,8 kΩ, 0,125 W
1	Widerstand 2,2 kΩ, 0,125 W
4	Widerstand 10 kΩ, 0,125 W
1	Trimmerwiderstand 100 Ω „liegend“
1	keramischer Scheibenkondensator 220 pF
1	Folienkondensator 2,2 nF
2	keramischer Scheibenkondensator 0,1 μF
2	Elektrolytkondensator 100 μF/16V

Bild 7.13 Stückliste für die CPU-Einheit

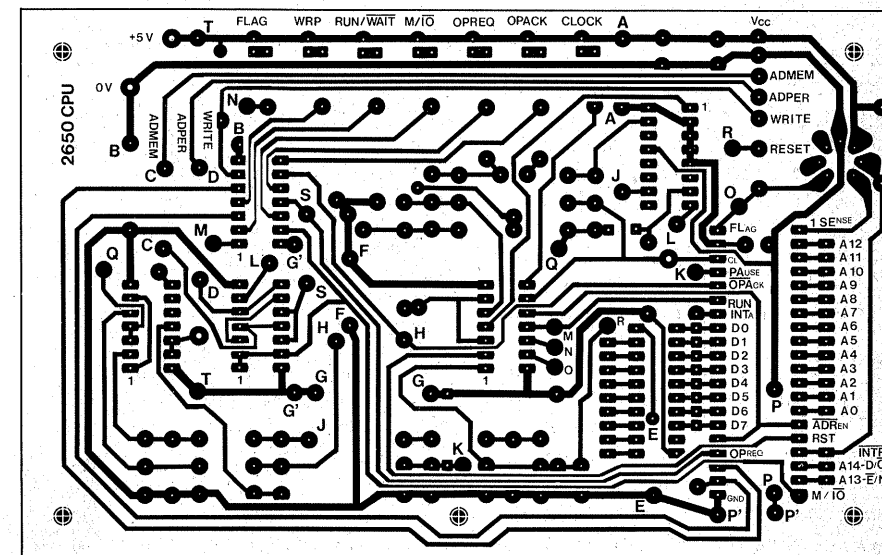
Hinweise zum Aufbau

Die Bilder 7.14, 7.15 und 7.16 zeigen das Leiterbahnbild, die Bestückungsseite und die Drahtbrücken auf der Lötseite.

Beginnen Sie die Bestückung mit den 11 Drahtbrücken auf der Oberseite der Platine (A-A, B-B, C-C, D-D, E-E, F-F, G-G, H-H, P-P, P'-P'). Vergessen Sie vor allem P-P nicht: Die Drahtbrücke liegt unter der Fassung für den Mikroprozessor. E-E liegt unter der Fassung für einen später eventuell zu ergänzenden Bustransceiver. Die Reihenfolge der übrigen Bestückung auf der Oberseite der Platine ist beliebig. Den Widerstand R8 löten Sie nur dann ein, wenn Sie für IC2 wirklich einen LS-Typ (74LS76) verwenden. Sofern Sie einen Standard-

typ einsetzen, weil Sie ihn z. B. zufällig in der Bastelkiste haben, **entfällt R8**. Der Grund: C7 bildet mit R8 (genauer, dem Eingangswiderstand des Eingangs \bar{S} und dem parallelgeschalteten R8) das Differenzglied zum Setzen von FF2. In einem Standard-TTL-Eingang (Bild 2.32) ist der Innenwiderstand mit ca. 4 kΩ so klein (R1 in Bild 2.29 a und 2.32), daß er bei einer Kapazität von 220 pF nicht weiter verringert werden darf. Bei LS-Typen ist R1 mit 16 bis 20 kΩ dagegen sehr viel größer. Die Zeitkonstante mit 220 pF wäre zu lang, und deshalb wird R8 parallelgeschaltet, um den Gesamtwiderstand zu verringern. Falls Sie die Befestigungslaschen der Schalter S1, S2, S3 und S4 kürzen wollen, stellen Sie den Schieber jeweils auf die Seite, deren Lasche Sie absägen werden. Der Schieber verhindert das Schalterinnere und verhindert, daß Sägespäne eindringen. Spannen Sie dann je zwei Schalter mit den Anschlüssen nach oben in ei-

Bild 7.14 Leiterbahnbild für die CPU-Platine



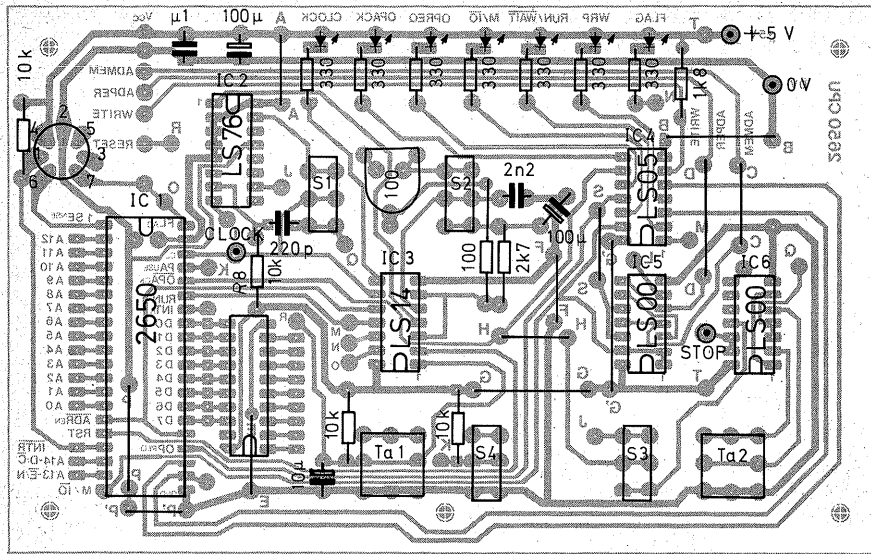


Bild 7.15 Die Bestückungsseite der CPU-Platine

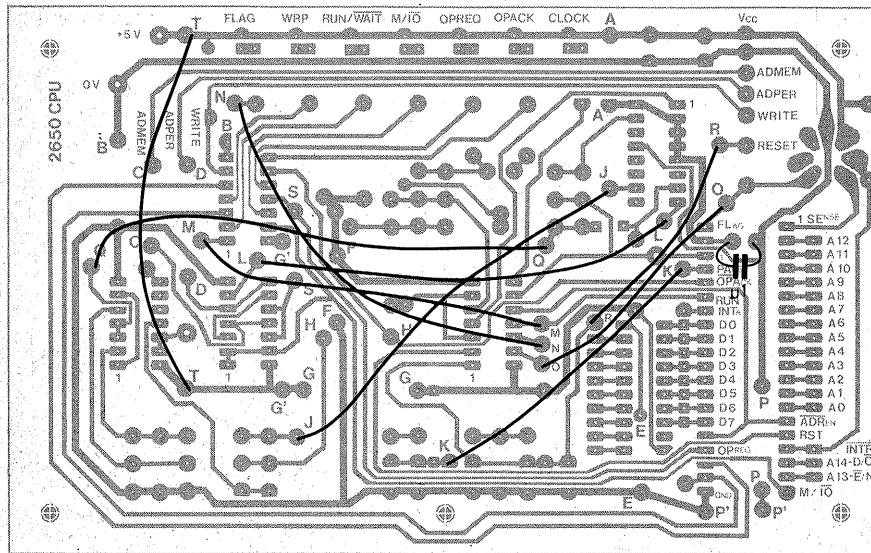


Bild 7.16 Die Drahtbrücken auf der Lötseite der CPU-Platine

nen kleinen Schraubstock (Bild 7.17), und sägen Sie die Laschen mit der Laubsäge (feinstes Metallsägeblatt) ab. Wenn Sie auf der „inneren Breitseite“ der Lasche sägen, dient Ihnen der Schalterkörper als Führung, und Ihnen gelingt ein glatter Schnitt. Sie brauchen dann nur noch die Schnitt-

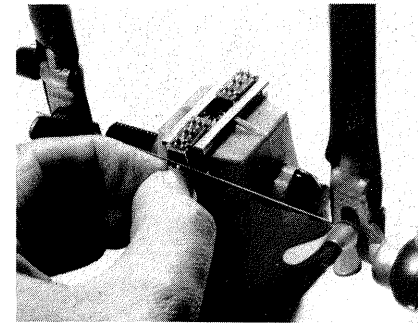
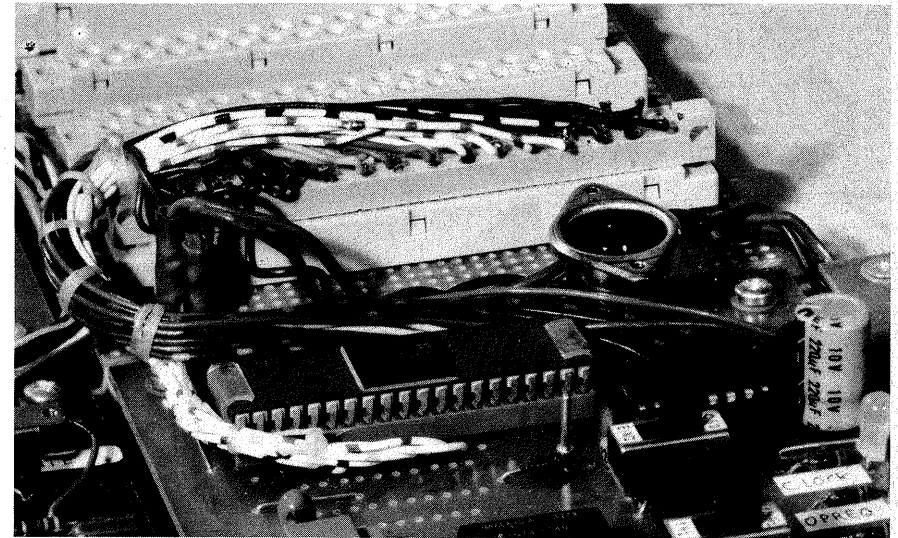


Bild 7.17 Absägen der Befestigungslaschen von Schiebeschaltern

Bild 7.18 Anschluß der CPU-Einheit an die Busplatte



kanten mit einer Feinschlichtfeile zu entgraten.

Die vorgesehene 7polige Diodenbuchse ist vielleicht nicht überall zu haben. Sie dient in erster Linie zum Anschluß des Kassetteninterfaces. Sie können sie durch jede andere Steckverbindung ersetzen, sofern sie mindestens fünf Pole hat und Sie sie auf oder in der Nähe der Platine montieren können.

Auf der Platine ist neben den Anschlüssen D0 bis D7 eine 20polige DIL-Fassung für einen Bustransceiver (74LS245) vorgesehen. Der kann notwendig werden, wenn Sie Ihren Computer zu einem größeren System ausbauen. In der Einfachstversion mit der schrittweisen Dateneingabe bzw. -anzeige wäre er nur mit einer zusätzlichen Steuerlogik zu gebrauchen, kurz, am Anfang stört er. Daher lassen Sie ihn weg – er fehlt ja auch im Stromlaufplan (Bild 7.12). Sie löten daher die Datenleitungen unmittelbar an die Prozessoranschlüsse. Be-

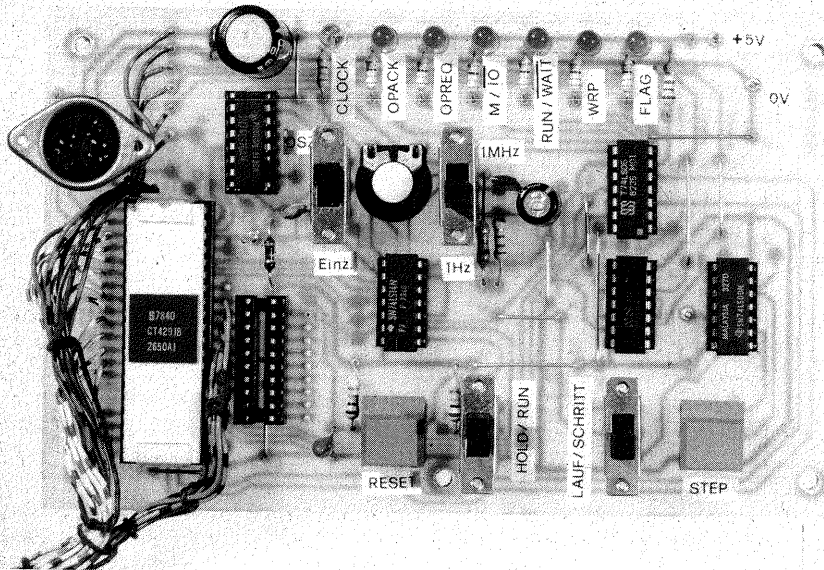


Bild 7.19 Die fertige CPU-Einheit

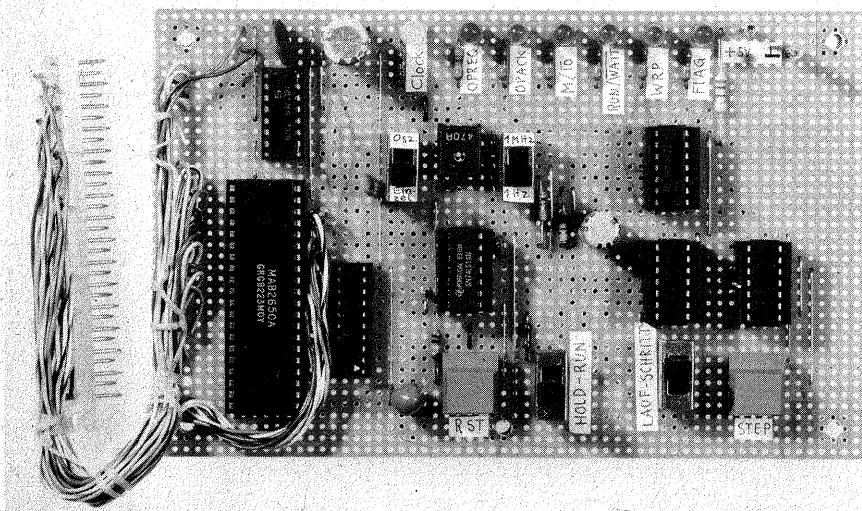


Bild 7.20 Die fertige CPU-Einheit auf einer Experimentierplatte

achten Sie, daß die Numerierung am Prozessorsockel gegenläufig zu der Reihenfolge auf der Busplatte verläuft!

Die Anschlußdrähte für die Stiftleiste löten Sie zuerst an die Platine und verbinden sie dann mit der Stiftleiste. Die Drähte sollten Sie in einem kurzen Bogen an die Stiftleiste führen, für die Platz 7 in Bild 5.5 vorgesehen ist (siehe auch Bild 7.18).

Bild 7.19 zeigt die fertige CPU-Platte.

Hinweise zum Aufbau auf einer Experimentierplatte

Bild 7.20 zeigt den Aufbau der CPU auf einer Experimentierplatte. Die Anordnung der Bauelemente unterscheidet sich nicht wesentlich von der auf der Platine. Lediglich die Anschlußbuchse für das Kassetteninterface fehlt. Diese Buchse sollten Sie außerhalb der Experimentierplatte montieren, damit diese nicht übermäßig durch Druck oder Zug belastet wird.

Beim Aufbau beginnen Sie am besten mit den Tastern, Schaltern und IC-Fassungen. Dann stellen Sie die Verbindungen 0 V und VCC für alle Fassungen her und messen nach, ob die Versorgungsspannung richtig anliegt. Beim Aufbau sollten Sie unbedingt stufenweise vorgehen und jede Stufe gleich nach ihrer Fertigstellung kontrollieren. Die nachstehende Prüfanleitung hilft Ihnen dabei. Als Stufenfolge empfiehlt sich:

- Anzeigeeinheiten (LEDs und IC3, 4); alle LEDs müssen leuchten, weil die Invertereingänge ja noch offen sind. Verbinden Sie die Eingänge kurz mit 0 V, dann müssen die LEDs verlöschen.
- Taster-Flip-Flop (Ta 2, IC6);
- die beiden Flip-Flops von IC2;
- der CLOCK-Oszillator (IC3);

- die Einzelschrittsteuerung (IC5, IC2);
- alle restlichen Verbindungen mit der Fassung von IC1. Der Widerstand R2 (H für INTREQ) befindet sich auf der Lötseite der Experimentierplatte.

Prüfung der CPU

Stecken Sie die ICs mit Ausnahme des Mikroprozessors in die Fassungen, und verbinden Sie die CPU mit der Busplatte.

Prüfen der Steuersignale

Datenschalter Da auf DAFLOT

S1 auf OSZ

S2 auf 1 Hz

S3 auf SCHRITT

S4 beliebig

Folgende Anzeigen müssen sich einstellen:

Auf der CPU-Platte

CLOCK blinkt hoffentlich (siehe Seite 127)

OPREQ H

M/IO H

RUN/WAIT H

WRP H

FLAG H

OPACK H, wenn nicht, dann Pin 8 von IC2 (74LS76) kurz mit 0 V verbinden, dann muß OPACK auf H umschalten.

Auf der Dateneingabe und -anzeige

D0 bis D7: H

ADMEM L

ADPER H

WRITE L

S3 auf LAUF; $\overline{\text{OPACK}}$ muß auf L gehen.

S3 wieder auf SCHRITT zurückstellen.

Pin 23 ($\overline{\text{R/W}}$) von IC1 (= Pin 5 von IC4) mit 0 V verbinden (Prüfkabel + Stecknadel) – $\overline{\text{WRITE}}$ muß H anzeigen;

Pin 20 ($\overline{\text{M/IO}}$) von IC1 mit 0 V verbinden – $\overline{\text{ADMEM}}$ muß H anzeigen, dafür geht $\overline{\text{ADPER}}$ auf L.

Pin 23 erneut mit 0 V verbinden und Datenschalte Da auf $\overline{\text{WRITE}}$ stellen – LED $\overline{\text{WRITE}}$ muß von H auf L wechseln.

Prüfen des Tastenentprell-Flip-Flops (N5, N6)

Spannung an Pin 3 (oder 4) von IC6 messen.

In Ruhestellung der Taste STEP muß H vorhanden sein, beim Drücken von STEP muß sich L einstellen (Gegenprüfung durch Messung an Pin 1 bzw. Pin 6: Sprung von L auf H).

Prüfen des Flip-Flops FF2 für die SCHRITT-Schaltung

S3 auf SCHRITT stellen – nach den STEP-Kontrollen muß die LED $\overline{\text{OPACK}}$ L anzeigen. Pin 8 von IC2 (bzw. Pin 6 von IC5) kurzzeitig mit 0 V verbinden – die LED $\overline{\text{OPACK}}$ muß danach H anzeigen;

STEP drücken – $\overline{\text{OPACK}}$ muß auf L wechseln;

S1 auf EINZEL stellen – die LED $\overline{\text{CLOCK}}$ muß in Ruhestellung H anzeigen.

Mit dem Betätigen von STEP muß $\overline{\text{CLOCK}}$ von H auf L springen.

Prüfen des Oszillators

S1 auf OSZ zurückstellen, S2 auf 1 Hz.

Die LED $\overline{\text{CLOCK}}$ muß blinken – wenn nicht, kann es daran liegen, daß R7 zu groß ist, wenn Sie z. B., um auf längere Taktzeiten zu kommen, R7 mit einem Wert $> 2,2 \text{ k}\Omega$ eingesetzt haben.

Falls Sie ein Standard-IC (7414 statt 74LS14) verwenden, muß der Widerstand kleiner als $1 \text{ k}\Omega$ sein.

Wenn die LED $\overline{\text{CLOCK}}$ nach eigenen Wünschen zu schnell blinkt, können Sie C6 vergrößern, indem Sie auf der Kupferseite der Leiterplatte einen zweiten Elko dazulöten.

Ableich der $\overline{\text{CLOCK}}$ -Frequenz auf 1 MHz:

S2 auf 1 MHz – die LED $\overline{\text{CLOCK}}$ muß dunkel leuchten.

Der Verlust an Helligkeit kommt dadurch zustande, daß die LED abwechselnd aufleuchtet und erlischt, aber so schnell, daß das Auge dem schnellen Wechsel nicht zu folgen vermag.

Wenn Sie einen Frequenzzähler haben, können Sie die $\overline{\text{CLOCK}}$ -Frequenz unmittelbar an dem Lötnagel neben Pin 38 von IC1 messen. Wenn nicht, klemmen Sie ein kurzes Prüfkabel als Antenne an diesen Lötnagel. Stellen Sie ein Mittelwellenradio auf die Frequenz 1 MHz und bringen es in unmittelbare Nähe der kleinen Sendeantenne. Verstellen Sie R6 langsam, bis Sie das $\overline{\text{CLOCK}}$ -Signal im Radio wie einen unmodulierten Sender empfangen. Abends werden Sie Ihren „Sender“ durch das Interferenzpfeifen am leichtesten finden.

Prüfen von $\overline{\text{INTREQ}}$ und RESET

An Pin 17 ($\overline{\text{INTREQ}}$) von IC1 müssen Sie H messen können.

An Pin 16 (RESET) muß sich L eingestellt haben; beim Betätigen von Ta 1 (RST) muß sich H einstellen.

Prüfen der Datenleitungen

Datenschalter Da auf $\overline{\text{WRITE}}$ stellen. D0 bis D7 auf L (zum vorderen Rand der Leiterplatte) stellen.

Spannung an Pin 33 (D0) von IC1 messen, dabei den Schalter D0 von L auf H und zurück schieben – der Wechsel L – H – L muß an Pin 33 zu messen sein.

Ebenso D1 bis D7 (Pin 32 bis 26) überprüfen.

Prüfung der Adreßleitungen

Prüfen Sie mit dem Ohmmeter, ob zwischen der Buchse 4 (A0) einer Federleiste auf der Busplatte und dem Pin 14 (A0) der Fassung von IC1 ein Durchgang besteht.

In gleicher Weise prüfen Sie auch die übrigen Adreßleitungen.

Funktionsprüfung mit dem Prozessor

VCC abschalten und IC1 einsetzen. Dabei sollten Sie die Anschlußstifte möglichst nicht berühren.

VCC wieder einschalten und folgendes „Programm“ eingeben. Dies ist zugleich auch die Bedienungsfolge in der Einfachstversion:

Datenschalter Da auf $\overline{\text{WRITE}}$

S1 auf OSZ

S2 auf 1 Hz

S3 auf SCHRITT

S4 auf RUN

Drei $\overline{\text{CLOCK}}$ -Pulse lang RST drücken, dann warten, bis $\overline{\text{OPREQ}}$ wieder H ist.

D0 bis D7 auf 04₁₆ (LODI, R0), dann STEP.

Beobachtung:

$\overline{\text{OPACK}}$: L – H – H

$\overline{\text{OPREQ}}$: H – L – H (siehe auch Bild 7.10).

D0 bis D7 auf beliebige Zahl, z. B. 07₁₆, dann STEP.

Beobachtung: wie oben.

Damit haben Sie die Zahl 07₁₆ in das Arbeitsregister R0 geladen.

Nun prüfen Sie nach, ob die eingegebene Zahl auch wirklich in R0 steht: D0 bis D7 auf F0₁₆ (WRTC) stellen, dann STEP.

Beobachtung:

$\overline{\text{OPACK}}$: L – H – H

$\overline{\text{OPREQ}}$: H – L – H

$\overline{\text{M/IO}}$: H – L – L

WRP: H – L – L – H

Die Daten-LEDs verlöschen teilweise, je nach Inhalt des Registers R0.

Datenschalter Da auf DAFLOT umschalten. Der Inhalt von Register R0 erscheint auf den Datenanzeigen.

Wenn das funktioniert, haben Sie allen Grund, sich zu freuen, denn es ist Ihnen gelungen, einen arbeitsfähigen Computer – wenn auch in der einfachsten möglichen Version – zu bauen.

Erste Programmierübungen mit dem Computer

Mit diesem Einfachstcomputer können Sie bereits arbeiten und die Wirkung vieler Befehle erkunden. Auf dieser Fertigungsstufe können Sie auch das Programm-Status-Wort beobachten und die Veränderung der Bits durch bestimmte Operationen kennenlernen. Eine ausführliche Befehlsliste finden Sie im Anhang 2. Probieren Sie möglichst viele Befehle aus, wobei Sie die Sprungbefehle noch außer acht lassen können, denn die bekommen ihre Bedeutung erst mit dem Vorhandensein eines Speichers. Aber die Datentransportbefehle sowie die arithmetischen und logischen Befehle können Sie sich schon vornehmen, dazu gehören auch die Rotierbefehle.

Arbeiten Sie ruhig lange mit dieser Aufbaustufe, es lohnt sich. Sie lernen dabei die ersten Befehle auswendig. Sie bekommen einen sicheren Überblick über die Steuersignale, insgesamt lernen Sie viel von der Arbeitsweise eines Mikrocomputers kennen.

Kapitel 8

Die Adreßeingabe und -anzeige (Baugruppe 5)

Die Adreßeingabe und -anzeige hat zwei Aufgaben:

1. Sie zeigt den augenblicklichen Stand des Programmzählers an (siehe Seite 118).

2. Sie adressiert die Speicherplätze, in die das Benutzerprogramm eingegeben wird.

Da sie nahezu vollständig der Dateneingabe und -anzeigeeinheit gleicht, erübrigt sich eine technische Beschreibung. Sowohl die Anzeige der Nullen und Einsen als auch deren Erzeugung sind identisch (Bilder 6.4 bis 6.6).

Die Baugruppe ist für 12 Dualstellen ausgelegt (A0 bis A11). Damit lassen sich 4096 Speicherplätze adressieren bzw. anzeigen – das ist fürs erste, solange man noch Daten mit den Schiebeschaltern eingibt, mehr als genug. Der UM-Schalter Adr mit den Stellungen ADFLOT und ADVAL hat die dem Datenschalter Da entsprechende Funktion:

In der Stellung ADFLOT (**adreses float**, engl. die Adreßleitungen schweben) zeigt die Einheit die Adressen an, die der Mikroprozessor mit seinen Adreßausgängen abgibt. Die Stellung ADFLOT ist die normale Betriebs-

stellung, denn die Adreßleitungen müssen frei schweben können, wenn der Prozessor die Informationen aus dem Speicher liest oder in ihn einschreibt.

Die Stellung ADVAL (**adreses valid**, engl. die mit den Schiebeschaltern eingestellten Adressen gelten) **wird nur in einem einzigen Fall benutzt, wenn nämlich der Speicher mit einem Programm geladen wird, sonst nie!**

Außerdem enthält die Baugruppe noch den Taster M (**Memory**, engl. Speicher), der es ermöglicht, die Speicherleitung ADMEM **kurzzeitig** auf L zu schalten und damit den Speicher zu aktivieren. Dabei wird der Ausgang von N5 auf der CPU-Einheit ebenfalls für die Dauer des Tastendrucks kurzgeschlossen. Die Methode ist zugegebenermaßen etwas rau, doch für die **kurze** Zeit übersteht das Gatter aber diese grobe Behandlung, und deshalb kann der Schaltungsaufbau vereinfacht werden. Sonst hätte IC5 auf der CPU-Einheit ein o.c.-IC sein müssen (z. B. 7401), mit je einem Pull-Up-Widerstand an jedem Gatterausgang. Dann hätte sich ein Wired-NOR ergeben wie bei der WRITE-Leitung (N7 auf der CPU-

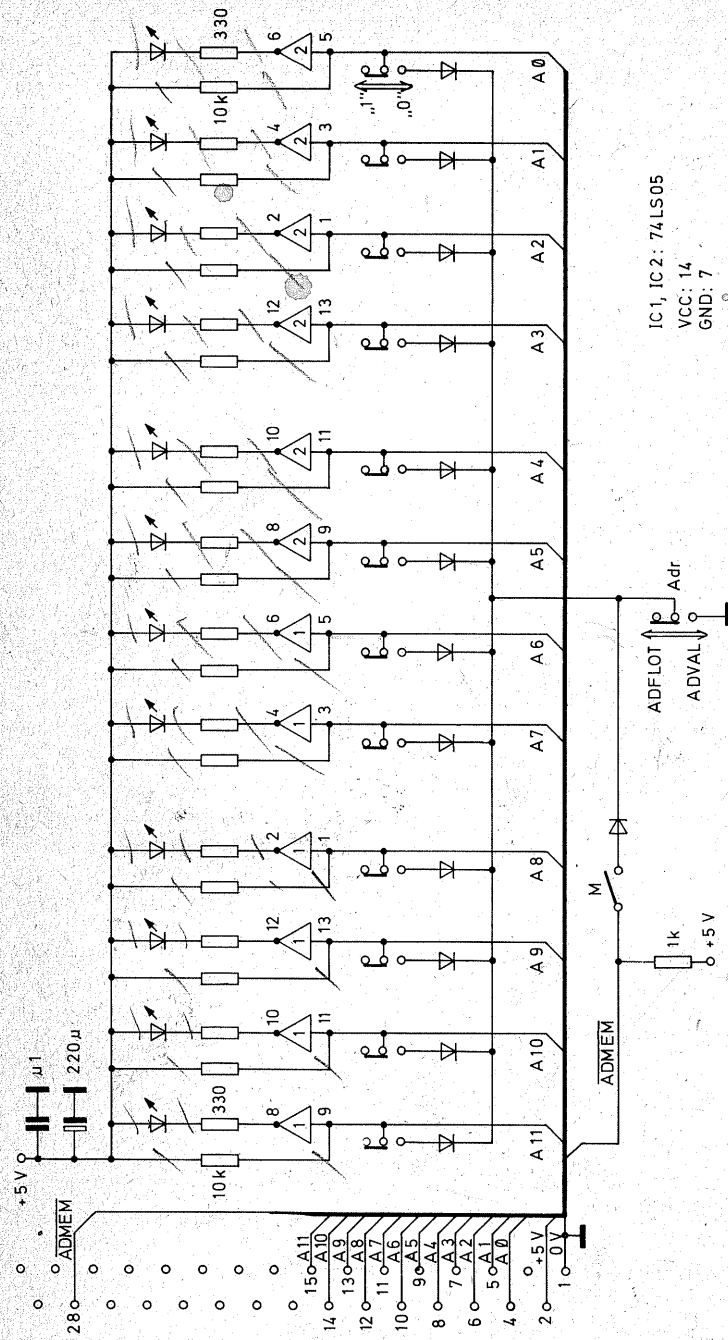


Bild 8.1 Stromlaufplan der Adreßeingabe und -anzeige

Einheit). Der Unterschied besteht darin, daß die WRITE-Leitung für längere Dauer auf L geschaltet wird, die Leitung ADMEM durch den Taster M aber nur für kurze Pulse. Aufbau und Prüfung der Baugruppe unterscheiden sich nicht von der Dateneingabe und -anzeige. Bild 8.1 zeigt den Stromlaufplan, 8.3 das Leiterbahnbild, 8.4 die Bestückungsseite, 8.5 die Drahtbrücken auf der Lötseite, 8.6 die fertige Baugruppe und 8.7 schließlich den Aufbau auf einer Experimentierplatte. Montieren Sie die fertige Adreßeinheit links neben der Dateneingabe und -anzeige auf die Grundplatte (unten links in Bild 1.18). Zum Anschluß der Stiftleiste ist der Steckplatz 1 in Bild 5.5 vorgesehen. Schließen Sie die Adreßeinheit an, wobei Sie den Schalter Adr in die Stellung ADFLOT bringen. Die übrigen Adreßschalter können beliebig stehen. Je nach Stand des Mikroprozessors werden irgendwelche Anzeigen

- | | |
|----|--|
| 1 | Leiterplatte |
| 1 | 31polige Stiftleiste, Baureihe GdsW |
| 2 | Löttafel RTM 1,3 mit Steckhülsen |
| 13 | kleiner Schiebeschalter |
| 1 | Shadow-Digitaster „Mini“ Typ REK |
| 2 | Fassung DIL 14 |
| 2 | 74LS05 |
| 12 | LED |
| 13 | Silizium-Allzweckdiode, z. B. 1 N 4148 o. ä. |
| 12 | Widerstand 330 Ω, 0,125 W |
| 1 | Widerstand 1 kΩ, 0,125 W |
| 12 | Widerstand 10 kΩ, 0,125 W |
| 1 | Elektrolytkondensator 100 bis 220 µF/16 V |
| 1 | keramischer Scheibenkondensator 0,1 µF |

Bild 8.2 Stückliste für die Adreßeingabe und -anzeige

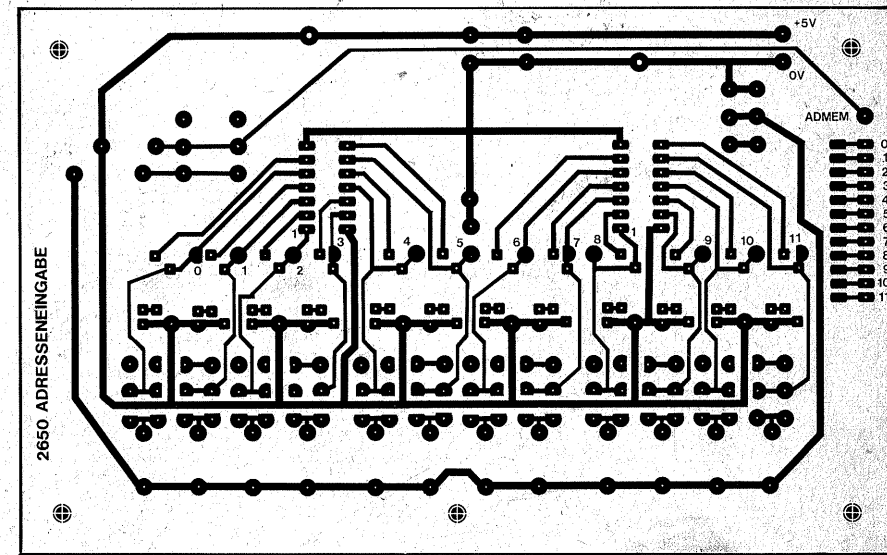


Bild 8.3 Leiterbahnbild für die Adreßeingabe und -anzeige

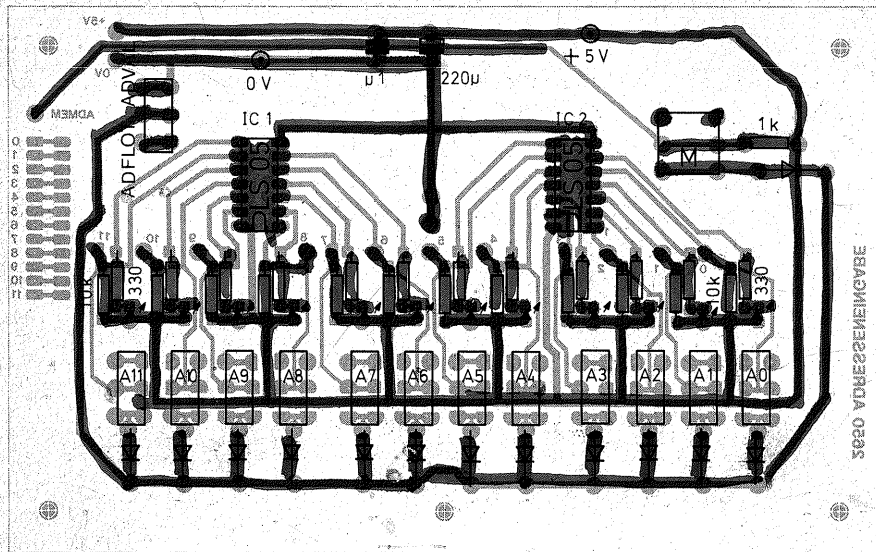


Bild 8.4 Die Bestückung der Adreßeingabe und -anzeige

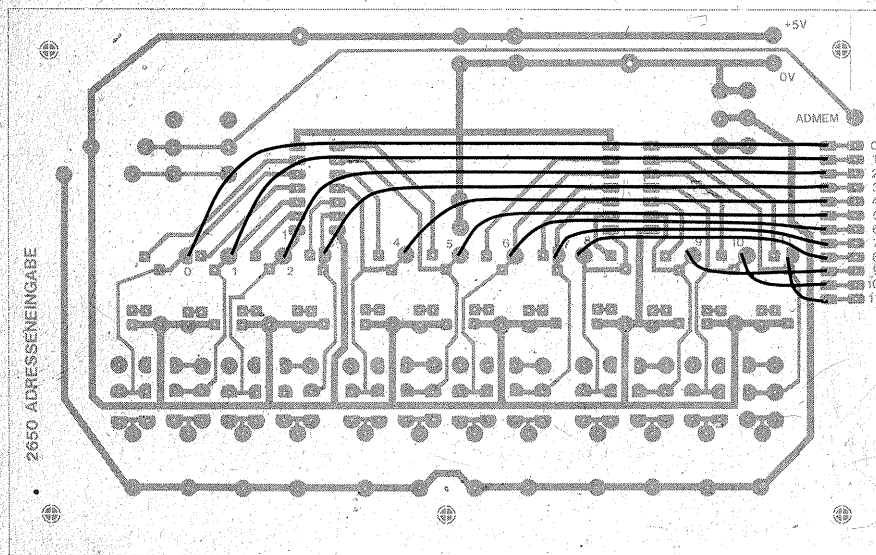
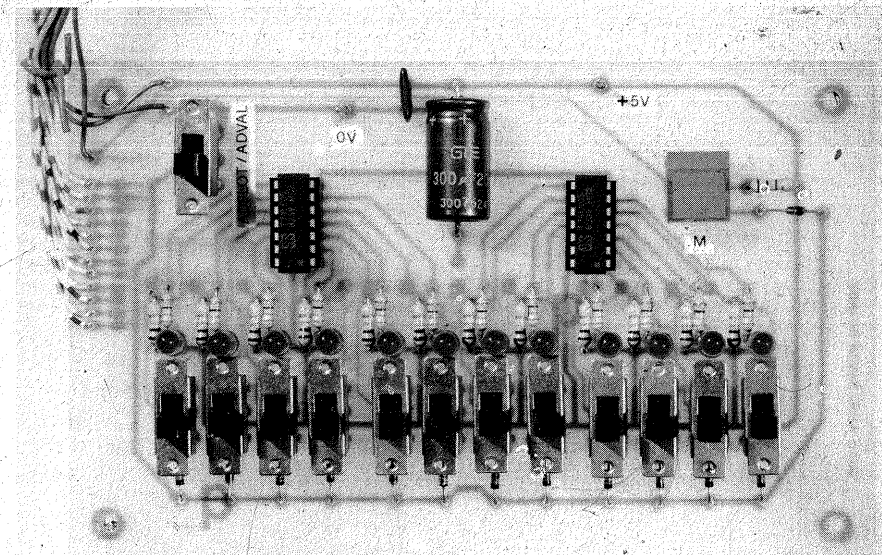
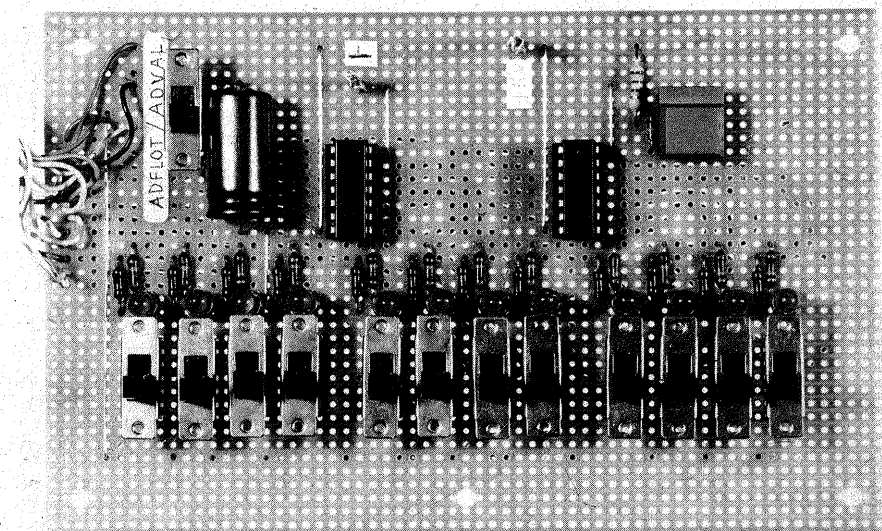


Bild 8.5 Die Drahtbrücke auf der Lötseite



Oben:
Bild 8.6 Die fertige Adreßeingabe und -anzeige

Unten:
Bild 8.7 Der Aufbau der Adreßeingabe und -anzeige auf einer Experimentierplatte



aufluchten. Wenn Sie den Prozessor durch Betätigen der RESET-Taste (Ta 2) zurücksetzen, müssen alle LEDs verlöschen, denn der Prozessor zeigt auf die Anfangsadresse 0000₁₆.

Eine Adresse besteht wegen der insgesamt 15 Adreßleitungen (A0 bis A14) immer aus 4 Nibbles = 2 Bytes, z. B. 01 A5. **Beim 2650 werden Adressen so eingegeben, wie man sie gewohnterweise von links nach rechts schreibt. Zuerst** gibt man das höherwertige Byte ein (im obigen Beispiel 01), **danach** das niedrigerwertige (im obigen Beispiel A5). Diese scheinbar logische Reihenfolge ist durchaus nicht selbstverständlich. Bei vielen anderen Prozessoren muß man zuerst das niedrigerwertige Byte eingeben, dann das höherwertige!

Arbeiten Sie nun kleinere Programme wie früher ab. Sie werden dabei beobachten, wie der Prozessor mit jedem Byte, das er verarbeitet, eine Adresse weiter hochzählt – im Dualsystem, versteht sich. Sie können schon daran erkennen, wie Speicherplätze geordnet werden: Immer schön nach der Reihe der Dualzahlen, von 0 beginnend.

Sie können jetzt auch schon Sprungbefehle eingeben und beobachten, wie der Prozessor auf die Zieladresse zeigt. Geben Sie z. B. „1F – STEP“ (BCTA) ein, anschließend „00 – STEP“ für das obere Byte der Adresse, dann „16₁₆ – STEP“ für das untere Byte, so werden Sie sehen, daß der Prozessor auf die Adresse 00 16₁₆ zeigt.

Wenn Sie nun ein kleines Addierprogramm anfügen, z. B.

75	08	CPSL	(Lösche C-Bit im PSL)
04	07	LODI, R0, 07	(Lade 07 ₁₆ in R0)
84	11	ADDI, R0, 11	(Addiere zu [R0] 11 ₁₆)
F0		WRTC, R0	(Schreibe [R0] auf den Datenbus),

werden Sie beobachten können, daß der Prozessor von der angesprungenen Adresse aus weiterzählt.

Jetzt ist es auch an der Zeit zu überprüfen, ob das Differenzierglied R8/C7 auf der CPU-Einheit richtig arbeitet.

Ein „ordentlicher“ Elektroniker hätte statt dessen ein Mono-Flop mit einer Pulsdauer von 0,2 µs eingesetzt. Das hätte aber ein IC nebst Fassung mehr gekostet, außerdem noch Platz auf der Leiterplatte.

Probleme werden Sie mit dem Differenzierglied wahrscheinlich nicht haben, trotzdem sollten Sie es überprüfen, weil an dieser Stelle Bauteiltoleranzen doch wirksam werden könnten.

Geben Sie auf der Datenplatte den Befehl C0 (NOP, no operation) ein. Schalten Sie S2 auf 1 MHz, S3 auf SCHRITT, und tasten Sie den Taster STEP. Mit jedem Tastendruck läuft, wie Ihnen inzwischen wohlvertraut ist, je ein Maschinenzyklus ab. Der Prozessor führt keine Arbeit aus, mit Ausnahme der, daß der Programmzähler einen Schritt weiterrückt. Dieser Befehl ist für den Programmierer bei der Entwicklung von Programmen sehr hilfreich. Ist der Programmierer an einer Stelle unsicher, ob er vielleicht einen oder mehrere Schritte ergänzen muß, so fügt er an der Stelle eine Reihe von NOP-Befehlen ein und hält sich dadurch Speicherstellen frei, die er später gegebenenfalls mit echten Befehlen füllen kann. Oder wenn er Befehle aus dem Programm herausnehmen muß, fügt er statt ihrer

NOP-Befehle ein, dann braucht er die Adressierung aller folgenden Befehle nicht zu ändern.

Sie nehmen für diesen Test den NOP-Befehl, weil Sie nur das Weiterrücken des Programmzählers beobachten werden und sich daher die Mühe, ein bestimmtes Programm einzugeben, ersparen können.

Beobachten Sie jetzt, ob die Einzelschrittsteuerung auch für den schnellen Betrieb mit der CLOCK-Frequenz von 1 MHz richtig bemessen ist. Mit jedem Tastendruck müssen Sie auf der Adreßeinheit beobachten können, daß der Programmzähler um genau einen Schritt weiterrückt.

– Ist die Funktion unsicher und rückt er nicht bei **jedem** Tastendruck weiter, so hilft es, wenn Sie R8 entfernen. Dieser Effekt konnte bei den vielen nachgebauten Exemplaren nur beobachtet werden, wenn für IC2 und/oder IC6 Standardtypen statt LS-Typen verwendet wurden. Zählt der Programmzähler jeweils um **zwei** Schritte weiter, ist die Zeitkonstante des Differenziergliedes zu lang, d. h. daß Sie eventuell einen LS-Typ an der Stelle von IC2 und/oder IC6 eingesetzt haben, nicht aber R8. Sie können R8 gegebenenfalls auch bis auf 4,7 kΩ verringern.

Der Fall kann auch auftreten, wenn C7 deutlich größer ist als 220 pF. Überprüfen Sie daher u. U. auch die Kapazität von C7.

Kapitel 9

Der Speicher (Baugruppe 6)

Bisher haben Sie alle Befehle und Daten einzeln Byte für Byte eingegeben. Der Prozessor hat die Programme noch nicht selbständig abgearbeitet, sondern das Programm hatten Sie entweder im Kopf oder auf Ihrem Notizblock. Auf letzterem standen dann wohl alle Bytes schön der Reihe nach notiert, und Sie haben sie der Reihe nach eingegeben. Sie mußten sie dem Prozessor einzeln eingeben, denn Ihre Notizen waren für ihn unzugänglich – er kann ja Ihre Handschrift nicht lesen.

Wenn es Ihnen aber gelingt, Ihren Notizblock so zu gestalten, daß der Prozessor die Bytes lesen kann, wenn Sie ihm Ihren Notizblock als eine Einrichtung anbieten, welche die H- bzw. L-Pegel der Bytes liefert, dann kann sich der Prozessor Ihr Programm allein abholen und es selbständig abarbeiten – außerordentlich flink sogar, und Sie können sich darauf verlassen, daß er dabei keine Fehler macht. Dieser elektronische Notizblock ist der Speicher.

Die Organisation der Speicheradressen

Es wurde bereits wiederholt darauf hingewiesen, daß ein Speicher aus einer Kette von der Reihe nach nummerierten Plätzen besteht, von denen jeder ein Datenwort speichern kann. Bei einem Prozessor mit der Datenbusbreite von 8 Bit ist das 1 Byte. Das ist bei vielen gebräuchlichen Mikroprozessoren der Fall. Hat der Mikroprozessor eine andere Datenbusbreite, etwa 16 oder gar 32 Bit, dann muß unter einer Adresse eben auch ein 16-Bit- oder ein 32-Bit-Wort zu speichern sein.

Die „Maßeinheit“ für Speicheradressen ist das Kilo(byte), abgek. K oder Kbyte. $1\text{ K} = 2^{10} = 1024$, nicht zu verwechseln mit k (Kleinbuchstabe) = mal 1000 (z. B. in kg, km usw.). Da es doch etwas ungewohnt ist, sich in solchen Größenordnungen Mengen (Zahlen) im Dual- oder Hexadezimalsystem vorzustellen, gibt die folgende Tabelle (9.1) eine Art „Meßlatte“ zur Orientierung über den Speicherbereich bis 32 Kbyte in hexadezimaler Schreibweise. Das ist eben der Bereich, den der 2650 unmittelbar adressieren kann.

Page 0	Page 2
1. K 0000 – 03FF	17. K 4000 – 43FF
2. K 0400 – 07FF	18. K 4400 – 47FF
3. K 0800 – 0BFF	19. K 4800 – 4BFF
4. K 0C00 – 0FFF	20. K 4C00 – 4FFF
5. K 1000 – 13FF	21. K 5000 – 53FF
6. K 1400 – 17FF	22. K 5400 – 57FF
7. K 1800 – 1BFF	23. K 5800 – 5BFF
8. K 1C00 – 1FFF	24. K 5C00 – 5FFF

Page 1	Page 3
9. K 2000 – 23FF	25. K 6000 – 63FF
10. K 2400 – 27FF	26. K 6400 – 67FF
11. K 2800 – 2BFF	27. K 6800 – 6BFF
12. K 2C00 – 2FFF	28. K 6C00 – 6FFF
13. K 3000 – 33FF	29. K 7000 – 73FF
14. K 3400 – 37FF	30. K 7400 – 77FF
15. K 3800 – 3BFF	31. K 7800 – 7BFF
16. K 3C00 – 3FFF	32. K 7C00 – 7FFF

Bild 9.1 Übersicht über den Speicherbereich im 1-K-Abstand

1.K	000 00 00 0000 0000 – 000 00	11 1111 1111
2.K	000 01 00 0000 0000 – 000 01	11 1111 1111
3.K	000 10 00 0000 0000 – 000 10	11 1111 1111
4.K	000 11 00 0000 0000 – 000 11	11 1111 1111

a)

1–2.K	000 0 000 0000 0000 – 000 0	111 1111 1111
3–4.K	000 1 000 0000 0000 – 000 1	111 1111 1111
5–6.K	001 0 000 0000 0000 – 001 0	111 1111 1111
7–8.K	001 1 000 0000 0000 – 001 1	111 1111 1111

b)

Bild 9.2 Übersicht a) über die letzten 10 Stellen im 1-K-Abstand; b) über die letzten 11 Stellen im 2-K-Abstand; die jeweils vorangehenden Stellen zählen dual vorwärts

Wenn Sie diese Tabelle im Dualsystem betrachten, werden Sie bemerken, daß sich die letzten Stellen zyklisch wiederholen, gleichgültig, ob Sie die Stellen 1-Kbyte-weise oder 2-Kbyte-weise betrachten. Es ändern sich **nur** die vorangehenden Stellen. Sie zählen (dual) der Reihe nach weiter. Tabelle 9.2a zeigt das für den 1-Kbyte-Abstand, Tabelle 9.2b für den 2-Kbyte-Abstand.

Diese Regelmäßigkeit wird benutzt, wenn man den Speicherbereich mit Speicherbausteinen füllt. Selbst wenn man einen Speicherbaustein zur Hand hätte, der den gesamten Speicherbereich ausfüllte, wäre er doch nur von begrenztem Wert, weil man den Speicherbereich in der Regel ja mit verschiedenartigen Bausteinen ausfüllen will (siehe Seite 150, 154), die je nach Bedarf unterschiedlich zusammengeschaltet werden.

Sofern die Organisation des Speicherbereichs für die Programmierung wichtig ist, gilt jeweils die Organisation des benutzten Mikroprozessors. Die Einteilungseinheit ist die **Page** (engl. Seite eines Buchs oder Notizblocks). Wie viele Adressen eine Page umfaßt, ist von Prozessor zu Prozessor verschieden.

Beim 2650 ist der adressierbare Speicherbereich von 32 Kbyte in vier Pages unterteilt. Bei den meisten Befehlen sind 13 Bits für die Festlegung der entsprechenden Speicherplätze vorgesehen. Damit kann man die 8-Kbyte jeder einzelnen Page adressieren.

Die 13 niedrigerwertigen Bits A0 bis A12 geben die Adresse der jeweiligen Page an. Um den gesamten 32-K-Speicherraum adressieren zu können, bedarf es zweier weiterer Bits, der sogenannten Page-Bits. Die Bits der Adreßleitungen A13 und A14 bestimmen zusammen die gewählte Page:

	A14	A13	Bereich dekadisch	Bereich hexadezimal
Page 0	0	0	0- 8191	0000-1FFF
Page 1	0	1	8192-16383	2000-3FFF
Page 2	1	0	16384-24575	4000-5FFF
Page 3	1	1	24576-32767	6000-7FFF

Es gibt keinen eigenen Befehl, mit dessen Hilfe die Page-Bits gesetzt werden. Sie werden jedoch durch Sprung- oder durch Unterprogramm-Sprungbefehle gesetzt, weil bei diesen alle 15 Bits für die Adressierung verwendet werden.

Die beiden Page-Bits werden in einem besonderen 2-Bit-Register gespeichert (Bild 7.2, innerhalb der Ein-/Ausgabeverwaltung). Sie liegen an den Adreßleitungen 13 und 14 an, bis sie durch einen weiteren Sprungbefehl oder durch das RESET-Signal geändert bzw. gelöscht werden.

Bei Nicht-Sprungbefehlen (die 13 Adreß-Bits geben den entsprechenden Speicherplatz in der aktuellen Page an) können mit Hilfe der indirekten Adressierung Speicherplätze des gesamten Adreßbereichs (32 Kbyte) angesprochen werden, weil dann die effektive Adresse über alle 15 Adreß-Bits verfügt. In diesem Fall werden die beiden Page-Bits nur für die Zeit des Speicherzugriffs geändert und nehmen danach wieder den ursprünglichen Wert an.

Zu beachten ist weiterhin, daß ein auftretender Interrupt den Prozessor immer in den niedrigsten Adreßbereich (Page 0) zwingt.

Die Zusammenschaltung von Speicherbausteinen

Bei einem Speicherbaustein ist es nicht nur wichtig zu wissen, wie viele Bits er speichern kann, sondern wie diese Bits organisiert sind.

Nehmen wir an, ein Speicherbaustein könne 4096 Bits speichern, so sagt diese Angabe allein über die Verwendungsmöglichkeit noch nichts aus. Wichtig zu wissen ist, ob Bits zu Wörtern oder Teilen von Wörtern zusammengefaßt sind, und wenn ja, wie viele. Daher gibt man die interne Organisation der Bits als $X \times Y$ an; X bezeichnet dabei die Anzahl der Adressen, Y die Anzahl der von einer Adresse angesprochenen Bits.

1. Möglichkeit

Die 4096 Bits sind **einzeln** mit den Adressen 0_{10} bis 4095_{10} zu erreichen, dann ist der Speicherbaustein als $4\text{ K} \times 1\text{ Bit}$ organisiert. Wenn man diesen Baustein für einen 8 Bit breiten Datenbus verwenden will, muß man acht Stück davon verwenden (Bild 9.3). Alle werden parallel an die Adreßleitungen A0 bis A11 angeschlossen, ebenso werden alle Steuerleitungen parallelgeschaltet. Jeder Baustein übernimmt eine Datenleitung (D0 bis D7).

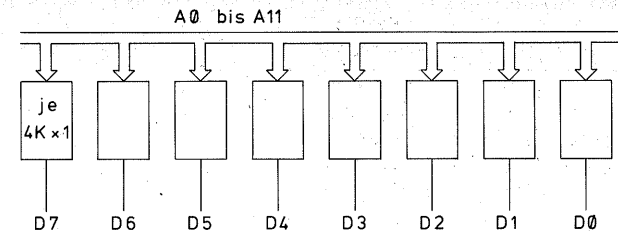


Bild 9.3 4-K-Speicher aus acht 1-Bit-Bausteinen

2. Möglichkeit

Die 4096 Bits sind als $1\text{ K} \times 4\text{ Bit}$ organisiert. Dann benötigt man für einen 8-Bit-Datenbus nur zwei Bausteine, hat aber nur einen Adressierbereich von 1 Kbyte (A0 bis A9; Bild 9.4). Wenn man auf 4 Kbyte kommen will, benötigt man wiederum acht Bausteine – nur kann man den Speicherbereich eben auch kürzer halten.

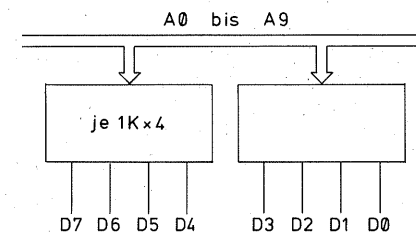


Bild 9.4 1-K-Speicher aus zwei 4-Bit-Bausteinen

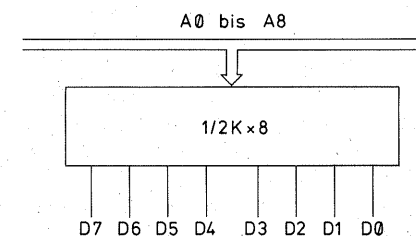


Bild 9.5 1/2-K-Speicher mit einem 8-Bit-Baustein

3. Möglichkeit

Die 4096 Bits sind als $512 \times 8\text{ Bit}$ organisiert. Für einen 8-Bit-Datenbus benötigt man nur noch einen einzigen Baustein. Dafür kommt man aber auch nur $1/2\text{ Kbyte}$ weit (Bild 9.5). Für einen Speicherumfang von 4 Kbyte sind wieder acht Bausteine nötig.

Aus den Bildern 9.3 und 9.4 geht hervor, wie Speicherbausteine zum Zweck der Busverbreiterung parallelgeschaltet werden können. Zur Erweiterung des Speicherraumes, also der Anzahl der Adressen, können Speicherbausteine **hintereinandergeschaltet** werden.

Bild 9.6 zeigt das Schaltungsprinzip an 1-K-Bausteinen. Das Prinzip ist aber überall gleich, unabhängig davon, ob man $1/4\text{-Kbyte}$ - oder 4-Kbyte -Bausteine oder noch größere verwendet.

Wir betrachten den einfachsten Fall, in dem alle hintereinanderschaltenden Bausteine den gleichen Adreßumfang haben. Um 1 Kbyte zu adressieren, sind die Adreßleitungen A0 bis A9 erforderlich.

Schauen Sie nun auf Tabelle 9.2a: Das 1. Kbyte benötigt nur die Adreßleitungen A0 bis A9. Betrachten Sie nun die höherwertigen Adreßleitungen A10, A11, ... als neue Dualzahl. Für das 1. Kbyte ist die neue Dualzahl: $0_2 = 0_{10}$.

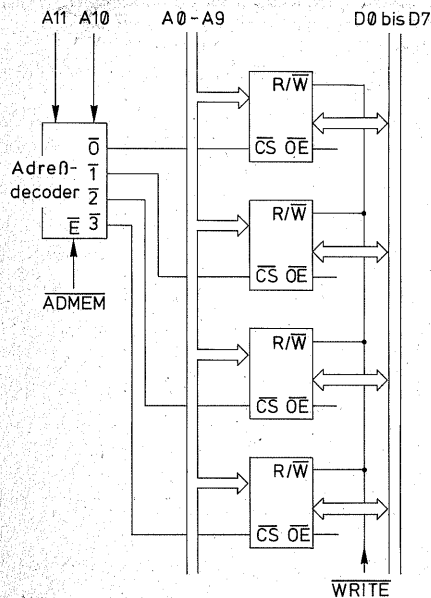


Bild 9.6 Hintereinanderschaltung von Speicherbausteinen zur Erweiterung des Speicherumfangs

Das 2. Kbyte erfordert wieder die Adreßleitungen A0 bis A9, die vorangehende Dualzahl ist nun aber $1_2 = 1_{10}$.

Das 3. Kbyte benötigt wieder dieselben Endadressen, aber vorangehend $1_2 = 3_{10}$.

Und so kann es weitergehen, bis der Adressierbereich vollkommen ausgenutzt ist.

Wenn man die den niedrigerwertigen Adreßleitungen vorangehenden Leitungen in einem 1-aus-n-Code decodiert, erhält man für jeden Speicherbaustein ein Steuersignal.

Alle Speicherbausteine haben einen Freigabeeingang, der meist mit \overline{CS} (chip select, engl. Bausteinauswahl) bezeichnet wird. Der Speicherbaustein wird nur dann aktiviert, wenn

dieser Eingang L-Signal erhält. Dieses L-Signal liefert ein Decoder, der die nächsthöheren Adreßleitungen decodiert.

In Bild 9.6 sollen vier Bausteine nacheinander gesteuert werden. Dafür werden die beiden nächsthöheren Adreßleitungen A10 und A11 decodiert (1-aus-4-Code):

A11	A10	$\bar{0}$	$\bar{1}$	$\bar{2}$	$\bar{3}$
L	L	L	H	H	H
L	H	H	L	L	H
H	L	H	H	L	H
H	H	H	H	H	L

Diese Tabelle zeigt, wie das \overline{CS} -Signal nach der Reihenfolge der vorangehenden Dualzahl in A10/A11 jeweils einen aktivierenden L-Pegel erzeugt, so daß die vier Speicherbausteine in Bild 9.6 in dieser Reihenfolge **nacheinander einzeln** aktiviert werden. Es werden nie zwei zugleich aktiviert. **Daher ist es möglich, alle Bausteine parallel an den Datenbus und an die niedrigerwertigen Adreßleitungen anzuschließen.**

Es soll aber nicht verschwiegen werden, daß die Decodierung mit nur zwei höherwertigen Adreßleitungen unvollkommen ist. Wenn man ganz sichergehen will, muß man in den Decoder alle vorangehenden Adreßleitungen einbeziehen.

Wie funktioniert der Decoder?

Bild 9.7 zeigt den Logikplan des 1-aus-8-Decoders 74LS138. Er hat drei Eingänge A0, A1 und A2, welche die Dualstellen 2^0 , 2^1 und 2^2 bezeichnen. Auf die Eingänge folgen Inverter, so daß intern sowohl die Eingangssignale (nach dem zweiten Inverter, doppelte Negation) als auch die invertierten Signale (nach dem ersten Inverter) zur Verfügung stehen. Der Rest ist eine einfache UND-Ver-

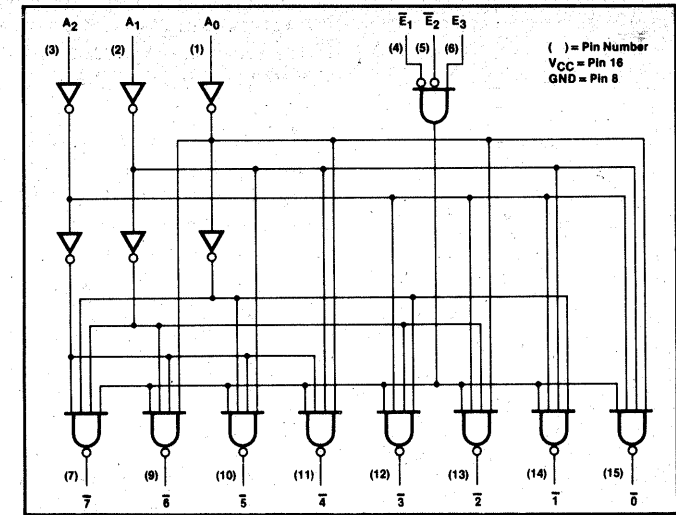


Bild 9.7 Logikplan des 1-aus-8-Decoders 74LS138 (nach VALVO-Unterlagen)

knüpfung, durchgeführt mit je einem NAND-Gatter, so daß das decodierte Signal **aktiv L** ist – gerade, wie es zur Steuerung der \overline{CS} -Eingänge benötigt wird:

$$\begin{aligned} \bar{0} &= \overline{A_2 \cdot A_1 \cdot A_0} \\ \bar{1} &= \overline{A_2 \cdot A_1 \cdot \bar{A}_0} \\ \bar{2} &= \overline{\bar{A}_2 \cdot A_1 \cdot A_0} \\ \bar{3} &= \overline{\bar{A}_2 \cdot A_1 \cdot \bar{A}_0} \\ &\text{usw.} \end{aligned}$$

Die NAND-Gatter verfügen jeweils über einen vierten Eingang, mit dem sie gesperrt werden können. Dieser Steuereingang ist wiederum die Zusammenfassung mehrerer Steuereingänge, mit denen man bestimmen kann, ob der Decoder decodieren soll oder nicht. Das kann man von höherwertigen Adressen abhängig machen und/oder von einem anderen Steuersignal, z. B. dem Signal \overline{ADMEM} (Bild 9.6), das auf diese Weise bestimmt, ob überhaupt ein Speicher

angesprochen wird. Wenn der Decoder gesperrt wird, erscheint an keinem Ausgang ein L, also wird auch kein Speicher freigegeben.

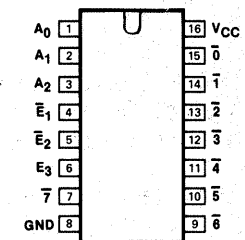


Bild 9.8 Anschlußbelegung des 74LS138 (nach VALVO-Unterlagen)

Die Funktionstafel (Bild 9.9) zeigt, daß der Decoder 74LS138 nur dann decodiert, wenn die E-Eingänge (E von enable, engl. Freigabe) E_1 UND $\overline{E_2} = L$ UND $E_3 = H$ sind.

Die Bezeichnung der Adreßeingänge A0 bis A2 ist relativ zu sehen. A0 wird an die Adreßleitung angeschlossen, die als nächste auf die unmittelbar zu

INPUTS						OUTPUTS							
E_1	E_2	E_3	A_0	A_1	A_2	$\bar{0}$	$\bar{1}$	$\bar{2}$	$\bar{3}$	$\bar{4}$	$\bar{5}$	$\bar{6}$	$\bar{7}$
H	X	X	X	X	X	H	H	H	H	H	H	H	H
X	H	X	X	X	X	H	H	H	H	H	H	H	H
X	X	L	X	X	X	H	H	H	H	H	H	H	H
L	L	H	L	L	L	L	H	H	H	H	H	H	H
L	L	H	H	L	L	L	H	L	H	H	H	H	H
L	L	H	H	H	L	H	H	H	L	H	H	H	H
L	L	H	L	L	H	H	H	H	H	L	H	H	H
L	L	H	H	L	H	H	H	H	H	H	L	H	H
L	L	H	L	H	H	H	H	H	H	H	H	L	H
L	L	H	H	H	H	H	H	H	H	H	H	H	L

NOTES
H = HIGH voltage level
L = LOW voltage level
X = Don't care

Bild 9.9 Funktionstabelle des 74LS138
(nach VALVO-Unterlagen)

den Speicherbausteinen führenden Adreßleitungen folgt, also A10 in Bild 9.6. A1 und A2 decodieren die nächsthöheren, also A11 und A12.

Wie alle Bausteine, die an einen Datenbus angeschlossen werden, verfügen die Speicherbausteine über Tri-State-Ein-/Ausgänge für die Datenleitungen. Diese werden über den Steuereingang \overline{OE} (output enable, engl. Ausgangsfreigabe) aktiviert oder in den hochohmigen Zustand geschaltet, was eine elektronische Abtrennung vom Datenbus bedeutet. Im einfachsten Fall schaltet man den \overline{OE} -Eingang mit dem \overline{CS} -Eingang desselben Bausteins zusammen.

Schreib-/Lesespeicher (RAM, siehe Seite 155, 159) verfügen noch über einen Eingang, der die Richtung des Datenflusses steuert (Bild 9.15). Dieser also bestimmt, ob Daten in den Speicher eingeschrieben werden (W, to write, engl. schreiben) oder aus ihm gelesen (R, to read, engl. lesen) werden. Der R/\overline{W} -Eingang wird oft auch mit \overline{WE} (write enable, engl. Schreibfreigabe) bezeichnet. Die R/\overline{W} -Eingänge der Bausteine können parallelgeschaltet und gemeinsam durch die \overline{WRITE} -Leitung gesteuert werden. Wirksam wird das Steuersignal ja nur bei dem Baustein, der durch $\overline{CS} = L$ aktiviert ist.

Speichertypen

Es gibt eine große Anzahl von Speichertypen. Sie auch nur annähernd alle zu behandeln, würde über den Rahmen dieser Einführung bei weitem hinausgehen. Was man alles zum Speichern digitaler Signale benutzen kann – selbst wenn man sich auf Halbleiterbauelemente beschränkt, angefangen von der einzelnen Diode über den Thyristor bis hin zu den hochintegrierten Schaltungen! Daher soll sich die Beschreibung auf die wichtigsten Halbleiterspeicher beschränken. Einen zusätzlichen Speicher bildet dann noch das Kassetteninterface zusammen mit der Tonbandkassette.

Wichtig für Sie ist zunächst die Unterscheidung zwischen **flüchtigen** und **nichtflüchtigen** Speichern. Begegnen Sie diesem Unterschied sicherlich schon, z.B. beim Umgang mit dem Taschenrechner. Wenn Sie diesem eine Zahl eingeben, so behält er sie nur so lange, wie die Betriebsspan-

nung eingeschaltet bleibt. Schalten Sie den Rechner aus, so ist auch die eingegebene Zahl verschwunden. Sie stand eben in einem flüchtigen Speicher. Die Flüchtigkeit mag Ihnen als Nachteil erscheinen, Sie können dafür aber auch den Inhalt des Speichers schnell und beliebig oft ändern. Speicher dieser Art heißen RAM (von random access memory, engl. Speicher mit wahlfreiem Zugriff). „Wahlfrei“ heißt, daß man je nach Belieben in diesen Speicher Daten einschreiben oder aus ihm lesen kann.

Einer anderen Speicherart begegnen Sie, wenn Sie eine der Funktionstasten des Taschenrechners bedienen, z. B. „+“, „-“, „×“ und „÷“. Dabei ist es gleichgültig, ob, wie oft oder wie lange die Betriebsspannung abgeschaltet war. Drücken Sie auf die Additionstaste, so läuft das Additionsprogramm ab. Es steht in einem nichtflüchtigen Speicher, einem sogenannten **Festwertspeicher**. Dieser Speicher verliert seinen Inhalt nicht, wenn die Betriebsspannung abgeschaltet wird, dafür kann er in der Regel auch nur gelesen werden. Er trägt deswegen auch die Kurzbezeichnung ROM (von read only memory, engl. Nur-Lesespeicher).

Das Taschenrechnerbeispiel stellt Ihnen nicht nur die zwei Hauptgruppen von Speichern vor, sondern auch deren typische Anwendungsgebiete: Ein **ROM (Festwertspeicher)** wird überall dort eingesetzt, wo dieselben Programme, dieselben Werte immer wieder unverändert benutzt werden, und die nach dem Einschalten des Geräts sofort ohne besondere Eingabe zur Verfügung stehen sollen. In unserem Computer wird z. B. ein Festwertspeicher benutzt, der u. a. ein Programm enthält, das die Siebensegmentanzeigen bedient. Dieses Pro-

gramm soll später immer zur Verfügung stehen. Darum ist es in einem Festwertspeicher untergebracht. Das **RAM** ist der typische **Arbeitspeicher**, in dem die **veränderlichen Werte** (beim Taschenrechner die verschiedenen Zahlen, Ergebnisse) untergebracht werden. Das RAM ist also der Speicher, in den Sie später Ihre eigenen Programme eingeben.

Die Struktur von Speicherbausteinen

Zur inneren Struktur eines Speicherbausteins gehören ein oder zwei **Adreßdecoder**, eine **Matrix** aus den decodierten Adreßleitungen und den Datenleitungen sowie ein **Eingangs-/Ausgangs-Tri-State-Puffer** mit einer **Steuerlogik**.

Bild 9.10 zeigt Ihnen die Strukturelemente eines wortorientierten Speicherbausteins in geraffter Weise:

Der **Adreßdecoder** formt die als Dualzahlen ankommenden Adressen in einen 1-aus-n-Code um, so daß bei jeder Dualzahl nur eine einzige Adreßleitung der Speichermatrix (Adr) aktiviert wird. Im Prinzip funktioniert der Adreßdecoder wie der in Bild 9.7 beschriebene Decoder, nur daß er eben einen viel größeren Umfang hat. Speicher mit großem Adressierungsumfang verfügen über zwei Adreßdecoder. Der zweite Adreßdecoder faßt Speicherblöcke nach dem Prinzip zusammen, wie es in Bild 9.6 dargestellt ist.

Der **Eingangs-/Ausgangspuffer** gemeinsam der **Steuerlogik** ermöglicht es, die Richtung des Datenflusses zu bestimmen (R/\overline{W}), die Ausgänge in den leitenden oder hochohmigen Zustand

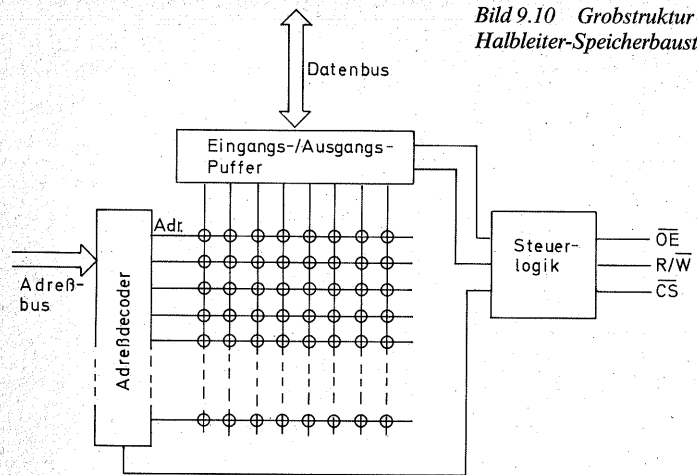


Bild 9.10 Grobstruktur eines Halbleiter-Speicherbausteins

zu bringen und dadurch zu entscheiden, ob der Baustein an den Datenbus angeschlossen oder elektronisch von ihm getrennt wird (OE), oder insgesamt zu entscheiden, ob der Baustein überhaupt benutzt wird oder nicht (CS).

Die Matrix aus den decodierten Adreß- und den Datenleitungen ist das eigentliche Speicherfeld. Jeder Kreuzungspunkt ist der Platz für 1 Bit. In Bild 9.10 sind als Beispiel acht Datenleitungen nebeneinander gezeichnet. Eine Adreßleitung bedient 8 Bit zugleich, der dargestellte Speicher ist also wortorientiert, und zwar als $n \times 8$ Bit (Bild 9.5).

Die Art, wie ein Speicherpunkt realisiert ist, entscheidet über den Speichertyp.

Festwertspeicher

Das ROM

Am einfachsten wird das Speicherfeld als Diodenmatrix aufgebaut (Bild 9.11). Die Datenleitungen liegen über einen Widerstand an +U, haben

also den Pegel H. Die Adreßleitungen werden im Falle der Aktivierung auf L geschaltet. Wo über eine Diode eine Verbindung geschaffen ist, kann die Datenleitung bei der Anwahl der zugehörigen Adresse auf L gezogen werden, wo die Diode nicht vorhanden ist, bleibt die Datenleitung bei der Anwahl durch die Adreßleitung auf H. Die Dioden sind zur Entkopplung der Leitungen notwendig (siehe die Entkopplung der Datenleitung, Seite 102).

Die Diodenmaske wird vom Halbleiterhersteller nach Kundenwünschen produziert. Ein ROM dieser Art kommt für den Selbstbauer nicht in Betracht, zeigt dafür aber die Speicherstruktur sehr schön.

Das PROM

Eine Variante des ROM ist das PROM (programmable ROM, engl. programmierbares ROM). Darin sind alle Dioden vorhanden und über hauchdünne Nickelchromsicherungen mit der Daten- oder Adreßleitung verbunden (Bild 9.12). Die jeweils de-

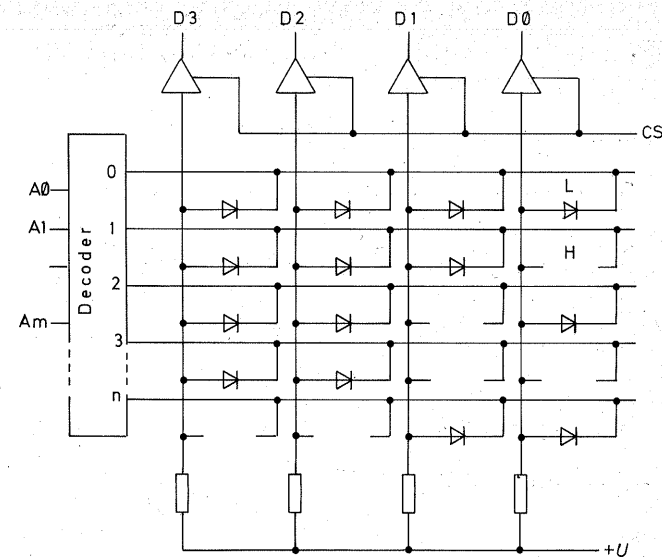


Bild 9.11 Aufbau eines ROM mit einer Diodenmatrix

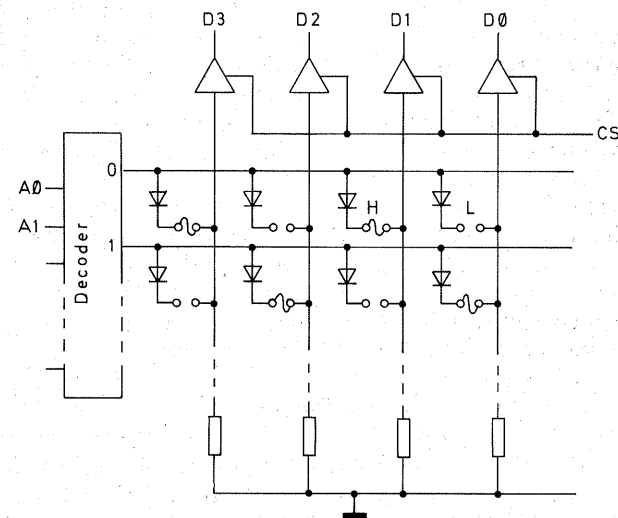


Bild 9.12 Aufbau eines PROM: Die fehlenden Verbindungen wurden beim Programmieren weggebrannt

codierte Adreßleitung ist H, während die Datenleitung über einen Widerstand auf L liegt.

Da zunächst alle Dioden vorhanden sind, haben alle Datenleitungen im unprogrammierten Zustand ein H. Zum Programmieren wird überall dort, wo ein L erscheinen soll, die Sicherung mit einem Stromstoß von einer Stärke, wie sie im Normalbetrieb mit Sicherheit nicht vorkommt, durchgebrannt. Nun liegt die Datenleitung nur noch über dem Widerstand auf L.

Auch das PROM ist für den Selbstbauer ein unwirtschaftlicher Speicherbaustein. Für die Industrie ist es natürlich sehr interessant, weil sie dann ihre Programme nicht mehr außer Haus geben muß.

Das EPROM

Der (gegenwärtig) ökonomischste Festwertspeicher für den „Einzelfall“ ist das EPROM (erasable PROM, engl. ausradierbares, löschbares PROM, Bild 9.13). Wieder sind die Knotenpunkte zwischen Adreßleitungen und Datenleitungen die Speicherpunkte. Die Verbindungen werden durch einen MOSFET (T1) und einen FAMOS-Transistor (floating avalanche-injection MOS, engl. Überflutungs-Stoßinjektions-MOS) hergestellt.

Der FAMOS-Transistor hat ein „schwebendes“ Gate, ein Gate ohne Anschluß (T2). Wenn das Gate nicht geladen ist (links in Bild 9.13), sperrt T2, und mag die Adreßleitung auch T1 durchschalten, so kann trotzdem kein H (+U) auf die Datenleitung (D1 in Bild 9.13) gelangen. Die Datenleitung führt L-Signal, der invertierende Ausgangspuffer formt es zu einem H um.

Mittels einer hohen Spannung (ca.

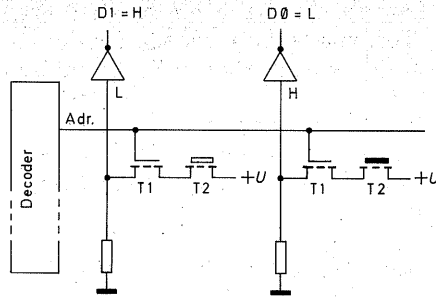


Bild 9.13 Speicherzelle des EPROM

25 V) kann das schwebende Gate von T2 durch den Tunneleffekt aufgeladen werden (rechts in Bild 9.13). Dadurch wird T2 leitend, und sobald die Adreßleitung T1 durchschaltet, erhält die Datenleitung über T1 und T2 +U (= H). Am Ausgang der Datenleitung steht wegen des Inverters ein L. Wegen der hervorragenden Isolation bleibt die Ladung für viele Jahre erhalten. Man darf damit rechnen, daß ein programmiertes EPROM für mindestens zehn Jahre brauchbar bleibt, denn innerhalb dieser Zeit soll das schwebende Gate nur etwa 30% seiner Ladung verlieren.

EPROM-Bausteine sind äußerlich leicht zu erkennen: Der Kristall ist durch ein Quarzglasfenster abgedeckt und damit sichtbar – zugleich aber auch, und darauf kommt es an, der Lichtstrahlung zugänglich.

Mit starker UV- oder Röntgenbestrahlung läßt sich nämlich die Eigenleitung des Kristalls so weit erhöhen, daß die Ladung innerhalb von 20 bis 30 Minuten abfließt. Dann ist das EPROM gelöscht und kann erneut programmiert werden. Normales Tageslicht reicht zum Löschen nicht aus, es sei denn, man legt das EPROM im Juni bei wolkenlosem Himmel im Hochgebirge oder an der See tage-

lang in die pralle Sonnenstrahlung. Trotzdem sollte man das Fenster eines programmierten EPROM abdecken.

Im unprogrammierten Zustand führen alle Datenleitungen H – wie beim PROM. Programmiert werden die L-Pegel.

Schreib-/Lesespeicher (RAM)

Bei RAM-Typen ist zwischen **dynamischen** und **statischen** Speichern zu unterscheiden.

Eine **dynamische** Speicherzelle besteht im wesentlichen aus einem integrierten Kondensator kleiner Kapazität (ca. 0,25 pF, Bild 9.14). Die Adreßleitung kann den MOSFET in den leitenden Zustand steuern und dadurch den Kondensator C mit der Spannung laden, die an der Datenleitung D steht. Bei H an D wird C geladen, bei L an D wird C nicht geladen.

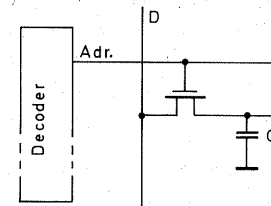


Bild 9.14 Speicherzelle eines dynamischen RAM

Zum Lesen steuert die Adreßleitung den MOSFET wieder in den leitenden Zustand, und die Kondensatorladung erscheint auf der Datenleitung als H oder L.

Dynamische Speicherzellen sind sehr klein, es lassen sich außerordentlich viele davon auf einem Kristall unterbringen. Sie haben aber einen Nachteil: Die Kondensatorladung hält nicht lange vor. Sie muß mindestens alle

2 ms „aufgefrischt“ werden. Übliche Auffrischungszeiten liegen in der Größenordnung von 60 µs. In diesem Zeitabstand werden alle Adreßleitungen zum Auffüllen der Kondensatoren durchlaufen.

Dynamische Speicherzellen sind also nur für einen schnellen Betrieb geeignet und kommen daher für unseren Computer, der ja beliebig langsam sein soll, nicht in Frage.

Fast alle Mikroprozessoren benutzen für interne Register dynamische Speicherzellen. Man kann diese Prozessoren deshalb nicht beliebig im Betrieb anhalten oder die Clockfrequenz herabsetzen oder gar jeden Impuls einzeln takten. Daher wurde für den hier beschriebenen Computer der MAB 2650 ausgewählt, der keine dynamischen, sondern statische Speicherzellen enthält.

Eine **statische** Speicherzelle besteht im wesentlichen aus einem D-Latch (siehe Seite 57), das von einer Steuerlogik umgeben ist (Bild 9.15).

Das D-Latch übernimmt mit dem Taktimpuls am Takteingang E (enable, engl. Freigabe) den Zustand des D-Eingangs und speichert ihn am

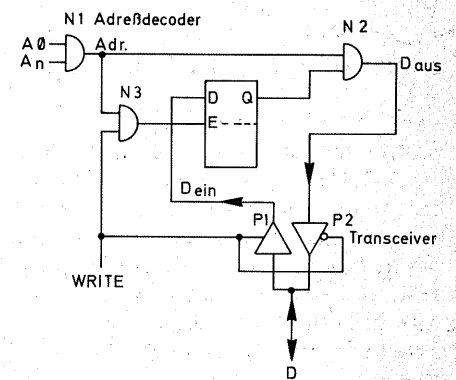


Bild 9.15 Speicherzelle eines statischen RAM

Ausgang Q so lange, wie die Betriebsspannung erhalten bleibt oder mit einem neuen Taktimpuls ein anderer Wert eingeschrieben wird.

N1 stellt – stark vereinfacht – den internen Adreßdecoder dar. Alle Aktivitäten der Speicherzelle sind davon abhängig, daß der Decoder diese Zelle anwählt.

1 Bit kann nur eingeschrieben werden, wenn die Decodierbedingung von N1 erfüllt ist UND das WRITE-Signal als Taktsignal aktiv ist.

Der Speicherwert Q erscheint als ein H am Ausgang D_{aus} nur dann, wenn der interne Adreßdecoder die Speicherzelle anwählt UND $Q = H$ ist. Hat Q den Wert L, so erscheint auch an D_{aus} ein L-Signal.

Das D-Latch hat aber zwei Datenleitungen, eine Eingangsleitung D_{ein} und eine Ausgangsleitung D_{aus} , der Datenbus verfügt aber nicht über getrennte Ein-/Ausgangsleitungen. Das Problem, mit nur einer Leitung für Ein- und Ausgang auszukommen, löst ein **Tri-State-Bustransceiver** (transmitter/receiver, engl. Sender/Empfänger). Es handelt sich dabei um zwei antiparallel geschaltete Tri-State-Puffer, die durch entgegengesetzte Signale (Invertierung an P2) jeweils leitend oder hochohmig gemacht, d.h. elektronisch abgetrennt werden.

Beim Schreiben leitet P1, und P2 ist abgetrennt; beim Lesen leitet P2, und P1 ist abgetrennt.

An eine intern decodierte Adreß- und Taktleitung (Ausgänge von N1 und N3) können weitere D-Latches angeschlossen werden, so daß mit einer Adresse ein ganzes Byte anzusteuern ist.

Das EAROM

Das Bindeglied zwischen RAM und ROM ist das EAROM (electrically alterable ROM, engl. elektrisch veränderbares ROM). Bisweilen werden Speicher dieser Gruppe auch als E^2 PROM (electrically erasable PROM, engl. elektrisch löschares PROM) bezeichnet. Das EAROM gehört zu den nichtflüchtigen Speichern, denn es verliert den Speicherinhalt nicht nach dem Abschalten der Betriebsspannung.

Das Geheimnis dieser Speicher ist wieder das schwebende Gate, wie beim EPROM. Es kann durch eine hohe Spannung von 21 oder 25 V – je nach Typ – geladen, aber eben auch wieder entladen werden. Der Vorteil gegenüber dem EPROM ist, daß man nicht mit der augenschädigenden UV-Strahlung gleich den gesamten Speicherinhalt löschen muß, sondern einzelne Bytes durch je einen Impuls löschen und anschließend neu programmieren kann.

Die Typen HN 48016 (Hitachi) und 2816 (Intel) sind anschlusgleich mit den in der Speicherplatte verwendeten Bausteinen 6116 und 2716 und können statt ihrer in die IC-Fassungen eingesetzt werden. Leider sind diese Bausteine noch sehr teuer und im Einzelhandel kaum erhältlich. Das kann sich aber rasch ändern, denn der Halbleitermarkt ist für jede Überraschung gut.

Die Schaltung der Speicherplatte

Die Schaltung des Speichers (Bild 9.17) weist keine Besonderheiten zu den bisher dargestellten Prinzipien auf. Für den Grundaufbau reicht ein RAM-Baustein, zur Erweiterung ist ein Festwertspeicher mit dem Betriebsprogramm („Monitor“) in Form eines EPROM vorgesehen. Das EPROM benötigen Sie aber erst, wenn Sie die Tastatur und Siebensegmentanzeige in Betrieb nehmen wollen.

Zur Wahl der Bausteine führten folgende Überlegungen:

- Für einen 8-Bit-Prozessor muß ein 8 Bit „breiter“ Speicher zur Verfügung stehen. Am einfachsten läßt dieser sich mit Bausteinen realisieren, die in einer Breite von 8 Bit wortorientiert sind.
- Der Arbeitsspeicher soll durch ein **statisches RAM**, der Festwertspeicher durch ein EPROM realisiert werden.

– RAM und EPROM sollen den gleichen Adressierungsbereich besitzen, damit eine einfache Decodierung der höheren Adreßleitungen ausreicht.

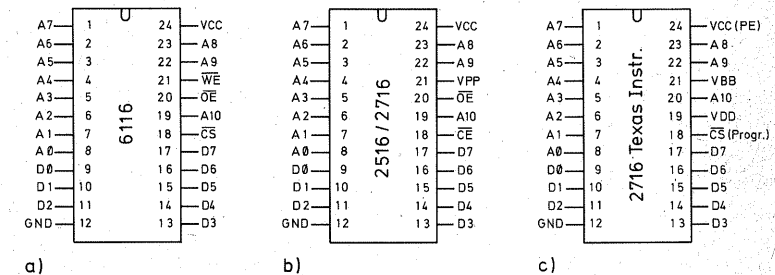
– Alle Bausteine sollen mit einer einzigen Betriebsspannung von +5 V auskommen. Es gibt nämlich auch Speicher, die zwei (+5 V, +12 V, z. B. 9208) oder gar drei Betriebsspannungen (+5 V, –5 V, +12 V, z. B. 2708) benötigen.

– Auch der Preis der Bausteine spielt eine Rolle: Wieviel kostet ein Speicherbereich von $1 K \times 8$ Bit? Relativ am preisgünstigsten für kleinere Speicher sind 2-K-Bausteine.

Nach diesen Überlegungen wurde als RAM-Baustein der 6116 – je nach Hersteller HM 6116 (Hitachi), IDT 6116 (IDT), IH 6116 (Intersil), UDN 6116 (Sprague) oder μ PD 4016 (NEC) – ausgewählt, als EPROM der Baustein 2716, der ebenfalls von verschiedenen Herstellern gefertigt wird. Die Bausteine sind anschlusgleich und untereinander austauschbar, wobei auf folgendes zu achten ist (Bild 9.16 a, b):

Der Anschluß WE (write enable), der beim RAM die Richtung des Datenflusses bestimmt, entspricht dem Anschluß VPP des EPROM (Programmierspannung), die beim Lesen auf H (= +5 V) liegen muß. Wird ein 6116

Bild 9.16 Anschlußbelegungen
a) des statischen RAM-Bausteins 6116,
b) des EPROM-Bausteins 2716 der meisten Hersteller, ihm entspricht der Typ 2516 von Texas Instruments, c) des EPROM-Bausteins 2716 von Texas Instruments



einen 2poligen UM-Schalter vertauscht werden. Damit ist es möglich, entweder das RAM (IC2, Schalterstellung 1) oder das EPROM (IC3, Schalterstellung 2) an den Anfang des Adressbereichs zu legen. Das ist aus folgendem Grund für Sie wichtig: Wenn Sie den RESET-Taster drücken, springt der Programmzähler des Mikroprozessors auf die Adresse 0000 zurück. Sofern Sie (noch) ohne Tastatur und Siebsegmentanzeige arbeiten wollen, benötigen Sie am Anfang des Adreßbereichs das RAM. Wollen Sie dagegen Tastatur und Anzeige benutzen, muß das EPROM mit dem Betriebsprogramm am Anfang des Adressbereichs stehen. Mit dem UM-Schalter ist die Speicherkarte für beide Betriebsarten geeignet.

Hinweise zum Aufbau

Bild 9.19 zeigt das Leiterbahnbild der Speicherkarte. Die Lötäugen für die Speicher-ICs müssen notgedrungen schmal sein. Bohren Sie sie daher möglichst genau an den durch das Ätzen bereits vorgekörnten Stellen und benutzen Sie dafür einen Bohrer mit 0,8 mm Ø, höchstens jedoch 1 mm Ø.

- | | |
|---|--|
| 1 | Leiterplatte |
| 1 | 31polige Stiftleiste, Baureihe GdsW |
| 1 | Fassung DIL 16 |
| 2 | Fassung DIL 24 (evtl. 4 Stück) |
| 1 | 74LS138 |
| 1 | 6116 |
| 1 | EPROM 2716 mit Monitor-Programm |
| 2 | Widerstand 1 kΩ, 0,125 W |
| 2 | keramischer Scheibenkondensator 0,1 µF |
| 1 | Elektrolytkondensator 47 bis 100 µF/16 V |
| 1 | kleiner Schiebeschalter 2 UM |

Bild 9.18 Stückliste für den Speicher

Die Bohrungen für die Stiftleiste müssen dicker sein (1 mm bis 1,3 mm Ø). Setzen Sie zuerst die Stiftleiste ein (Bestückungsplan Bild 9.20) und löten Sie sie stramm auf die Karte. Am besten verfahren Sie dabei wie beim Einlöten der Federleisten in die Busplatte (Bild 5.6), indem Sie erst einen der mittleren Stifte anlöten, dann das Zinn nochmals verflüssigen und mit einem Holzstab kräftig neben der Lötstelle auf die Leiterplatte drücken. Danach löten Sie entsprechend je zwei Lötstifte an den Enden der Leiste stramm an, zuletzt alle übrigen in beliebiger Reihenfolge.

Dann folgen die Drahtbrücken auf der Oberseite der Leiterplatte. Die **Drahtbrücke C - C** löten Sie noch nicht ein.

Wenn Sie die IC-Fassungen einlöten, sollten Sie den LötKolben unbedingt in Längsrichtung zu den Leiterbahnen halten (Bild 1.11). Da nämlich zwischen den Anschlußstiften jeweils eine Leiterbahn hindurchgeführt ist, sind die Abstände zwischen einem Lötauge und den beiden benachbarten Leiterbahnen sehr eng. Mit einem zu den Leiterbahnen quergestellten LötKolben sind Kurzschlüsse nicht zu vermeiden. Falls einmal eine Zinnbrücke entsteht, saugen Sie das Zinn mit (eventuell selbstgefertigter) Entlötlitze ab (Bild 1.15).

Von den beiden Widerständen ist im Bestückungsplan einer auf der Platine gezeichnet. Sie können aber auch beide Widerstände unmittelbar auf die Kupferbahnen löten.

Den UM-Schalter verdrahten Sie auf der Lötseite der Leiterplatte (Bild 9.21). Löten Sie an die sechs Anschlußbahnen zuerst 0,5 bis 0,6 mm starke Drähte. Achten Sie darauf, daß die beiden Schaltersektionen gut voneinander getrennt sind, bei allen übr-

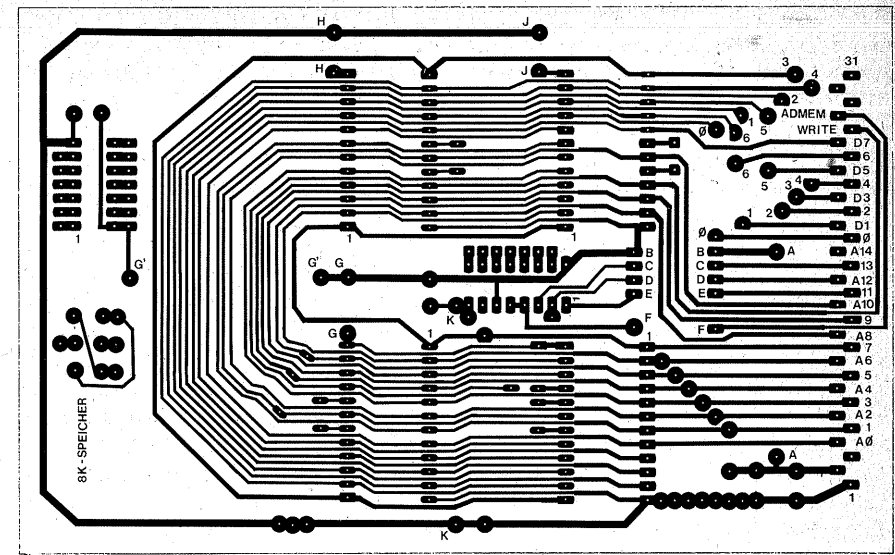


Bild 9.19 Leiterbahnbild der Speicherplatine

gen Schaltern, die Sie bisher eingelötet haben, sind sie parallelgeschaltet – hier nicht. Dann setzen Sie den Schalter ein, löten ihn erst an den Mittelanschlüssen fest, richten ihn aus und löten dann die restlichen vier Drähte an.

Den fertigen Platinenaufbau sehen Sie in Bild 9.22.

Hinweise zum Aufbau auf einer Experimentierplatte

Setzen Sie zuerst die Stiftleiste ein, und markieren Sie sich mit einem Faserschreiber auf den Kupferbahnen jeden fünften Stift; diese Orientierungshilfe erleichtert Ihnen die Verdrahtung erheblich. Setzen Sie die IC-Fassungen nicht zu dicht nebeneinander, sondern lassen Sie neben jedem IC-Anschluß Leiterbahnen mit einer Länge von drei Lochabständen stehen. Auf diese können Sie dann be-

quem die Daten- und Adreßleitungen auflöten.

Auf der Platine sind je zwei ICs mit Rücksicht auf die Leiterbahnführung spiegelbildlich zueinander angeordnet. Auf der Experimentierplatte ordnen Sie die ICs am besten in gleicher Richtung an. Das hilft Ihnen, Verdrahtungsirrtümer zu vermeiden.

Beim Verdrahten der Adreß- und Datenleitungen beginnen Sie am besten mit jeweils der Reihe, die der Stiftleiste zugewandt ist (Anschlüsse 1 bis 12 in Bild 9.21). Löten Sie zuerst „Doppelleitungen“ (Bild 1.27) an alle Adreß- und Datenanschlüsse von IC2, führen Sie das eine Ende einer Doppelleitung an die Stiftleiste, das andere an den parallelen Anschluß von IC3. Überprüfen Sie jeden Leitungszug sofort nach der „Abhakliste“ (siehe Seite 168) auf Richtigkeit und eventuelle Kurzschlüsse zu benachbarten Leiterbahnen.

Ehe Sie den UM-Schalter einlöten, trennen Sie die Leiterbahnen zwi-

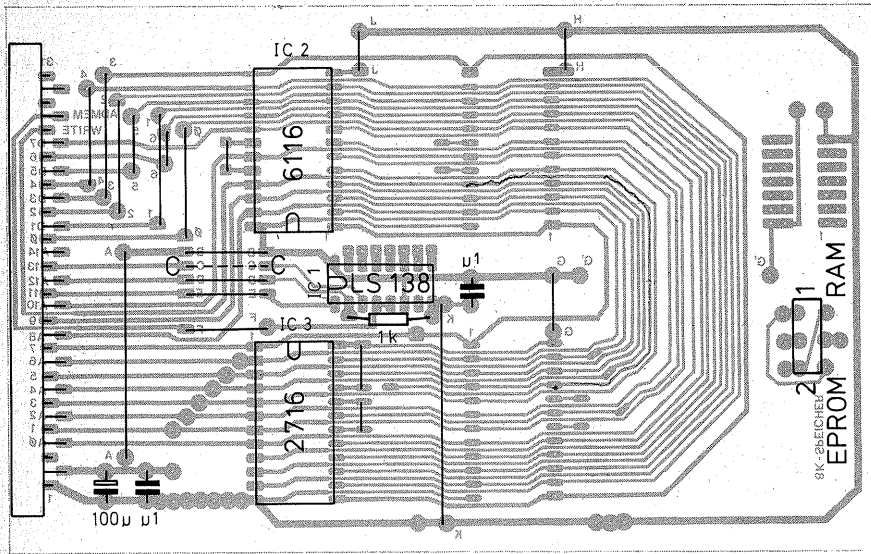


Bild 9.20 Bestückungsseite der Speicherplatine

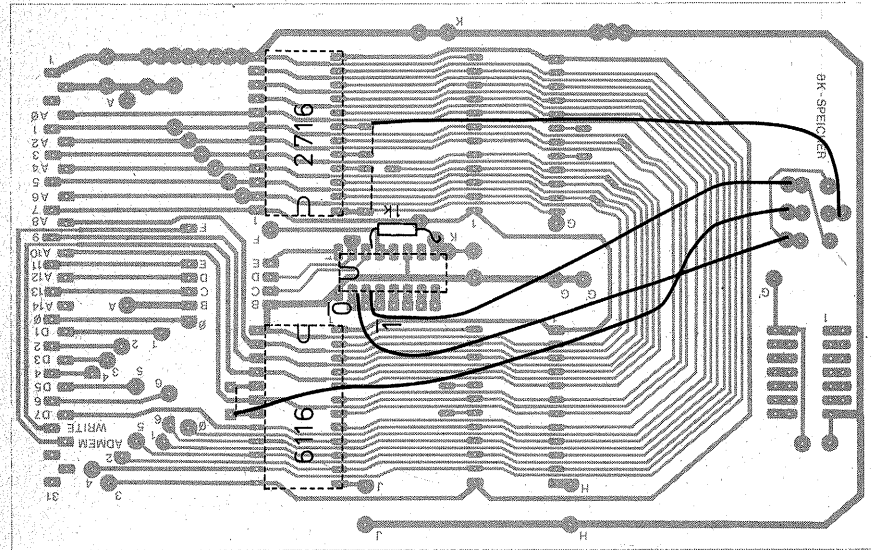


Bild 9.21 Die Verdrahtung auf der Lötseite der Speicherplatine

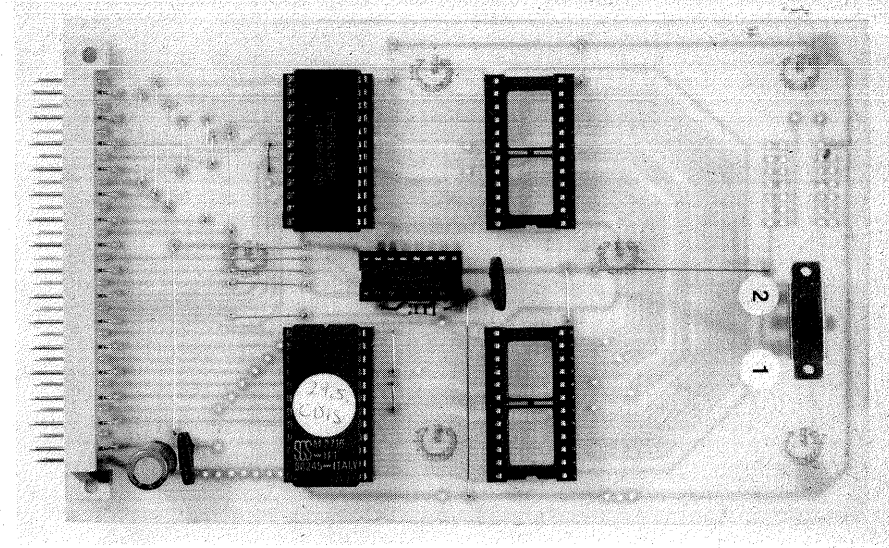


Bild 9.22 Der fertige Speicher

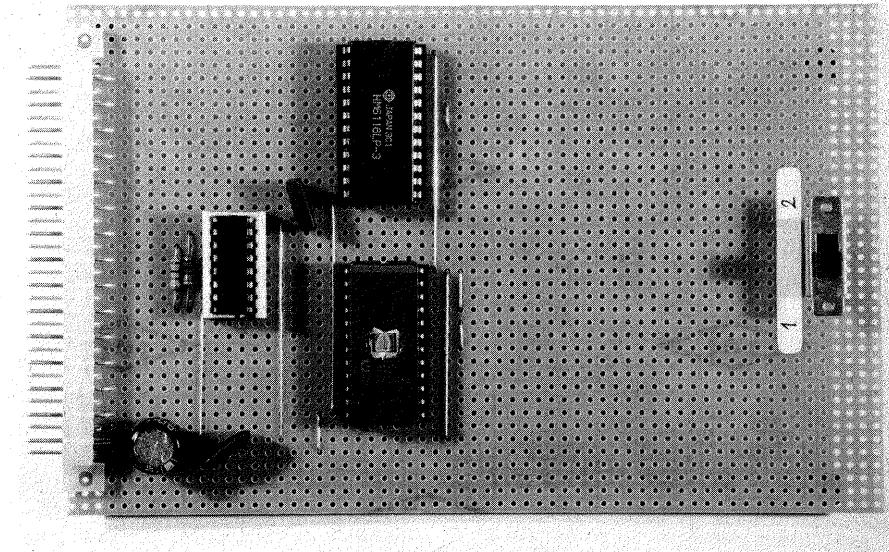


Bild 9.23 Der fertige Speicher auf einer Experimentierplatte

schen den beiden Anschlußreihen auf. Nach dem Einlöten des Schalters ist das recht schwierig. Bild 9.23 zeigt den fertigen Aufbau des Speichers auf einer Experimentierplatte.

Prüfen des Speichers

Prüfen Sie den Baustein zuerst **ohne** die ICs.

1. Prüfen Sie mit dem Ohmmeter, ob der UM-Schalter richtig arbeitet:

- Zwischen den Ausgängen 0 und 1 des Decoders (Pin 15 und 14 von IC1) müssen Sie $\infty \Omega$ messen.
- In der Schalterstellung „1“ (RAM) müssen Sie von Pin 15/IC1 nach den Anschlüssen 18 und 20 von IC2 Durchgang messen, desgleichen von Pin 14/IC1 nach den Anschlüssen 18 und 20 von IC3.
- In der Schalterstellung „2“ (EPROM) müssen Sie von Pin 15/IC1 Durchgang nach den Anschlüssen 18 und 20 von IC3 und von Pin 14/IC1 nach den Anschlüssen 18 und 20 von IC2 messen können.

2. Prüfen der Adreß- und Datenleitungen

Klemmen Sie ein Anschlußkabel Ihres Ohmmeters an Stift 4 der Stiftleiste ($\cong A0$), und klemmen Sie an das andere Anschlußkabel eine Stecknadel, damit Sie in die Buchsen der IC-Fassungen tasten können. Tippen Sie nun Anschluß 8 von IC2 und IC3 an, Sie müssen Durchgang messen können. Damit prüfen Sie, ob die Leiterbahn heil bzw. Ihre Verdrahtung richtig zugeordnet ist. Danach tippen Sie alle anderen IC-Anschlüsse an, dabei müssen Sie $\infty \Omega$ messen können.

Falls nicht, haben Sie einen Kurzschluß bzw. eine falsche Zuordnung in Ihrer Verdrahtung gefunden.

Entsprechend verfahren Sie mit allen übrigen Adreß- und Datenleitungen. Die folgende „Abhakliste“ soll Ihnen dabei helfen:

	Stiftleiste	IC2/3	IC1
A0	4	8	-
A1	5	7	-
A2	6	6	-
A3	7	5	-
A4	8	4	-
A5	9	3	-
A6	10	2	-
A7	11	1	-
A8	12	23	-
A9	13	22	-
A10	14	19	-
A11	15	-	1
A12	16	-	2
(A13)	17	-	3)
D0	19	9	-
D1	20	10	-
D2	21	11	-
D3	22	13	-
D4	23	14	-
D5	24	15	-
D6	25	16	-
D7	26	17	-
<u>WRITE</u>	27	21/IC2	-
<u>ADMEM</u>	28	-	4,5
0V	1	12	8
+5V	2	24	16

Bei der Adreßleitung A12 werden Sie auch an der Stiftleiste einen Widerstand von $1k\Omega$ gegen 0V messen können.

Der obige Test mag Ihnen umständlich erscheinen. Lassen Sie ihn aber nicht aus! Wie leicht schleichen sich bei einer Platine mit enger Leiterbahnführung in die schmalen Leiterbahnen Unterbrechungen oder zwischen ihnen Kurzschlüsse ein!

Falls Sie den Speicher auf einer Ex-

perimentierplatte aufbauen, sollten Sie den Abhakttest nach der Herstellung eines jeden Leitungszuges **sofort** durchführen.

3. Stecken Sie IC1 (74LS138) und IC2 (6116) in die Fassungen.

- **Überzeugen Sie sich, daß die ICs, besonders der RAM-Baustein, richtig herum in den Fassungen stecken.** Der 6116 – ebenso der 2716 – übersteht eine Falschpolung der Versorgungsspannung nicht! Ein solcher Irrtum wird teuer!
- Schieben Sie den UM-Schalter in Stellung „1“ (RAM).
- Ziehen Sie die Stiftleiste der CPU aus der Busplatte, dann sind Sie sicher, daß die CPU abgetrennt ist, und Sie den Speicher wirklich allein testen.

Die folgende Testanleitung ist zugleich die Bedienungsanleitung zum Laden und Kontrollen des Speichers, nur mit dem Unterschied, daß Sie später nicht jedesmal den CPU-Stecker ziehen müssen, sondern den Mikroprozessor dadurch (elektronisch) abtrennen, daß Sie ihn in den WAIT-Zustand laufen lassen.

An die Busplatte sind nun angeschlossen

- das Netzteil,
- die Dateneingabe und -anzeige,
- die Adreßeingabe und -anzeige und nun
- der Speicher (Karte in eine beliebige Federleiste stecken).

Schalterstellungen

Datenschalter Da: WRITE
Adreßschalter Adr: ADVAL

Anzeigen:

Die LED WRITE ist dunkel und zeigt damit \bar{W} an („Daten einschreiben“). Die LED ADMEM leuchtet (= H)

und zeigt damit an, daß der Speicher noch nicht aktiviert ist.

Stellen Sie nun mit den Schiebeschaltern die Adresse 00 ein; alle Adreß-LEDs sind dunkel.

Stellen Sie mit den Schiebeschaltern ein beliebiges Datenwort ein. Z. B. 04₁₆ (LODI, R0).

Drücken Sie auf den Taster M. Die LED ADMEM erlischt und zeigt an, daß nun der Speicher aktiviert wird, genauer der Adreßdecoder wird freigegeben, und dieser aktiviert mit $\bar{0} = L$ an \bar{CS} den RAM-Baustein.

Sobald Sie den Taster M loslassen, zeigt die LED ADMEM wieder H an. Jetzt prüfen Sie, ob der Speicher Ihr Datenwort angenommen hat.

Schieben Sie den Datenschalter Da in Stellung DAFLOT, und drücken Sie wieder den Taster M. Nun erscheint das Datenwort, das Sie eben eingegeben haben, in den Daten-LEDs.

Wenn Sie alle vorangegangenen Tests erfolgreich durchgeführt haben, dürfte auch jetzt der Erfolg nicht ausbleiben. Falls doch, gibt es nur wenige

Fehlermöglichkeiten

1. Stecken wirklich **alle** IC-Pins in der Fassung? Es ist mehrfach vorgekommen, daß sich beim Einsetzen des RAM-Bausteins ein IC-Anschluß umgebogen hatte.

2. Geben alle Fassungsanschlüsse guten Kontakt? Eine Speicherplatte machte erhebliche Schwierigkeiten. Die Ursache war eine IC-Fassung, die unsicheren Kontakt mit den IC-Pins gab. Sie können die Kontaktgabe mit dem Ohmmeter analog zur Abhakliste prüfen, indem Sie nun nicht die Federn der IC-Fassungen antippen, sondern die IC-Pins selbst.

3. Decodiert der Decoder? Im Ruhezustand müssen Sie an den Anschlüssen 18 und 20 von IC2 ein H messen.

Sobald Sie den Taster M drücken, muß die Spannung von H auf L springen. Wenn nicht, messen Sie an Anschluß 15 ($\bar{0}$) von IC1. Wenn dort der Pegel von H auf L springt, kann die Fehlerursache nur noch im UM-Schalter bzw. dessen Verdrahtung liegen. Auf einer Speicherkarte schaltete dieser Schalter unzuverlässig, so etwas kommt schon einmal vor.

Wenn der Pegel an IC1 nicht von H auf L springt, arbeitet der Decoder nicht richtig. Messen Sie dann, **welcher** Ausgang von H auf L wechselt, so finden Sie den Decodierfehler. Springt z.B. $\bar{4}$ auf L, so ist der Anschluß A2 (\cong A13) nicht wirklich auf L, springt $\bar{2}$ auf L, so ist der Anschluß A1 = H, und wenn $\bar{6}$ statt $\bar{0}$ decodiert wird, sind sowohl A1 als auch A2 = H – der Fehler liegt dann in den Widerständen. Wenn Sie einen Standardtyp (74138 statt 74LS138) verwenden, kann es nötig werden, die Widerstände von 1 k Ω auf ca. 680 Ω oder noch weiter zu verringern.

4. An weiteren Fehlermöglichkeiten gibt es nur noch kalte Lötstellen oder Kurzschlüsse. Bei der Suche nach Kurzschlüssen können Ihnen die Adreß- und die Dateneingabe helfen: Läßt sich wirklich **jede LED einzeln** ein- oder ausschalten? (Schalterstellungen **WRITE** bzw. **ADVAL**). Wenn Sie Kopplungen feststellen, haben Sie auch schon die fehlerhaften Leitungen gefunden.

Unterbrechungen finden Sie allerdings nur mit der obigen Abhakliste. Wenn alle Tests positiv verlaufen sind, stecken Sie die Stiftleiste der CPU wieder an ihren alten Platz.

Die Arbeit mit dem Speicher

Den Speicher laden

Stellen Sie zunächst Da auf DAFLOT und Adr auf ADFLOT. Lassen Sie den Mikroprozessor in den WAIT-Zustand laufen, indem Sie S3 auf LAUF und S4 auf HOLD stellen. Benutzen Sie nicht die Einzelschrittschaltung, denn mit ihr arbeitet der Mikroprozessor nur jeweils 1 Byte ab. Wird aber PAUSE (Anschluß 37) auf L geschaltet, so arbeitet der Prozessor den **gesamten** Befehl ab, ehe er in den WAIT-Zustand übergeht. Steht zufällig ein Mehr-Byte-Befehl an, so müßten Sie den STEP-Taster u. U. mehrfach betätigen, bis alle Bytes, die zum Befehl gehören, abgearbeitet sind. Steht S3 in der Stellung LAUF, so arbeitet der Prozessor so lange weiter, bis er den WAIT-Zustand erreicht hat.

Sie erkennen den Wait-Zustand daran, daß die LED RUN/WAIT erlischt und daß alle Adreß- und Daten-LEDs aufleuchten – aber nur mit den Schalterstellungen ADFLOT und DAFLOT. Dann wissen Sie, daß alle Adreß- und Datenanschlüsse des Prozessors, mit Ausnahme von A13 und A14, im hochohmigen Zustand, also vom Bus abgetrennt sind.

Nun stellen Sie die Schalter Da auf **WRITE** und Adr auf **ADVAL**. **Die Stellung ADVAL wird nur zum Laden oder Kontrolllesen des Speichers benutzt, sonst nie!**

Stellen Sie mit den Schiebeschaltern der Adreßeingabe 00 ein. Alle Programme müssen (vorerst) in der Adresse 00 beginnen, weil der Prozessor nach RESET bei dieser Adresse zu arbeiten anfängt.

Nun stellen Sie das gewünschte Datenwort ein.

Danach drücken Sie den Taster M. Damit übernimmt der Speicher das Byte von der Dateneingabeeinheit.

Prüfen Sie beim 1. Byte nach, ob der Speicher es angenommen hat, indem Sie Da auf DAFLOT stellen und M drücken. Dabei muß das Byte in den Daten-LEDs erscheinen. Danach schieben Sie Da in die Stellung **WRITE** zurück.

Schalten Sie nun die Adressen um einen Schritt weiter, stellen danach das folgende Byte auf der Dateneingabe ein, drücken M, und so geht es fort, bis Sie Ihr gesamtes Programm eingegeben haben.

Machen Sie es sich zur festen Regel, daß Sie erst die Adressen weiterschalten und danach die Daten, sonst bleiben Eingabefehler nicht aus. Meist bestehen sie darin, daß die Adressen nicht richtig eingestellt sind.

Schließen Sie – wenigstens am Anfang – jedes Programm mit dem **HALT-Befehl (40)** ab. Der Prozessor arbeitet ja unermüdlich weiter. Wenn er das eingegebene Programm abgearbeitet hat, arbeitet er ohne den **HALT-Befehl** weiter. Er findet in den folgenden Speicheradressen zufällige Bytes vor, die er seiner Befehlsstruktur gemäß als Befehle, Daten oder Adressen interpretiert. Er wird also zufällige Befehle ausführen, wahrscheinlich auch Sprung- und Store-Befehle, und wenn Sie Pech haben (und das ist bei der großen Anzahl der Speicherplätze sehr wahrscheinlich), überschreibt er Ihr mühsam eingegebenes Programm. **Also: Der letzte Befehl heißt immer HALT.**

Den Speicherinhalt lesen

Das Eingabeverfahren zeigt zwar sehr deutlich alle Vorgänge, die zum Laden des Speichers erforderlich sind, es zieht aber auch Fehler mit geradezu magischer Gewalt an. Überprüfen Sie daher Ihr Programm, indem Sie es Byte für Byte aus dem Speicher lesen. Stellen Sie Da in Stellung DAFLOT, lassen Sie Adr in Stellung **ADVAL**. Stellen Sie die Adresse 00 ein, und drücken Sie M – das unter der Adresse 00 gespeicherte Byte erscheint in den Daten-LEDs. Stellen Sie nun Adresse um Adresse weiter, drücken Sie jedesmal M, und lesen Sie jedes Byte.

Wenn ein Byte nicht richtig ist, geben Sie es korrigiert ein, indem Sie Da wieder auf **WRITE** schieben, das gewünschte Byte mit den Datenschaltern einstellen und wieder den Taster M drücken. Danach schieben Sie Da wieder in Stellung DAFLOT und lesen Ihr Programm weiter.

Ein Programm abarbeiten lassen

Wenn das Programm richtig im Speicher steht, **schieben Sie den Schalter Adr in Stellung ADFLOT und, sofern nicht bereits geschehen, Da in Stellung DAFLOT. Wenn Sie diese Umschaltung vergessen, wird Ihr Programm sofort zerstört, sobald Sie den Prozessor laufen lassen.**

Einstellungen auf der CPU-Platte:

- S1 : 1 Hz zum Beobachten, sonst 1 MHz
- S2 : OSZ
- S3 : LAUF
- RESET : 3 Clock-Pulse lang drücken, damit der Programmzähler auf die Startadresse 00 springt, danach oder währenddessen
- S4 : von HOLD auf RUN schieben.

Nun läuft das Programm ab.
 Sie können nun jedes beliebige Programm eingeben und sämtliche Befehle verwenden.

Lassen Sie z.B. folgendes Miniprogramm laufen:

Adr.			
00	75	0111 0101	CPSL, 03 C-Bit im PSL löschen
01	08	0000 1000	
02	04	0000 0100	LODI, R0 07 R0 mit Zahl 07 ₁₆ laden
03	07	0000 0111	
04	84	1000 0100	ADDI, R0, 10 zu (R0) 10 ₁₆ addieren
05	10	0001 0000	
06	F0	1111 0000	WRTE, R0 (R0) auf den Datenbus schreiben
07	40	0100 0000	HALT

Das Programm ist hier zur leichten Übersicht Byte für Byte bzw. Adresse für Adresse notiert. Normalerweise schreibt man einen ganzen Befehl (1, 2 oder 3 Bytes) in eine Zeile (siehe Programme ab Seite 255).

Wenn Sie nun den Ablauf des Programms beobachten, werden Sie sehen,

- wie der Programmzähler Adresse um Adresse weiterzählt;
- wie alle gespeicherten Bytes während der Taktzeiten T1 und T2 auf dem Datenbus erscheinen;
- daß aber alle LEDs während der Taktzeit T0 (OPREQ = L) aufleuchten, sich der Datenbus also im hochohmigen Zustand befindet (Bild 7.10).

Letzteres gilt auch für den Schreibbefehl: Das Ergebnis der Addition erscheint beim Befehl WRTE (ein Befehl mit zwei Maschinenzyklen = sechs Clock-Pulsen) für nur drei Taktzeiten sichtbar, dann ist es verschwunden. Sie können es überhaupt nur bemerken, wenn Sie den Prozessor im 1-Hz-Takt oder im Einzeltakt arbeiten lassen. Mit der für Prozessoren üblichen hohen Clockfrequenz von 1 MHz oder darüber würden Sie

von der Arbeit oder dem Ergebnis gar nichts wahrnehmen.

Sie benötigen daher eine Einrichtung, die das Ergebnis dauerhaft auffängt. Diese Einrichtung ist der **Portbaustein**.

Kapitel 10

Die Porteinheit (Baugruppe 7)

Ein Port (lat./engl. Tor, Ein-/Ausgang) ist ein Zugang zum Datenbus, der es ermöglicht, dem Datenbus Informationen zu entnehmen (Ausgangsport) oder auf ihn zu schreiben (Eingangsport). Eine Schaltung, die den Datenfluß in beiden Richtungen ermöglicht, ist ein Universalport.

Es gibt eine Reihe von sehr komfortablen, hochintegrierten Portbausteinen, die einen recht einfachen Hardware-Aufbau erlauben, doch sieht man wegen ihrer komplexen Schaltungen und der Vielfalt ihrer Möglichkeiten leicht den Wald vor lauter Bäumen nicht. Daher ist unsere Porteinheit aus mehreren Einzelbausteinen aufgebaut, die Schaltung selbst beschränkt sich auf das Wesentliche.

Schaltungstechnik

Das Schaltungsprinzip ergibt sich unmittelbar aus den Befehlen, die einen Prozessor zum Schreiben oder Lesen veranlassen. Der 2650 kennt je einen erweiterten („extended“) Schreib- bzw. Lesebefehl. „Erweitert“ heißt „2-Byte-Befehl“, im Gegensatz zu den 1-Byte-Befehlen WRTE/REDD und WRTE/REDC (siehe Seite 346-348). Das 1. Byte der erweiterten Befehle enthält den eigentlichen Schreib- oder Lesebefehl, das 2. Byte enthält eine Adresse, in die geschrieben oder aus der gelesen werden soll.

Die Befehle heißen:

WRTE (write extended)	1 1 0 1	0 1 r r	
aus Register	R0	D 4	Adresse
	R1	D 5	
	R2	D 6	
	R3	D 7	
REDE (read extended)	0 1 0 1	0 1 r r	
in Register	R0	5 4	Adresse
	R1	5 5	
	R2	5 6	
	R3	5 7	

Wenn das 2. Byte, die Adresse, abgearbeitet wird, erscheinen während der Taktzeiten T1 und T2

- auf dem Datenbus die Daten,
- auf dem Adreßbus das 2. Byte als Adresse.
- Außerdem sind
 - M/I \bar{O} = L
 - OPREQ = H,
 - woraus sich das Signal \overline{ADPER} = L ergibt.

Je nachdem, ob der Schreib- oder Lesebefehl benutzt wird, ist R/W H oder L, d.h. die Steuerleitung \overline{WRITE} ist L (schreiben) oder H (lesen).

Als Adresse des 2. Bytes können alle Möglichkeiten eines Bytes ausgenutzt werden. Es sind also 256 verschiedene Ports adressierbar.

Zum Bedienen eines Ports stehen als Signale zur Verfügung und müssen verwendet werden:

1. Die Adresse, welche decodiert wird und ein Steuersignal abgibt;
2. das Signal \overline{WRITE} , das über die Richtung des Datenflusses entscheidet, und
3. das Signal \overline{ADPER} , das überhaupt eine Peripherie (Port) öffnet.

Die Behandlung eines Ports unterscheidet sich prinzipiell nicht von der einer Speicherstelle. Tatsächlich gibt es weit verbreitete Prozessoren, die keine besonderen Schreib- oder Lesebefehle für die Peripherie kennen. Peripherien werden dann wie Speicherstellen behandelt, in die man hineinschreibt oder aus denen man liest. Die Befehle sind dann die Speicherbefehle STORE (die CPU schreibt in eine Speicherstelle) oder LOAD (Datentransport von einer Speicherstelle in die CPU). An die Stelle eines Signals \overline{ADPER} tritt dann ein Signal \overline{ADMEM} .

Auch beim 2650 kann man dieses als „memory mapped I/O“ (engl. in den

Speicher aufgenommener Ein-/Ausgang) bekannte Verfahren anwenden. Man muß dann aber einen bestimmten Speicherbereich dafür freihalten (und beim Programmieren immer daran denken!). Die besonderen Ein-/Ausgabebefehle sind also ein erfreulicher Luxus.

Die Möglichkeiten der Anschlüsse D/ \bar{C} (Pin 18) und E/ \bar{NE} (Pin 19) sind ein weiterer Komfort des 2650. Da das Prinzip der Portschaltung aber so allgemein wie möglich dargestellt werden soll, werden D/ \bar{C} und N/ \bar{NE} hier nicht behandelt.

Am Anfang jeder Portschaltung steht also – wie beim Speicher – ein Adreßdecoder. Dazu eignet sich z. B. der bereits dargestellte 74LS138 (siehe auch Kapitel 9, Bilder 9.7 bis 9.9). Damit kann man acht Adressen aus den 256 Möglichkeiten decodieren. Und so, wie im Speicher ein Decoderausgang jeweils einen einzigen Speicherbaustein aktiviert, öffnet im Portbaustein ein Decoderausgang jeweils nur einen einzigen Port.

Statt des 74LS138 wurde der 74LS154 als Decoder ausgewählt. Er hat vier Adreßeingänge (A0 bis A3) und kann dementsprechend 1-aus-16 decodieren. Bild 10.1 zeigt den Logikplan des 74LS154, Bild 10.2 die Anschlußbelegung. Das Schaltungsprinzip unterscheidet sich nicht grundsätzlich von dem des 74LS138: Die Adressen stehen nach ihrer einfachen bzw. doppelten Invertierung als \bar{A}_n bzw. A_n zur Verfügung und werden über NAND-Gatter zusammengefaßt. Daher ist der jeweils aktive Ausgang L. Ob eines der NAND-Gatter überhaupt auf L schalten kann, bestimmt wieder eine Steuerleitung, welche die Zusammenfassung der Steuereingänge \bar{E}_0 UND \bar{E}_1 ist. Nur wenn beide L sind (Invertierungen an den Eingängen

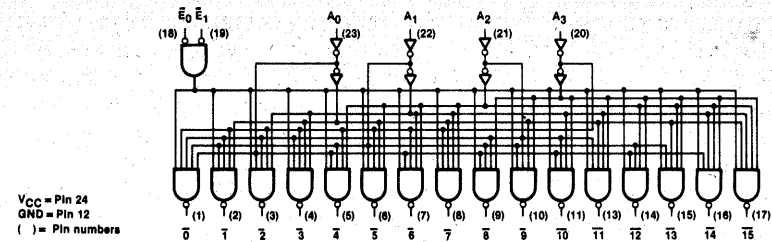


Bild 10.1 Logikplan des 74LS154 (nach VALVO-Unterlagen)

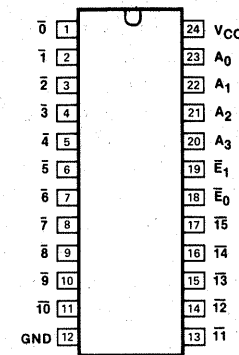


Bild 10.2 Anschlußbelegung des 74LS154 (nach VALVO-Unterlagen)

des UND-Gatters), können die NAND-Gatter des Decoders geöffnet werden. Bild 10.3 zeigt das Schaltverhalten als Funktionstabelle.

In Bild 10.4 ist der Stromlaufplan der Porteinheit dargestellt. IC1 ist der Adreßdecoder 1-aus-16, genauer 16 x 1-aus-32. IC1 decodiert die Adreßleitungen A0 bis A3, aber nur dann, wenn A4 L ist, denn A4 steuert den Eingang \bar{E}_1 .

Bild 10.3 Funktionstabelle des 74LS154 (nach VALVO-Unterlagen)

INPUTS						OUTPUTS																
\bar{E}_0	\bar{E}_1	A ₃	A ₂	A ₁	A ₀	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
L	H	X	X	X	X	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
H	L	X	X	X	X	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
H	H	X	X	X	X	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L	L
L	L	L	L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	H	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	H	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	H	L	L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	H	L	H	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	H	L	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	H	H	L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	H	H	H	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	L

H = HIGH voltage level
L = LOW voltage level
X = Don't care

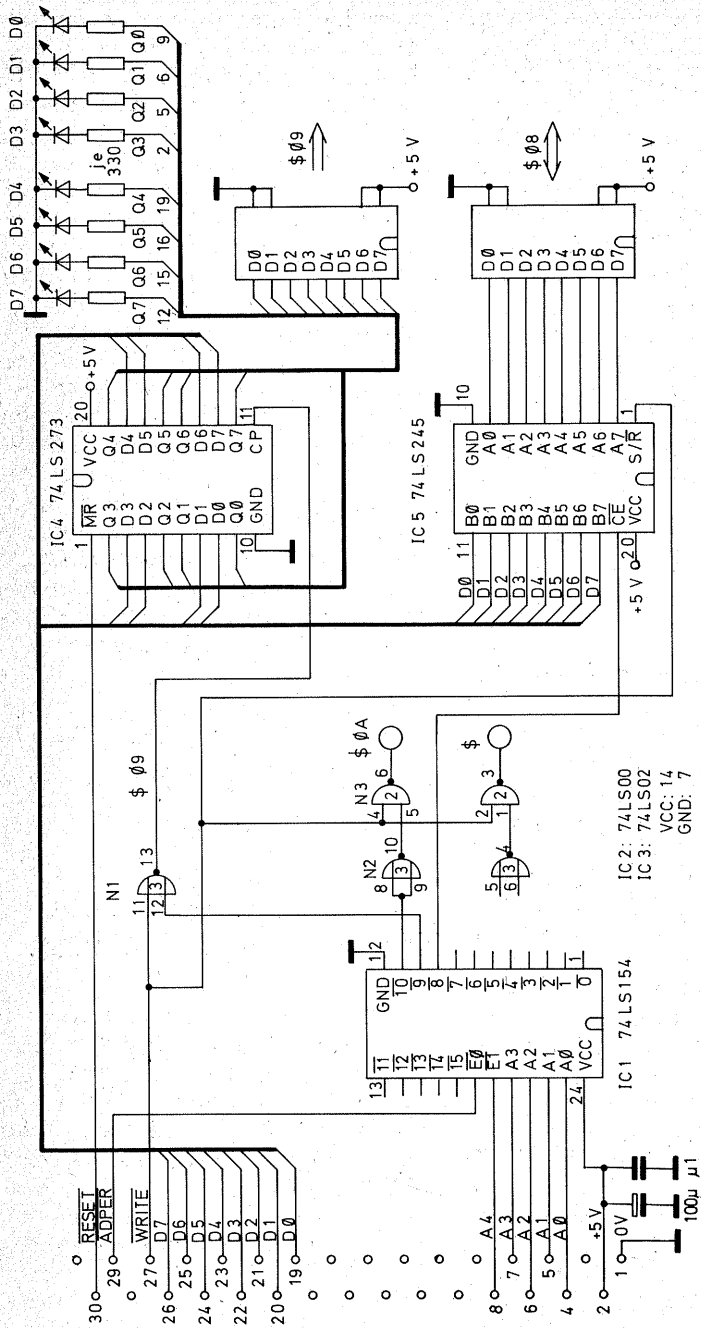


Bild 10.4 Stromlaufplan der Porteinheit

- | | |
|-----|--|
| 1 | Leiterplatte |
| 1 | 31polige Stiftleiste Baureihe GdsW |
| 1 | Lötinsel RTM 1.3 mit Steckhülse |
| 2 | Fassung DIL 14 |
| 2 | Fassung DIL 20 |
| 1 | Fassung DIL 24 |
| 2 | Testfassung mit Keil, DIL 16 EK |
| 2 | DIL-Stecker für Flachbandlitze DIS 16 |
| 1 | 74LS00 |
| 1 | 74LS02 |
| 1 | 74LS154 |
| 1 | 74LS245 |
| 1 | 74LS273 |
| 8 | LED |
| 8 | Widerstand 330 Ω, 0,125 W |
| (8) | Widerstand 1 kΩ, 0,125 W |
| 1 | Elektrolytkondensator 47 bis 100 µF/16 V |
| 1 | keramischer Scheibenkondensator 0,1 µF |

Bild 10.5 Stückliste für die Porteinheit

Die Decodierung ist unvollkommen, denn die 16 Möglichkeiten von A0 bis A3 wiederholen sich ja 16mal mit den höherwertigen Adreßleitungen A4 bis A7, so wie sich ja auch die unteren Speicheradressen zyklisch wiederholen (Tabelle 9.2). Durch die Einbeziehung von A4 für $\bar{E}1$ verringert sich die Anzahl der Wiederholungen auf 8 (A5 bis A7). Auch das ist noch mehr als genug, doch wenn man sich bei den Schreib-/Lesebefehlen auf die Adressen 00 bis 0F₁₆ beschränkt, können keine unerwünschten Effekte eintreten. Dafür läßt sich der Decodieraufwand auch gering halten. Wer dagegen alle 256 Adressen ausnutzen will, muß auch alle Adreßleitungen von A0 bis A7 in den Decoder einbeziehen. Um z. B. den 74LS154 so „dicht“ zu machen, daß er wirklich nur die Adressen 00 bis 0F₁₆ decodiert und keine Wiederholung, ist sicherzustellen, daß er gesperrt ist, wenn auch nur eine einzige Adreßlei-

tung des oberen Nibbles (A4 bis A7) H ist. Dazu genügt z. B. ein ODER-Gatter mit vier Eingängen. Sie erinnern sich: Schon ein einziges H an einem Eingang eines ODER-Gatters zwingt den Ausgang auf H. Wenn Sie den Ausgang dieses ODER-Gatters auf einen der Steuereingänge des Decoders schalten, gleichgültig ob auf $\bar{E}0$ oder $\bar{E}1$, so ist der Decoder für alle Adressen, die höher sind als 0F₁₆, gesperrt. Ein dafür geeigneter Baustein ist z. B. der 74LS25 (Bild 10.6). Er enthält zwei NOR-Gatter mit je vier Eingängen. Die beiden Gatter sind dann gemäß Bild 2.14b zu einem ODER-Gatter hintereinanderschalten, dabei müssen die Steuereingänge G auf H geschaltet werden. Doch wie bereits gesagt, nötig ist dieser Aufwand am Anfang sicherlich nicht, denn mit den 16 decodierten Portadressen können Sie schon sehr viel anfangen.

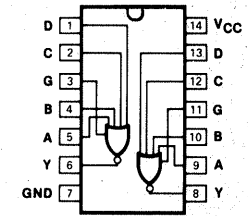


Bild 10.6 Anschlußbelegung des 74LS25. (nach VALVO-Unterlagen)

So wie die Leitung \overline{ADMEM} den Decoder des Speichers sperrt oder öffnet, steuert die Leitung \overline{ADPER} den Decoder der Porteinheit. Dazu ist der Steuereingang $\bar{E}0$ (Pin 18) mit \overline{ADPER} verbunden. Die Wahl aller weiteren Bausteine richtet sich danach, was ein Port jeweils können soll.

Ein Ausgangsport mit Auffangregister

Am Ende des vorigen Kapitels tauchte das Problem auf, daß das Ergebnis des Programms für nur wenige Mikrosekunden auf dem Datenbus erscheint und daher aufgefangen werden muß. Dazu dient das Auffangregister 74LS273 (IC4), das bereits als Beispiel für ein Register in Bild 2.28 vorgestellt wurde. Es enthält acht D-Flip-Flops, die in dem Augenblick, wenn der Clock-Pulse CP von L nach H wechselt, die Daten übernimmt, die an den D-Eingängen stehen. Bild 10.7 zeigt die Anschlußbelegung des 74LS273. Die Dateneingänge tragen die Bezeichnungen D0 bis D7, die zugehörigen Ausgänge die Bezeichnungen Q0 bis Q7. Zu D0 gehört Q0, zu D1 gehört Q1 usw. Die Flip-Flops sind untereinander gleichwertig und austauschbar. Die Namen ihrer Ein- und Ausgänge sagen nichts über ihre Verwendung aus. D0/Q0 muß nicht unbedingt für die Datenleitung D0 verwendet werden, man könnte D0/Q0 für jede andere Datenleitung verwenden. Die Ziffer bedeutet nicht wie bei den Ziffern des Datenbusses eine Zweierpotenz, sondern sie dient

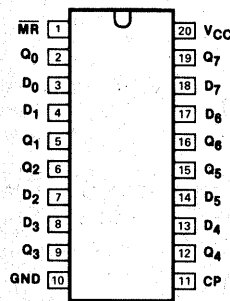


Bild 10.7 Anschlußbelegung des 74LS273 (nach VALVO-Unterlagen)

nur zur Unterscheidung der Flip-Flops, um zu einem Eingang auch den zugehörigen Ausgang zu markieren. Am besten macht man sich die Verhältnisse mit einem Pfeildiagramm klar, das den Datenfluß angibt (Bild 10.8). Gemeinsam sind den Flip-Flops nur die Clockleitung CP und die Rücksetzleitung MR (master reset)

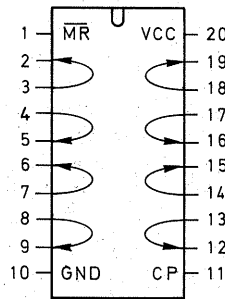


Bild 10.8 Datenfluß im Register 74LS273

Zum Setzen muß das Register am Clockeingang CP einen Spannungssprung von L auf H erhalten. Im Ruhezustand muß CP daher L sein. Wenn aber die decodierte Adresse (aktiv L) UND das Schreibsignal (WRITE aktiv L) zusammenfallen, dann soll CP von L auf H wechseln. Wenn aber auch nur eines der Steuersignale H ist, muß CP L bleiben.

Das Gatter, das diese Funktion erfüllt, ist das NOR-Gatter. Schon ein einziges H an einem der Eingänge zwingt den Ausgang auf L, und nur wenn alle Eingänge L sind, ist der Ausgang H (siehe Seite 50). Daher sind zur Datenübernahme WRITE und die decodierte Adresse 9 durch das Gatter N1 NOR-verknüpft. Die Adresse 9 ist willkürlich gewählt, jede andere zwischen 0 und 15 wäre möglich gewesen. Da diese Adresse für das Auffangregister IC4 auf der Plati-

ne festgelegt und auch im Betriebsprogramm fest eingeplant ist, merken Sie sich die Peripherie 09₁₆ (\$09, das Dollarzeichen steht für Peripherie) für den WRTE-Befehl, der in dieses Register schreiben soll. Das Register kann durch RESET an MR auf 00 gesetzt werden. Am Anfang trägt es zur Übersicht bei, wenn vor einem Programmstart die Anzeige völlig dunkel ist. Später kann es u. U. stören, wenn beim Drücken des RESET-Tasters auch eine im Port vorhandene Anzeige verschwindet, die man gern länger behalten hätte. Dann ist die RESET-Leitung zu unterbrechen. Auch auf der Platine ist sie eine Drahtbrücke (Lötseite, RST-RST, Bild 10.16), die leicht zu entfernen ist. Eventuell setzen Sie auch einen Schalter ein.

An den Ausgängen des Registers IC4 befinden sich die Anzeige-LEDs mit Vorwiderständen von 330Ω. Wenn die Bastelkasse es zuläßt, lohnt sich die Anschaffung von superhellen LEDs (siehe Seite 101), weil dann die Vorwiderstände vergrößert und somit die Ströme verringert werden können. Die Ausgänge sind nämlich über eine DIL-Fassung, in die ein DIL-Stecker paßt, nach außen geführt, damit über diesen Ausgangsport auch andere Peripheriegeräte, z. B. eine Ampelschaltung, ein Lautsprecher usw. gesteuert werden können. Dann ist es natürlich erstrebenswert, daß von dem Ausgangsstrom, den das Register zu liefern vermag, möglichst viel übrigbleibt.

Ein Universaleingangs-/Ausgangsport

Außerdem enthält die Baugruppe noch einen Universalport, der sowohl als Eingang als auch als Ausgang dienen kann.

Bei dem Auffangregister brauchte das Problem des Busanschlusses noch nicht berücksichtigt zu werden, weil ja nur **Eingänge** an den Bus angeschlossen sind. Bei einem **Eingangsport** liegen aber die **Ausgänge** des Bausteins am Bus. Der Eingangsbau- stein muß also über Tri-State-Ausgänge verfügen, damit er normalerweise vom Bus (elektronisch) abgeschaltet ist und nur für den Augenblick, in dem der Prozessor liest, mit dem Bus verbunden ist (siehe Seite 63, Bild 2.31). Hier wird, wie bereits beim Speicher, die Regel wirksam, daß **alle Ausgänge, die an einen Bus angeschlossen werden, ausnahmslos Tri-State-Ausgänge sein müssen**.

Tri-State-Puffer sind auch die Voraussetzung, daß ein Baustein als Eingang auf den Bus geschaltet werden darf. Oft sind zwei Puffer antiparallel zu einem Transceiver (transmitter/receiver, engl. Sender/Empfänger) vereinigt (Bild 10.9, vgl. auch Bild 9.15). Die beiden Steuereingänge verhalten sich durch die Negation eines Steuereingangs (P2) entgegengesetzt.

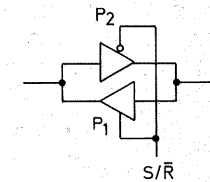


Bild 10.9 Prinzip des Transceivers aus zwei Tri-State-Puffern

Die Steuerleitung heißt S/R (send/receive, engl. senden/empfangen). Wenn P1 leitet, ist P2 hochohmig, und die Daten fließen von A nach B. Leitet P2, so ist P1 hochohmig, und die Daten fließen von B nach A. In dem Baustein 74LS245 sind acht

nichtinvertierende Transceiver vereinigt, gerade die richtige Anzahl für einen 8-Bit-Datenbus (Bild 10.10). Wie die Latches des Auffangregisters sind auch die Transceiver voneinander unabhängig und müssen nicht für bestimmte Datenleitungen benutzt werden (Bild 10.10c).

\overline{CE} ist wieder der chip-enable-Eingang. Über die UND-Gatter entscheidet er (Invertierungen beachten), ob das Steuersignal S/\overline{R} überhaupt wirksam werden kann. Über die Richtung des Datenflusses entscheidet S/\overline{R} (Invertierung am rechten UND-Gatter in Bild 10.10a beachten):

Wenn S/\overline{R} H ist, fließen die Daten von A nach B.

Wenn S/\overline{R} L ist, fließen die Daten von B nach A.

Dieser Bustransceiver bildet den Eingangs-/Ausgangsport auf unserer Porteinheit (IC 5 in Bild 10.4). Die A-Anschlüsse führen über einen DIL-16-Stecker nach draußen, die B-Anschlüsse liegen am Datenbus. Die Datenrichtung wird durch die Leitung \overline{WRITE} gesteuert.

Ist \overline{WRITE} H (lesen), so ist der Transceiver ein Eingang (A→B).

Ist \overline{WRITE} L (schreiben), so ist er ein Ausgang (B→A).

Ob der Baustein überhaupt aktiviert wird, darüber entscheidet \overline{CE} . Das \overline{CE} -Signal wird wieder aus der decodierten und durch \overline{ADPER} freigegebenen Portadresse erzeugt. Die Adresse 08_{16} ist willkürlich gewählt. Merken Sie sich aber, weil sie auf der Platine fest geschaltet ist. Das Decodersignal 8 wird diesmal **nicht** mit \overline{WRITE} verknüpft, weil \overline{CE} ja sowohl beim Schreiben als auch beim Lesen aktiv sein soll.

Achtung: Wenn dieser Port nicht benutzt wird, befinden sich alle Ausgangsleitungen im hochohmigen Zustand. Nachgeschaltete TTL-Eingänge interpretieren ihn als H. Aber auch einfache, nachfolgende Interface-Schaltungen (interface, engl. Verbindungstück), z. B. ein einfacher Transistoreingang (Bild 10.11a), können sich seltsam verhalten.

Ein Transceiver-Ein-/Ausgang ist im Tri-State-Zustand zwar hochohmig, aber nicht absolut dicht. Nicht nur,

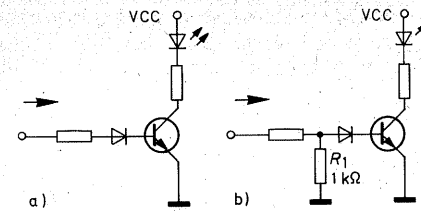


Bild 10.11 a) Einfaches Transistorinterface für eine LED-Anzeige, b) R_1 leitet den Eingangsstrom des Puffers ab

daß die Endstufentransistoren gewisse Leckströme zulassen, zum Ausgang ist ja ein Eingang parallelgeschaltet, und aus dem fließt ja ein Eingangsstrom heraus (siehe Seite 64). Er kann beim 74LS245 im Extremfall $-200 \mu A$ erreichen. Normalerweise ist er viel geringer und liegt in der Größenordnung von $-30 \mu A$. Dieser Strom kann aber einen Kleinleistungstransistor oder einen Darlington-Leistungstransistor durchschalten. Es ist daher nötig, den Strom gewissermaßen „abzuleiten“, indem man einen Widerstand $1 k\Omega$ gegen $0 V$ schaltet (R_1 in Bild 10.11b). Sie können dieses Problem ein für allemal beseitigen, indem Sie alle A-Anschlüsse des Transceivers mit Widerständen $1 k\Omega$ gegen $0 V$ ausstatten. Wie Sie das z. B. mit kleinen ($1/8 W$), senkrecht stehenden Widerständen bewerkstelligen können, zeigt Ihnen Bild 10.15. Eines müssen Sie dabei jedoch bedenken: Diese Widerstände wirken auch als Last für die Quelle, die der Prozessor über den Transceiver lesen soll. Das kann bei hochohmigen Quellen wieder zu Schwierigkeiten führen. Daher ist es besser, nur die über diesen Port zu steuernden Schaltungen, wie in Bild 10.11b dargestellt, gegebenenfalls mit den „Ableitwiderständen“ zu versehen.

Sie werden diesen Port anfangs wahrscheinlich noch nicht benötigen und brauchen daher auch IC5 nicht gleich zu kaufen. Die Schaltung soll Ihnen das Prinzip eines Eingangs-/Ausgangsports zeigen, zugleich auch die „kleinen“ Probleme, die oft großzügig übergangen werden und dann zu ernsthaften Schwierigkeiten führen.

Weitere Portadressen

Wenn ein Port durch ein L-Signal an \overline{CE} aktiviert wird, können Sie das Steuersignal unmittelbar dem Decoderausgang entnehmen. Benötigen Sie zum Setzen eines Registers ein H-Signal, so invertieren Sie den Decoderausgang, dann haben Sie im Ruhezustand ein L, im aktiven Zustand ein H.

Für einen Universalport können Sie **nur** das Decodersignal nehmen. Der Port wird dann immer geöffnet, wenn \overline{ADPER} UND die zugehörige Adresse erscheinen. Die Richtung des Datenflusses hängt dann vom Befehl ab, der die \overline{WRITE} -Leitung auf L (schreiben) oder H (lesen) setzt und damit S/\overline{R} steuert.

Soll ein Port aber **nur** Eingang oder **nur** Ausgang sein, dann würde es zwar auch genügen, \overline{CE} durch den Adreßdecoder allein zu steuern. Gönnen Sie sich aber den Luxus, die \overline{WRITE} -Leitung mit einzubeziehen. Dann öffnet der Port nur, wenn \overline{ADPER} UND Adresse UND Datenrichtung stimmen. Bei einer Verwechslung der Schreib-/Lesebefehle bleibt der Port dann zu. Es passiert zwar nichts, aber es kann auch kein Schaden durch Überschreiben von Daten entstehen.

Als Beispiel für diese Sicherung dient die Decodierung des Steuersignals für die Peripherie $0A_{16}$. Sie ist dazu vorgesehen, den Analog-Digital-

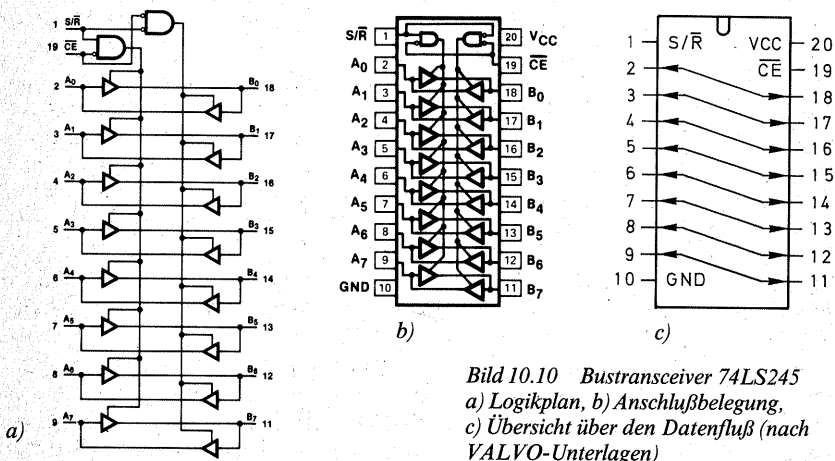


Bild 10.10 Bustransceiver 74LS245
a) Logikplan, b) Anschlußbelegung,
c) Übersicht über den Datenfluß (nach VALVO-Unterlagen)

Wandler zu lesen – nur zu lesen. Diese Adresse ist im Betriebsprogramm fest eingebaut; merken Sie sich daher bitte.

Zum Öffnen des Tri-State-Ausgangs (OE von IC1 in Bild 11.17) wird wieder ein aktiv-L-Signal benötigt. Das wird mit dem NAND-Gatter N5 (Bild 10.4) aus dem invertierten Decodersignal $\overline{10}$ UND dem H-Signal der WRITE-Leitung decodiert. Der Ausgang des NAND-Gatters schaltet nur dann auf L, wenn alle Eingänge H sind (siehe Seite 49). Beim Lesen ist WRITE H, bei der gewählten Adresse $0A_{16}$ ist der Ausgang $\overline{10}$ L, nach der Invertierung durch N4 jedoch H, deswegen ist genau dann der Ausgang $\$0A = L$ und kann den Analog-Digital-Wandler öffnen.

Auf der Platine ist noch Platz für zwei Fassungen DIL 20 vorgesehen, so daß Sie sich je nach Ihrem Bedarf weitere Ports einrichten können, als Transceiver wie beim Universaleingangs-/Ausgangsport, als Auffangregister zur Ausgabe wie Ausgangsport mit Auffangregister oder als Auffangregister, das ankommende Signale speichern soll. Solch ein Register muß dann Tri-State-Ausgänge haben. Geeignet ist z.B. der Baustein 74LS373, dessen Beschaltung Sie als Eingangsregister im Stromlaufplan des Analog-Digital-Wandlers finden (Bild 11.17). Dieses Register besteht aus transparenten Latches (siehe Seite 57). Das entsprechende, anschlussgleiche, aber aus flankengetriggerten Flip-Flops bestehende Register ist der Baustein 74LS374. Es übernimmt die Daten, wenn der Clockeingang CP (Pin 11 anstelle E beim 74LS373) von L auf H wechselt.

Außerdem finden Sie auf der Platine noch Platz für eine weitere Fassung DIL 16, so daß Sie einen zusätzlichen

Busstecker unterbringen können. Die vorhandenen NAND- und NOR-Gatter sind noch nicht ausgenutzt, so daß Sie sich Steuersignale, wie in den Beispielen beschrieben, decodieren können.

Hier sehen Sie auch die Grenzen einer Bauanleitung: Den idealen Computer gibt es ebensowenig wie das ideale Radio. Wenn Sie etwa im Omnibus ungestört und ohne selbst zu stören Radio hören wollen, benötigen Sie ein anderes Gerät, als wenn Sie zu Hause Ihre Lieblingsmusik in voller Lautstärke genießen. Wie Sie nun Ihre Porteinheit im einzelnen aufbauen, hängt davon ab, was Sie steuern oder lesen wollen. Da Sie aber die Konstruktionsprinzipien hier vorfinden, können Sie die Porteinheit nach Ihren Anwendungszwecken erweitern. Der Decoder liefert Ihnen die Steuersignale, der eigentliche Port kann sich, wie Sie beim Analog-Digital-Wandler sehen, auch auf einer anderen Baugruppe befinden.

Hinweise zum Aufbau

Beim Aufbau gibt es zu dem bereits bei den anderen Baugruppen Beschriebenen wenig Neues, das gilt auch für den Aufbau auf einer Experimentierplatte.

Bild 10.12 zeigt das Leiterbahnbild der Platine, Bild 10.13 die Bestückungsseite.

Beginnen Sie wieder mit dem Einsetzen der Stiftleiste und den Drahtbrücken auf der Bestückungsseite. Die übrigen Bauelemente können Sie in beliebiger Reihenfolge einsetzen.

Wenn Sie die Fassungen für die DIL-Stecker einsetzen, achten Sie darauf, daß sich die Keile vom Platinenrand

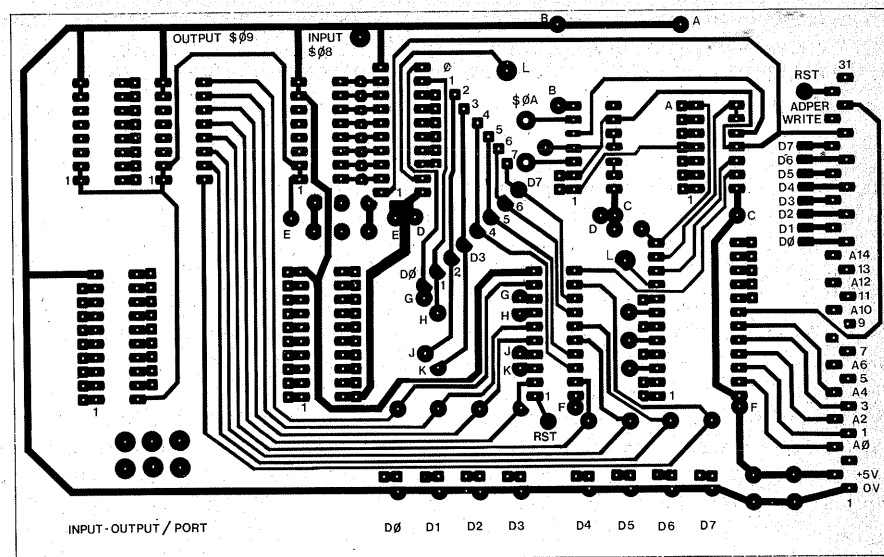


Bild 10.12 Leiterbahnbild der Porteinheit

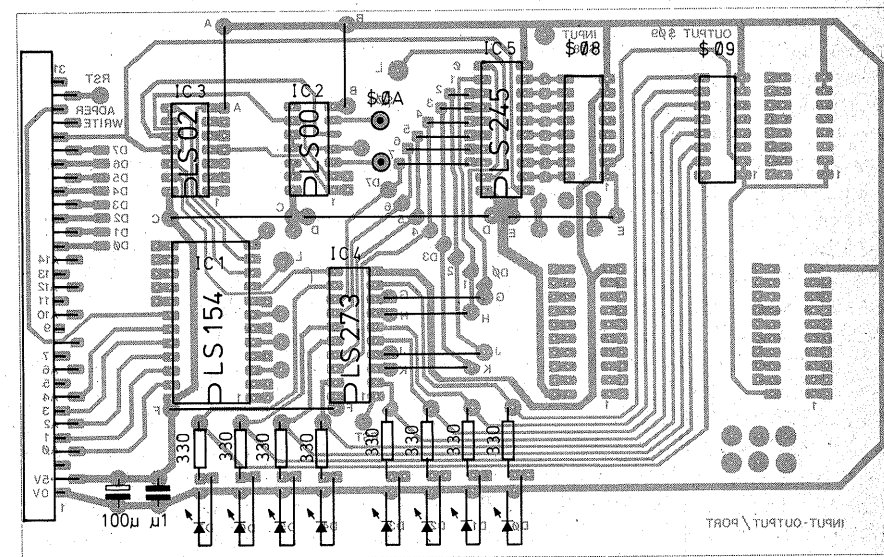


Bild 10.13 Bestückungsseite der Porteinheit

her bedienen lassen. Die Fassungen müssen fest auf der Leiterplatte aufsitzen. Die Stecker lassen sich nur schwer in die Fassungen drücken, und wenn sich der dazu nötige starke Druck auf die Leiterbahnen überträgt, reißen diese ab.

Die Anzeige-LEDs löten Sie so ein, daß sie auf der Leiterplatte **liegen**. Das Abwinkeln der Anschlußdrähte gelingt Ihnen am besten, wenn Sie sich aus einem Streifen Experimentierplatte im Rastermaß 2,5 mm eine Biegelehre herstellen (Bild 10.14). Biegen Sie zuerst den Kathodenanschluß ab, danach den Anodenanschluß, und zwar 2,5 mm weiter vom Gehäuse entfernt.

Falls Sie die acht „Ableitwiderstände“ zum Transceivereingang einlöten wollen, so löten Sie sie senkrecht auf die Platine und kürzen die freistehenden Drahtenden auf ca. 1 mm. Dann löten Sie den Massedraht zuerst am Rand der Platine an, biegen ihn so zu recht, daß er die Drahtstummel der Widerstände berührt, und löten ihn dann an die Widerstände (Bild 10.15).

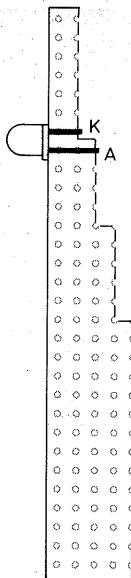


Bild 10.14 Abbiegen der LED-Anschlüsse mit der selbstgefertigten Biegelehre

Bild 10.15 Montage der Widerstände am Bustransceiver

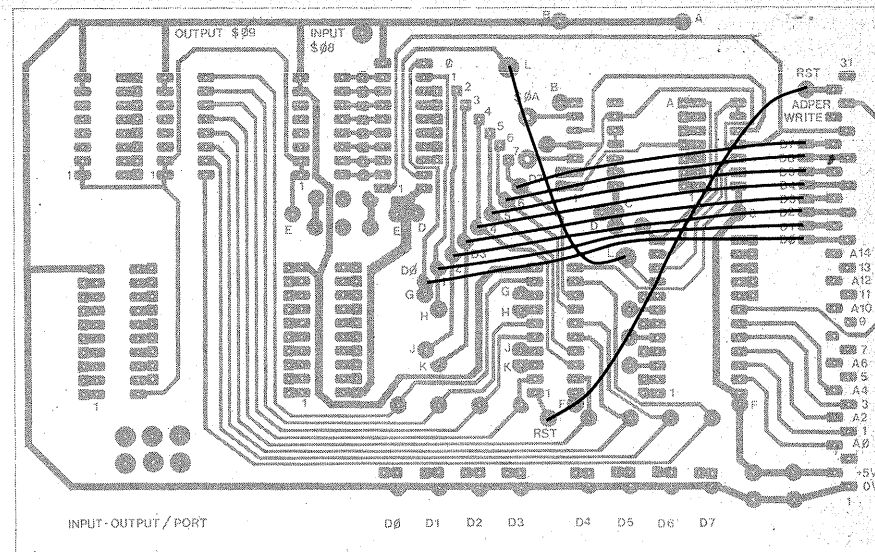
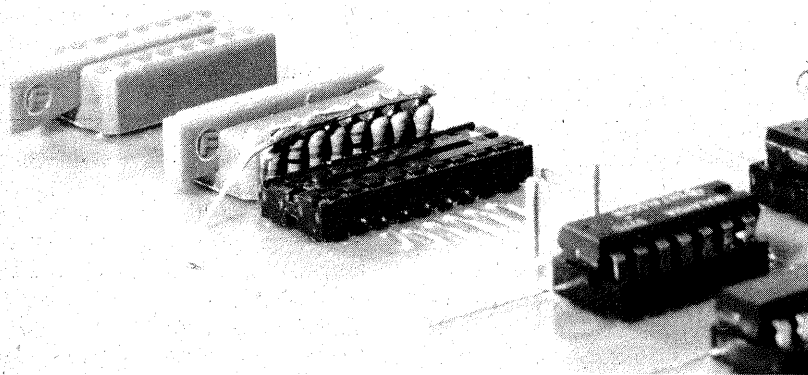


Bild 10.16 Drahtbrücken auf der Leiterbahnseite der Platine

Zum Schluß löten Sie die Drahtbrücken auf die Leiterbahnseite an (Bild 10.16).

Bild 10.17 zeigt Ihnen den fertigen Aufbau auf der Platine, Bild 10.18 den auf einer Experimentierplatte.

Die DIL-Stecker verdienen Ihre besondere Aufmerksamkeit. Sie sollen ja von den Keilen der Fassungen herausgeschoben werden. Die meisten DIL-Stecker sind aber auf der Stiftseite hohl. Daher füllen Sie das Loch mit einem Stückchen eingeklebtem Platinenmaterial aus (Bild 10.19).

Ferner sind die DIL-Stecker so gebaut, daß sich Computerflachbandlitze einpressen läßt. Die inneren Enden der Stifte sind nämlich geschlitzt. Wenn die Litze in den Schlitz gepreßt wird, schneidet der Steckerstift den Isoliermantel der Litze auf und stellt einen sicheren Kontakt zur Litze her.

Die Flachbandlitze ist recht steif. Beweglicher wird Ihr Kabelbaum, wenn Sie einzelne Litzen (0,14 mm²) in die Steckeranschlüsse pressen.

Setzen Sie den Stecker in die eingelötete DIL-Fassung. Dann können seine Stifte beim Einpressen der Litzen nicht beschädigt werden.

Zum Einpressen fertigen Sie sich ein Werkzeug:

Sie benötigen ein Stückchen Dübels Holz als Griff, ca. mit 10 mm Ø und 50 mm Länge, sowie ein Stückchen Eisendraht oder einen Nagel mit etwa 2 mm Ø und 40 mm Länge. Bohren Sie das Dübels Holz an einem Ende axial mit etwa 2 mm Ø auf, und pressen Sie den Draht etwa 15 mm hinein. Dann sägen Sie in das freie Ende des Drahtes mit einer PUK-Säge einen Schlitz von etwa 3 mm Tiefe. Schrägen Sie den Schlitz innen an seiner Öffnung mit einer Schlüsselfeile ab (Bild 10.20). Dann feilen Sie quer zum Schlitz eine kleine Vertiefung. Wenn Sie nun das Werkzeug auf ei-

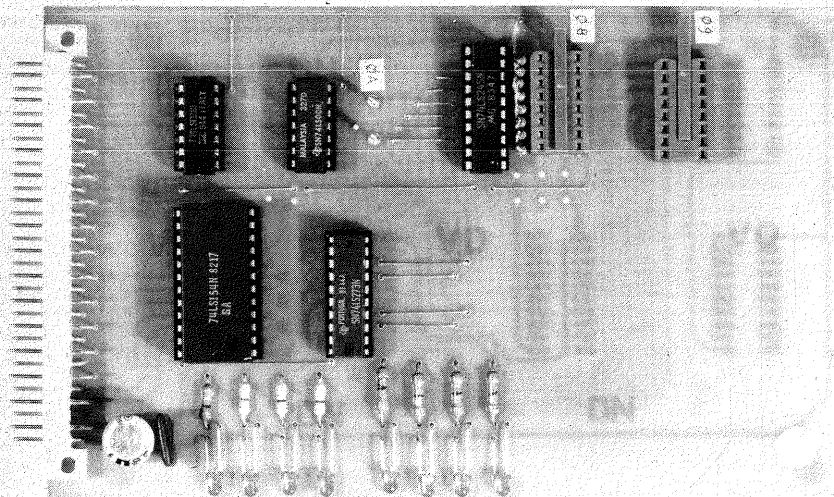


Bild 10.17 Die fertige Porteinheit auf der Platine

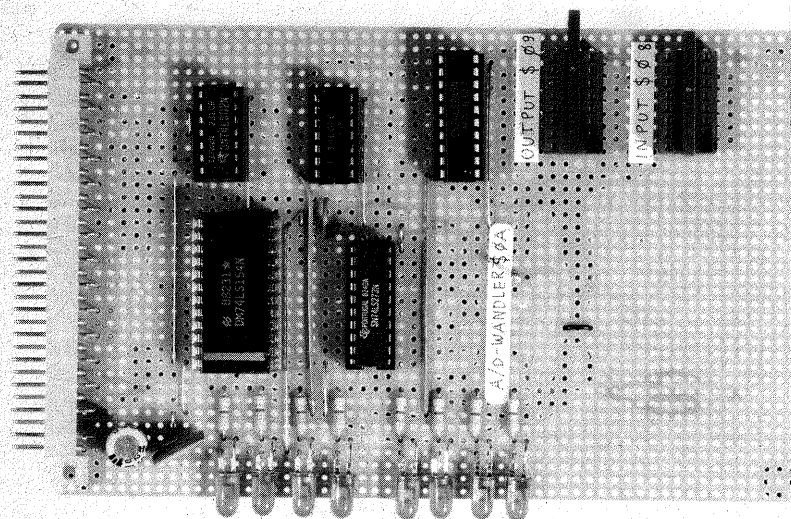
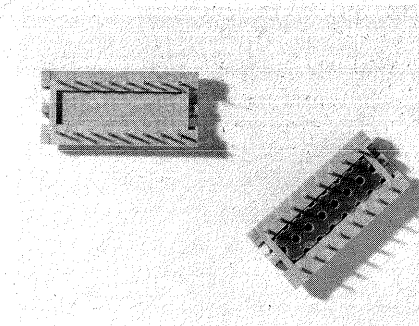


Bild 10.18 Die fertige Porteinheit auf einer Experimentierplatte



a) b)
Bild 10.19 Stecker DIS 16
a) ungefütert, b) gefüttert

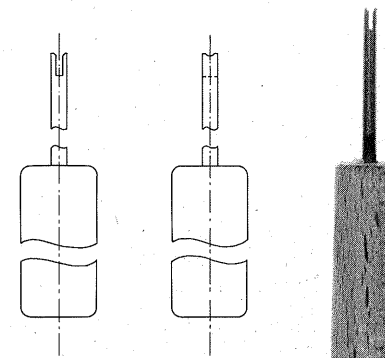


Bild 10.20 Werkzeug zum Einpressen der Litzen in die Steckeranschlüsse

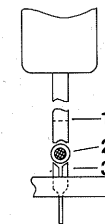


Bild 10.21 Einpressen der Litzen in die Steckeranschlüsse. 1. Einpreßwerkzeug, 2. isolierte Litze, 3. geschlitzter Steckeranschluß

nen Draht setzen, rutscht es nicht ab (Bild 10.21). Die Leitungen VCC und 0 V (GND) pressen Sie in je zwei Stifte (Bild 10.22), weil sie u. U. auch einmal einen stärkeren Strom übertragen sollen.

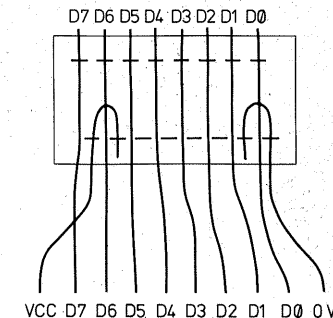


Bild 10.22 Belegung des Steckers DIS 16

Notieren Sie sich zu allen Leitungen, die Sie einpressen, Bitnummer und Farbcode, denn wenn Sie den Deckel des DIL-Steckers erst einmal aufgedrückt haben, ist er oft nur schwer zu entfernen, und Sie können nicht mehr einfach nachsehen, wohin eine Leitung gehört. Sie können sich auch mit Klebeband kleine Fahnen an die Leitungen heften, auf denen Sie die Bitnummern notieren. Achten Sie bei den Versorgungsleitungen auf die genormten Farben (VCC = rot, 0 V = blau).

Prüfung und Inbetriebnahme der Porteinheit

Stecken Sie die Porteinheit – zum Prüfen am besten über den Busadapter (vgl. Seite 90) – auf eine beliebige

Federleiste der Busplatte. Stecken Sie auch den Speicher auf und schieben den Schalter in Stellung „1“, so daß das RAM am Anfang des Adressierbereichs liegt.

Achtung: Der Port funktioniert nur in Verbindung mit dem Speicher!

Wenn das 2. Byte des Schreib-/Lesebefehls abgearbeitet wird, erscheint auf dem Datenbus das zu lesende oder zu schreibende Datenbyte. Wenn aber die Dateneingabe und -anzeige auf $\overline{\text{WRITE}}$ + Portadresse eingestellt ist, und das muß ja so sein, wenn man den Speicher nicht benutzt, so erzwingen die Schalterstellungen 1 Byte, welches der Port übernimmt (siehe Seite 102). In der Regel zeigt der Port dann nicht das gewünschte Datenbyte an, sondern die auf der Dateneingabe stehende Portadresse. Der Datenschalter Da muß also auf DA-FLOT stehen, dann kann der Lese- oder Schreibbefehl aber nur noch aus dem Speicher kommen.

Laden Sie folgendes kurzes Programm in den Speicher (siehe Seite 173):

Adresse			
00 04	LODI,R0	Lade Register R0	
01 55		mit 55 ₁₆	
02 D4	WRTE,R0	Schreibe (R0)	
03 09		in die Portadresse 09	
04 40	HALT		

Dann betätigen Sie RESET und lassen das Programm im 1-Hz-Takt laufen (S4 und HOLD auf RUN).

Wie Ihnen aus dem vorangegangenen Kapitel sicher schon vertraut ist, zählt der Programmzähler von 00 beginnend hoch. Wenn er bei 03 angekommen ist, wird er plötzlich die Portadresse 09 anzeigen, zugleich wird ADPER aktiv (L, die LED erlischt), und der Inhalt von R0 (55₁₆) erscheint in den Port-LEDs.

Wenn nicht, so gibt es nur wenige

Fehlermöglichkeiten

1. Messen Sie zuerst an allen ICs, ob zwischen ihren Anschlüssen GND und VCC wirklich 5 V Betriebsspannung stehen.

2. Sind alle LEDs richtig angeschlossen? (Kathode an der gemeinsamen Massebahn). Haben sie das Abbiegen ihrer Anschlüsse und das Einlöten überstanden? Ziehen Sie IC4 aus der Fassung, und tippen Sie die den LEDs abgewandten Enden der 330-Ω-Widerstände über ein Prüfkabel mit VCC an, die jeweilige LED muß aufleuchten. Wenn der Test positiv verlaufen ist, stecken Sie IC4 wieder in die Fassung.

3. Erscheint das Steuersignal an CP des 74LS273 (Pin 11)? Schalten Sie S1 von OSZ auf EINZEL, drücken Sie RESET, und takten Sie mit STEP mindestens dreimal, danach takten Sie das Programm durch. Wenn der Programmzähler wieder bei 03 angekommen ist, wird die Adreßanzeige wieder auf die Portadresse 09 springen. Dann hören Sie auf zu takten und lassen den Prozessor ruhen (eine Möglichkeit, die Ihnen die anderen marktgängigen Prozessoren nicht bieten können!). In diesem Zustand muß der Ausgang 9 von IC1 (Pin 10) L sein, Pin 13 von IC3 = Pin 11 von IC4 (CP) müssen dagegen H sein.

Wenn der Decoder an 9 nicht L hat, prüfen Sie, ob $\overline{\text{ADPER}}$ (= L) an $\overline{\text{E0}}$ (Pin 18) sowie die Adressen richtig an den Anschlüssen 19 (A4 = $\overline{\text{E1}}$ = L), 20 (A3 = H), 21 (A2 = L), 22 (A1 = L) und 23 (A0 = H) ankommen.

4. Wenn das Steuersignal an CP richtig ankommt, prüfen Sie, ob $\overline{\text{MR}}$ (Pin 1 von IC4) wirklich H ist, denn ein L setzt das Register sofort zurück, und es erscheint keine Anzeige.

5. Wenn die Signale $\overline{\text{MR}} = \text{H}$ und $\text{CP} = \text{H}$ richtig vorhanden sind, wird ein Fehler wahrscheinlich nur noch in den Datenleitungen vorhanden sein. Dann hilft nur noch die Suche nach Kurzschlüssen, kalten Lötstellen und Leiterbahnunterbrechungen der Datenleitungen. Vergessen Sie nicht, auch die IC-Fassungen in Ihre Prüfungen einzubeziehen. Manchmal liegen die Fehlerursachen ganz offenbar, und man sieht sie trotzdem nicht gleich. Z. B. sollte eine Porteinheit mit vier Schrauben festmontiert werden. Bei einem Schraubenloch war die Massebahn, die am Rand der Platine verläuft, durchtrennt.

Wenn der Port richtig anzeigt, wiederholen Sie das Programm mit der Clockfrequenz 1 MHz. Nun können Sie das Addierprogramm von Seite 172 und alle übrigen Programme wiederholen, nur müssen Sie statt der Befehle WRTE (F0) den erweiterten Schreibbefehl WRTE 09 einsetzen, also je nach Register D4, D5, D6 oder D7 (siehe Seite 173). Dann wird Ihnen auch bei der schnellen Clockfrequenz kein Arbeitsergebnis mehr entgehen.

In diesem Stadium ist der Computer als Computer „fertig“. Sie können mit ihm – in seiner Sprache – in einen Dialog eintreten, er wird für Sie in seiner Sprache arbeiten und Ihnen antworten. Sie können alle Befehle benutzen. Alles, was jetzt noch folgt, ist „nur“ noch Bedienungskomfort, freilich bringt der wieder neue Möglichkeiten mit sich, wie z. B. der Analog-Digital-Wandler zeigt.

Kapitel 11

Der Analog-Digital-Wandler (Baugruppe 8)

Mikrocomputer stellen sich – besonders in Kaufhäusern – oft so dar, als seien Tastatur und Bildschirm die einzigen Verbindungen zur Umwelt; dazu kommt dann vielleicht noch ein Drucker oder ein Kassetteninterface, als habe die Computerei es ausschließlich mit Schriftzeichen zu tun. Dies ist eine Verengung des Blicks, denn der eigentliche Anwendungsbereich des Mikrocomputers kann dabei gar nicht sichtbar werden. Mikrocomputer sind in der industriellen Anwendung meist in Geräten versteckt. Sie dienen der Steuerung und Regelung in Geräten und Maschinen der verschiedensten Art. Sie brauchen keine Eingabetastatur, denn sie enthalten ihr Arbeitsprogramm in einem ROM.

Die Daten, die sie erfassen, fallen aber meist in analoger Form an. Wenn z.B. Durchflußmengen, Drehwinkel, Druck, Temperaturen, Licht, die Stärke von Magnetfeldern usw. durch Sensoren (lat. Fühler) in elektrische Signale umgewandelt werden, entstehen Spannungen, die sich analog mit den erfaßten Vorgängen ändern. Mit den sich ändernden Spannungen allein kann der Computer

nichts anfangen, sie müssen erst in seine Sprache, in Bitmuster, umgewandelt werden. Dazu dient der Analog-Digital-Wandler (AD-Wandler). Er erzeugt ein zu einer bestimmten Spannung gehörendes Bitmuster. Der Computer kann das Bitmuster verarbeiten, im einfachsten Fall anzeigen. Mit Hilfe des AD-Wandlers wird er also zu einem Digitalvoltmeter.

Schaltungstechnik

Digital-Analog-Umwandlung mit einem $R/2R$ -Widerstandsnetzwerk

Es gibt verschiedene Prinzipien der AD-Wandlung. Vergleichsweise leicht zu durchschauen ist die Wandlung mit einem variablen Spannungsteiler (Bild 11.1). Der Spannungsteiler wird mit einer konstant gehaltenen Bezugsspannung („Referenzspannung“ U_{REF}) gespeist. Die Ausgangsspannung U_{aus} ergibt sich aus dem Verhältnis des „oberen“ Widerstandes R_o zum „unteren“ Widerstand R_u :

$$U_{aus} = \frac{R_u}{R_u + R_o} \cdot U_{REF}$$

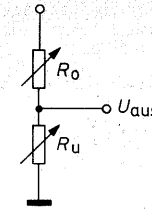


Bild 11.1 Erzeugung einer veränderlichen Ausgangsspannung mit einem variablen Spannungsteiler

Der absolute Wert der Widerstände ist nicht wichtig, sondern nur ihr Verhältnis zueinander. Die obige Formel gilt allerdings nur für den unbelasteten Spannungsteiler. Wenn aber die Teilwiderstände nicht sehr hochohmig sind und an den Ausgang ein Schaltglied mit einem sehr hohen Eingangswiderstand, z.B. ein FET oder ein Operationsverstärker, angeschlossen wird, so ist der Fehler vernachlässigbar gering, so daß die obige einfache Formel gilt.

Es ist vergleichsweise schwer, bei der Herstellung integrierter Schaltungen den absoluten Wert eines Widerstandes genau zu erreichen. Es ist dagegen leicht zu erreichen, daß Widerstände, die untereinander gleich sein sollen (wobei es auf den absoluten Wert weniger ankommt), auch wirklich nahezu gleich sind. Es liegt daher nahe, den Spannungsteiler aus einer Reihe untereinander gleicher Widerstände umschaltbar zu machen.

Ein sehr ausgeklügeltes System ist das $R/2R$ -Widerstandsnetzwerk (Bild 11.2). Es besteht nur aus Widerständen mit dem „einfachen“ Wert R und dem zweifachen Wert $2R$ ($R + R$). Die umschaltbaren Widerstände liegen entweder an U_{REF} oder an $0V$, einen offenen (undefinierten) Zustand gibt es nicht. Die Schalter tragen entsprechend ihrer Wertigkeit im Byte

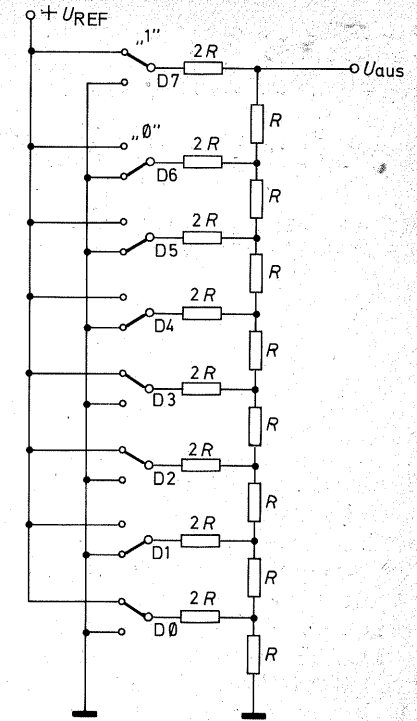


Bild 11.2 $R/2R$ -Netzwerk für acht Dualstellen

die Bezeichnungen D0 bis D7. Je nach Schalterstellungen ergeben sich unterschiedliche Spannungsteilverhältnisse. Sie sind für 8 Bit schwer zu überschauen. Daher ist die Schaltung noch einmal für nur 2 Bit gezeichnet (Bild 11.3).

Zur besseren Übersicht haben die vier Widerstände Namen bekommen (A, B, C und D). Die Schalter tragen entsprechend ihrer Wertigkeit in einer Dualzahl die Bezeichnungen D0 und D1. Wenn ein Schalter einen Widerstand an $0V$ legt, hat er den logischen Wert „0“, wenn er einen Widerstand an U_{REF} schaltet, den logischen Wert „1“.

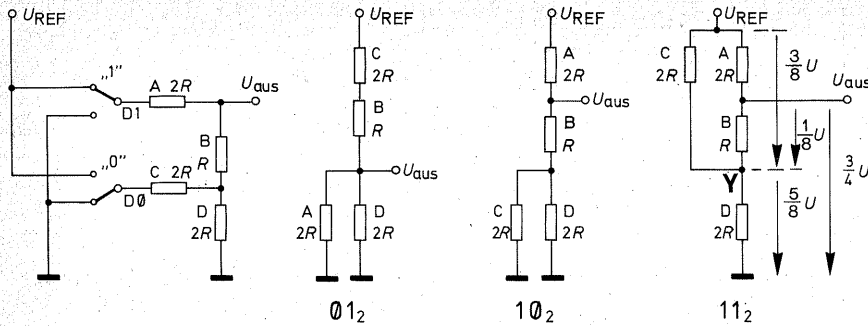


Bild 11.3 R-/2 R-Netzwerk für 2 Bit, D0 und D1

Für die Schalterstellung 00 muß sich die Ausgangsspannung 0 V ergeben, weil alle Widerstände an 0 V geschaltet sind und kein einziger an U_{REF} . Bei der Schalterstellung 01 liegen die Widerstände C und B in Reihe und bilden im Spannungsteiler den Wert für R_o mit 3 R. A und D, mit je 2 R parallelgeschaltet, bilden R_u mit 1 R. U_{REF} wird dadurch im Verhältnis 3:1 geteilt, U_{aus} hat somit den Wert von $\frac{1}{4} U_{REF}$:

$$U_{aus} = \frac{R_u}{R_u + R_o} \cdot U_{REF} = \frac{1}{1+3} \cdot U_{REF} = \frac{1}{4} U_{REF}$$

In der Schalterstellung 10 bildet A mit 2 R den Wert R_o . B mit 1 R und die Parallelschaltung von C und D mit resultierendem 1 R liegen in Reihe und bilden zusammen auch 2 R. U_{REF} wird daher im Verhältnis 1:1 geteilt. U_{aus} hat so den Wert $\frac{1}{2} U_{REF}$:

$$U_{aus} = \frac{R_u}{R_u + R_o} \cdot U_{REF} = \frac{2}{2+2} \cdot U_{REF} = \frac{1}{2} U_{REF}$$

In der Schalterstellung 11 liegen die

Verhältnisse schon etwas komplizierter:

Betrachten wir zunächst den Spannungsteiler mit dem Knotenpunkt Y: R_o besteht aus der Parallelschaltung von C (2 R) und A+B (zusammen 3 R). Der daraus resultierende Widerstand beträgt

$$R_o = \frac{2 \cdot 3}{2+3} R = \frac{6}{5} R$$

R_u besteht nur aus D (2 R). Also steht an Y eine Spannung von $\frac{5}{8} U_{REF}$:

$$U_Y = \frac{R_u}{R_u + R_o} \cdot U_{REF} = \frac{2}{2 + \frac{6}{5}} \cdot U_{REF} = \frac{5}{8} U_{REF}$$

Zwischen U_{REF} und Y bleibt damit eine Spannung von $\frac{3}{8} U_{REF}$. Sie wird durch A und B im Verhältnis 2:1 geteilt, so daß bei U_{aus} auf den $\frac{3}{8} U_{REF}$ von Y noch zusätzliche $\frac{1}{8} U_{REF}$ stehen. Also beträgt $U_{aus} \frac{3}{4} U_{REF}$. Wenn man die Schalter im Dualsystem von 00 beginnend hochzählt und entsprechend der Reihe der Dualzahlen schaltet, entsteht am Ausgang eine **Treppenspannung**, die den Dualzahlen entspricht. In Bild 11.4 ist sie für einen Dualzähler mit zwei Stellen gezeichnet, der immer wiederkehrend von 00 bis 11₂ zählt. Entsprechend den vier Möglichkeiten entstehen die

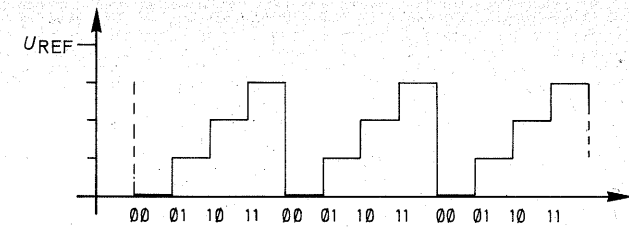


Bild 11.4 Erzeugung einer Treppenspannung durch ein R-/2 R-Netzwerk mit zwei Dualstellen

vier Spannungen $0 \cdot U_{REF}$, $\frac{1}{4} \cdot U_{REF}$, $\frac{1}{2} \cdot U_{REF}$ und $\frac{3}{4} \cdot U_{REF}$. Was hier der Einfachheit halber an nur 2 Bit vorgeführt wurde, gilt auch für ein umfangreicheres Netzwerk. Allgemein entspricht U_{aus} der Dualzahl in den Schalterstellungen:

$$U_{aus} = \frac{\text{Dualzahl}}{\text{Anzahl der Möglichkeiten}} \cdot U_{REF}$$

Bei 2 Bit (vier Möglichkeiten) ist die Treppe recht grob. Bei 8 Bit gibt es aber immerhin 256 Möglichkeiten, und das ist eine schon recht feine Abstufung. In Bild 11.2 wurde die Dualzahl 1000 1101 $\hat{=}$ 141₁₀ eingestellt. Die Ausgangsspannung beträgt daher

$$\frac{141}{256} \cdot U_{REF}$$

Wenn U_{REF} 2,56 V beträgt, so steht am Ausgang eine Spannung von 1,41 V.

In einem Netzwerk aus integrierten Widerständen, die untereinander gleich sind, darf man mit hoher Präzision rechnen.

Die Schalter können elektronische Schalter sein. Speziell für diesen Zweck entwickelte Transistoren schalten bis auf ca. 1 mV durch, „normale“ Transistoren haben dagegen eine Kollektor-Emitter-Sättigungsspannung von ca. 0,2 V.

Analog-Digital-Wandlung mit einem R-/2 R-Widerstandsnetzwerk, einem Dualzähler und einem Spannungs-komparator

Bisher wurde dargestellt, wie man zu einer Dualzahl die ihr entsprechende Spannung erzeugt. Damit steht ein Digital-Analog-Wandler zur Verfügung. Auch er wird oft benötigt. Aber wie macht man daraus einen Analog-Digital-Wandler, der zu einer vorhandenen Spannung die entsprechende Dualzahl erzeugt?

Dazu benötigt man einen Dualzähler mit so vielen Stellen, wie das R-/2 R-Widerstandsnetzwerk auch hat – zum Zähler gehört ein Taktgenerator. Außerdem benötigt man noch einen Spannungsvergleichler (lat. Spannungsvergleichler). Bild 11.5 enthält das Schaltungsprinzip:

Auf den nichtinvertierenden Eingang des als Spannungs-komparator geschalteten Operationsverstärkers wird die zu wandelnde Eingangsspannung U_{ein} gegeben. Der Taktgenerator veranlaßt den Zähler, von 00 beginnend hochzuzählen. Damit steigt die Ausgangsspannung U_{aus} des Widerstandsnetzwerks ebenfalls von 0 V beginnend an. Sie wird auf den invertierenden Eingang des Spannungs-komparators geleitet. Solange U_{aus} kleiner ist als U_{ein} , ist der Ausgang des Operationsverstärkers H; sobald jedoch die erste Treppenstufe über U_{ein} erreicht ist, springt der Ausgang des

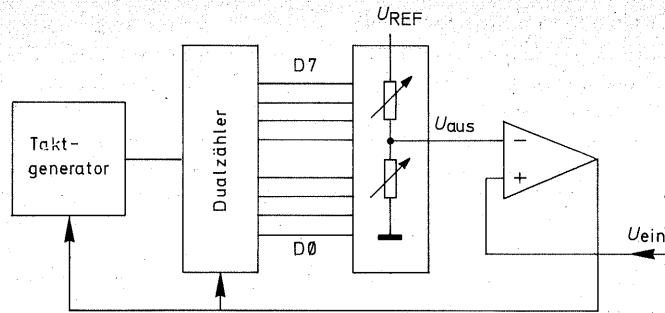


Bild 11.5 Blockschaltbild der zur AD-Wandlung mit einem R-/2 R-Netzwerk erforderlichen Funktionen: Dualzähler mit Taktgenerator, Widerstandsnetzwerk und Spannungskomparator

Operationsverstärkers von H auf $-U$, eine Diode am Ausgang schließt aber die negative Spannung kurz, so daß der Ausgang in Wirklichkeit nur L ist. Dieser Spannungssprung kann dazu benutzt werden, den Zähler anzuhalten. Dessen Ausgänge zeigen dann das Bitmuster, dessen zugehörige Spannung gerade eben über der Eingangsspannung U_{ein} liegt. Das Bitmuster kann vom Computer gelesen werden.

Ein DA/AD-Wandler nach diesem Prinzip ist der Baustein ZN 425 E von der Firma Ferranti (Bild 11.6). Er enthält ein 8gliedriges R-/2 R-Netzwerk,

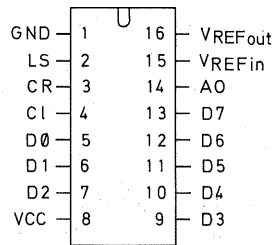


Bild 11.6 Anschlußbelegung des ZN 425 E (Ferranti)

dazu einen 8-Bit-Dualzähler mit den elektronischen UM-Schaltern, eine Referenzspannungsquelle für 2,56 V (sehr gut passend zum höchstmöglichen Zählerstand von 255_{10} !) sowie einen UM-Schalter, mit dem man bestimmen kann, ob der Baustein als DA- oder AD-Wandler arbeiten soll.

Die Anschlüsse bedeuten im einzelnen:
LS

(logic select, engl. Auswahl der Arbeitsweise, Pin 2). Mit LS=H arbeitet der Baustein als AD-Wandler, dann sind die Anschlüsse C0 bis C7 die Ausgänge des Dualzählers (C von counter, engl. Zähler).

Mit LS=L arbeitet der Baustein als DA-Wandler. Dann sind die Anschlüsse C0 bis C7 Eingänge, mit denen die einzelnen umschaltbaren Widerstände des Netzwerkes an 0 V oder U_{REF} geschaltet werden können. Ein an sie angelegtes Datenwort erzeugt am Ausgang des Widerstandsnetzwerkes eine entsprechende Ausgangsspannung.

CI (clock, Pin 4) ist der Takteingang für den Zähler.
 \overline{CR} (counter reset, Pin 3) ist der

Rückstelleingang für den Zähler. Mit $\overline{CR}=L$ wird der Zähler auf 00 gestellt.

AO (analogue output, analoge Ausgangsspannung, Pin 14) ist der Knotenpunkt des Spannungsteilers, an dem die Treppenspannung entsteht.

VREFin (reference voltage input, engl. Eingang für die Bezugsspannung, Pin 15) ist der Eingang für die Bezugsspannung, mit der das Widerstandsnetzwerk versorgt wird.

VREFout (reference voltage output, Pin 16) ist der Ausgang der im Baustein enthaltenen Bezugsspannungsquelle von 2,56 V. Am einfachsten ist es,

wenn man diese Bezugsspannung auf den Eingang V_{REFin} schaltet. Man kann aber auch eine andere Bezugsspannung wählen, mit der man V_{REFin} versorgt.

Bild 11.7 zeigt die eigentliche Wandlerschaltung der Baugruppe. Sie entspricht dem Blockschaltbild 11.5. Es kommt aber noch eine UM-Schaltlogik dazu (N3, N4, N5).

N3 und N4 bilden ein RS-Flip-Flop (siehe Seite 54). Wenn der Impuls Convert command \overline{CC} (aktiv L) eintrifft, wird der Zähler auf 00 gesetzt (siehe \overline{CR}) und gesperrt, solange \overline{CC} L ist. Zugleich wird das Flip-Flop gesetzt. Der Ausgang von N3 wird H und öffnet das Tor N5. Taktpulse können N5 so lange passieren, wie der Ausgang von N3 H ist. Sobald

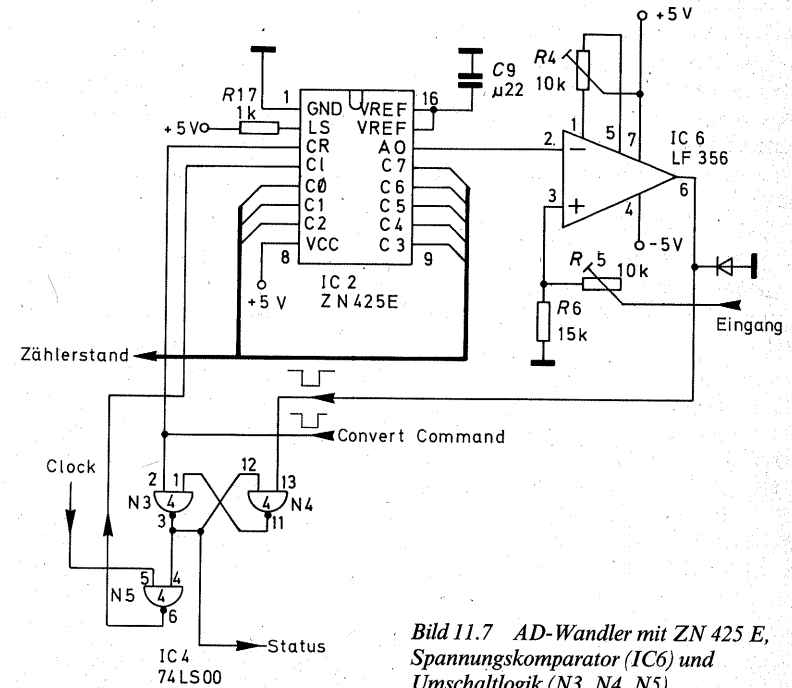


Bild 11.7 AD-Wandler mit ZN 425 E, Spannungskomparator (IC6) und Umschaltlogik (N3, N4, N5)

\overline{CC} wieder H geworden ist, beginnt der Zähler von 00 an zu zählen, und an AO steigt die Treppenspannung. Da sie mit 0V beginnt, ist der Ausgang von IC6 zuerst immer H, das Flip-Flop bleibt also gesetzt. Sobald die Treppenspannung aber eine Stufe über der Eingangsspannung erreicht hat, kippt der Ausgang von IC6 in Richtung -U, wegen der Diode aber nur auf L, so daß keine negative Spannung auf den Eingang von N4 gerät. Durch das Kippen von IC6 wird das Flip-Flop rückgesetzt. Der Ausgang von N3 wird wieder L, damit wird das Tor N5 für weitere Taktpulse geschlossen, und der Zähler bleibt stehen.

An der Leitung „Status“ (lat./engl. Zustand), die den Ausgangspegel von N3 wiedergibt, läßt sich abfragen, ob sich der Wandler gerade in einem Zählzyklus befindet (Status = H) oder ihn beendet hat und das Ergebnis an den Zählerausgängen ansteht (Status = L).

Jeder weitere Umwandlungsvorgang muß von einem Impuls \overline{CC} eingeleitet werden. Zugleich muß ein schnelles Taktsignal vorhanden sein. Beide Signale können aus der CPU abgeleitet werden. Das Taktsignal könnte man dem Taktgenerator der CPU entnehmen, den \overline{CC} -Impuls durch einen Befehl erzeugen. Man müßte dann einen Port aufbauen (ein 74LS25 als Adreßdecoder würde reichen), und durch einen WRTE-Befehl wäre der \overline{CC} -Impuls zu schreiben. Das hätte aber zur Folge, daß jeder Wandlerzyklus zu programmieren wäre.

Es ist daher bei geringem Mehraufwand günstiger, Taktpulse und \overline{CC} in der Baueinheit freilaufend zu erzeugen.

Der CLOCK-Oszillator für den Zähler

Als CLOCK-Generator genügt ein einfacher RC-Oszillator aus zwei Schmitt-Triggern (Bild 11.8), wie er bereits dargestellt wurde (siehe Seite 125). Diesmal wurden zwei NAND-Gatter mit Schmitt-Trigger-Eingängen verwendet. Von den vier Gattern des 74LS132 bleiben zwei ungenutzt, die aber dazu verwendet werden können, gegebenenfalls eine andere Portadresse zu decodieren. Würde wieder der 74LS14 eingesetzt, so blieben vier Inverter ungenutzt, mit denen im Falle eines Falles keine logischen Verknüpfungen möglich wären.

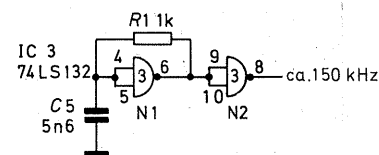


Bild 11.8 Der Taktgenerator für den AD-Wandler

Der CLOCK-Oszillator schwingt mit einer Frequenz von ca. 150 kHz, eine Periode dauert etwa 6,6 μ s. Ein Zählerdurchgang mit 256 Pulsen dauert damit etwa 1,7 ms. Die Zeit ist freilich entsprechend kürzer, wenn der Zähler vor 255₁₀ stehen bleibt. Zwischen zwei Zählungen vergehen die Pulsbreiten der beiden Mono-Flops (siehe unten) mit zusammen etwa 0,131 ms, so daß eine ganze Meßperiode im längsten Fall nicht einmal 2 ms dauert.

Die Erzeugung des Umwandlungsimpulses \overline{CC} und des Speicherübernahme-Impulses E

Der \overline{CC} -Impuls und der Speicherübernahmeimpuls E (siehe Seite 199) werden durch zwei Mono-Flops aus dem Statussignal erzeugt. Ein Mono-Flop ist ein Zeitgeberbaustein. Bild 11.9 zeigt die Grundsaltungen mit zwei NAND- (Bild 11.9a) oder zwei NOR-Gattern (Bild 11.9b) und jeweils einem RC-Glied.

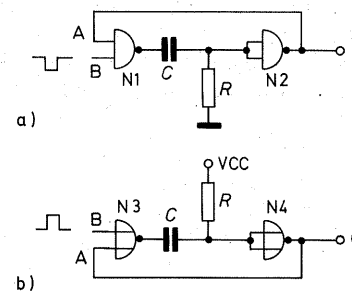


Bild 11.9 Grundsaltung des Mono-Flops a) mit NAND-Gattern, b) mit NOR-Gattern

Wir betrachten zunächst das Mono-Flop aus NAND-Gattern: Der Ausgang von N2 ist im Ruhezustand H, weil die Eingänge über $R=L$ sind. Das H von N2 wird auf den Eingang A von N1 übertragen. Damit ist der Ausgang von N1 so lange L, wie der Eingang B = H ist. Wenn der Eingang B von N1 ein L-Signal empfängt, springt der Ausgang von N1 auf H. C lädt sich über den Ausgang von N1 und R auf. Dabei entsteht durch den Ladestrom an R ein Spannungsabfall, der als positiver Spannungssprung am Eingang von N2 wirksam wird. Damit springt der Ausgang von N2 auf L. Dieses L wird auf den Eingang A von N1 übertragen,

weswegen der Ausgang von N1 H bleibt, auch wenn der L-Impuls am Eingang B wieder verschwunden ist. Sobald sich C aufgeladen hat, fällt an R keine Spannung mehr ab. Die Eingänge von N2 sind wieder L, der Ausgang von N2 springt auf H. Damit ist der Ausgangszustand wieder erreicht. C entlädt sich nun über den Ausgang von N1 und R. An R entsteht nun, durch den Entladestrom, ein negativer Spannungsabfall, der aber nicht als Schaltsignal wirksam wird.

Das Mono-Flop hat einen stabilen Zustand. Es kann durch einen Impuls für eine Dauer, die durch die Zeitkonstante des RC-Gliedes bestimmt wird, aus dem Ruhezustand ausgelenkt werden. Es kehrt danach von allein in den Ruhezustand zurück.

Man kann das Mono-Flop auch aus NOR-Gattern aufbauen (Bild 11.9b). Dann spielt sich das gleiche nur mit umgekehrten Pegeln ab.

Mono-Flops gibt es als fertige Bausteine, die über die Grundsaltung hinaus meist noch zusätzlichen Komfort enthalten, um die Anwendung flexibler gestalten zu können. Ein sehr vielfältig einsetzbarer Baustein ist der 74LS123. Er enthält zwei Mono-Flops (Bild 11.10). Die Funktionstafel (Bild 11.11) gibt ihr Verhalten an:

Q und \overline{Q} sind die Ausgänge. Sie verhalten sich einander entgegengesetzt – ebenso wie die Ausgänge eines Flip-Flops. Es können damit je nach Wunsch für die Auslenkzeiten H- oder L-Signale entnommen werden. R_D ist ein Rücksetzeingang (aktiv L). Damit kann das Mono-Flop in die Ruhelage rückgesetzt werden, auch wenn die Auslenkzeit noch nicht abgelaufen ist.

A und B sind die Setzeingänge. Sie sind nur wirksam, wenn $R_D = H$ ist.

\bar{A} setzt das Mono-Flop mit einem Spannungssprung von H nach L, aber nur, wenn währenddessen $B=H$ ist. Der Ruhezustand für A ist H.

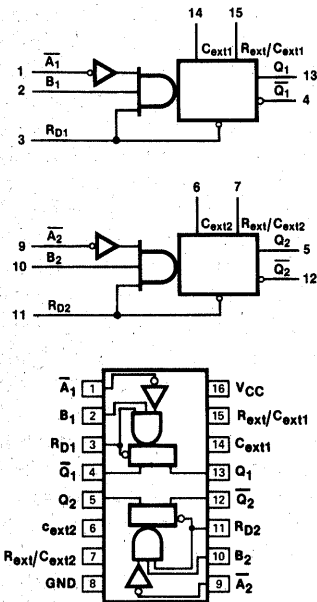


Bild 11.10 Logikplan der Mono-Flops im 74LS123 (nach VALVO-Unterlagen)

FUNCTION TABLE

INPUTS			OUTPUTS	
R_D	\bar{A}	B	Q	\bar{Q}
L	X	X	L	H
X	H	X	L	H
X	X	L	L	H
H	L	I	[Puls]	[Puls]
H	I	H	[Puls]	[Puls]
I	L	H	[Puls]	[Puls]

H = HIGH voltage level
 L = LOW voltage level
 X = Don't care
 I = LOW-to-HIGH transition
 ↓ = HIGH-to-LOW transition
 [Puls] = One HIGH-level pulse
 [Puls] = One LOW-level pulse

Bild 11.11 Funktionstafel des 74LS123 (nach VALVO-Unterlagen)

B setzt das Mono-Flop mit einem Spannungssprung von L nach H, aber nur, wenn währenddessen $\bar{A}=L$ ist. Der Ruhezustand für B ist L.

Und noch eine Eigenschaft ist wichtig. Mit dem Einschalten der Betriebsspannung entsteht an den Ausgängen ein Impuls Q bzw. \bar{Q} . Danach kippt das Mono-Flop in seine Ruhelage, aus der es dann durch Impulse an A oder B ausgelenkt werden kann. Diese Eigenschaft wird zum Starten der Schaltung ausgenutzt.

Die Pulsbreite t_w (w von width, engl. Breite) hängt vom RC-Glied ab, mit dem das Mono-Flop beschaltet wird. Der Widerstand liegt zwischen VCC und dem Anschluß R_{ext}/C_{ext} (Pin 15 beim Mono-Flop 1, Pin 7 beim Mono-Flop 2). Der zeitbestimmende Kondensator wird an die Anschlüsse C_{ext} und R_{ext}/C_{ext} (Pin 14/15 bzw. 6/7) angeschlossen (siehe Beschaltung in Bild 11.12). Bei Verwendung eines Elektrolytkondensators muß der Pluspol an R_{ext}/C_{ext} , der Minuspol an C_{ext} liegen.

Die Pulsbreite t_w ist nach der Formel

$$t_w = 0,28 \cdot R_{ext} \cdot C_{ext} \cdot (1 + \frac{0,7}{R_{ext}})$$

zu berechnen. Für das Mono-Flop 1 in Bild 11.12 beträgt t_w danach etwa 2,8 μ s, für das Mono-Flop 2 etwa 131 μ s.

Der \overline{CC} -Impuls wird von \bar{Q}_2 geliefert. In der Ruhelage sind \bar{Q}_2 und $\bar{Q}_1 = H$. Mit dem Einschalten der Betriebsspannung fallen diese Ausgänge auf L, und damit wird der Zähler auf 00 gesetzt, und sobald \bar{Q}_2 wieder H ist, startet der Zähler.

Während des Zählens ist die Leitung „Status“ H. Sobald der Spannungs-komparator das Flip-Flop N3/N4 rücksetzt, fällt das Statussignal von H auf L. Damit setzt es das Mono-

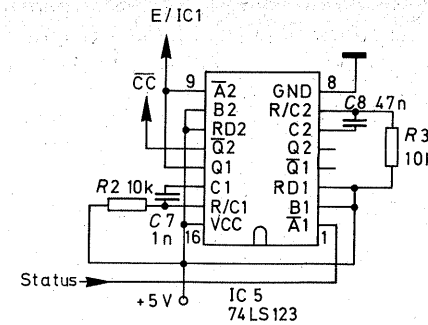


Bild 11.12 Erzeugung der Steuerimpulse \overline{CC} und E durch zwei Mono-Flops

Flop 1 (\bar{A}_1). Am Ausgang Q_1 entsteht ein H-Impuls, welcher auf den Enable-Eingang E des Registers (IC 1) geleitet wird und es veranlaßt, den Zählerstand zu speichern.

Wenn Q_1 wieder auf L fällt, wird durch den Spannungssprung von H auf L das Mono-Flop 2 gesetzt. Dadurch wechselt \bar{Q}_2 von H nach L, erzeugt also den \overline{CC} -Impuls und startet einen neuen Zählerzyklus.

Somit läuft der AD-Wandler in einem sich stetig wiederholenden Zyklus „Starten – Zählen – Speichern“.

Die Portschtaltung für den AD-Wandler

Die Zählerausgänge dürfen nicht unmittelbar auf den Datenbus geleitet werden, weil es keine Tri-State-Ausgänge sind. Außerdem geht das Zählerergebnis beim erneuten Start des Zählers verloren – der Zähler wird durch \overline{CC} ja auf 00 rückgesetzt. Das Zählerergebnis steht nur für die Pulsbreite des Mono-Flop 1 (etwa 2,8 μ s) an den Zählerausgängen, dann wird der Zähler durch das Mono-Flop 2 erneut gestartet. Da die CPU und der AD-Wandler asynchron laufen, wäre es reiner Zufall, wenn die CPU den AD-Wandler gerade dann läse, wenn das Ergebnis an den Zählerausgängen steht.

Daher ist das Ergebnis in einem Register zu speichern und dann über Tri-State-Puffer auf den Datenbus zu geben. Beides zusammen ergibt einen Port mit Eingangsspeicher.

Der Baustein 74LS373 enthält acht transparente Latches mit Tri-State-Ausgängen, also Puffern, die zwischen den eigentlichen Latchausgang und den IC-Ausgang geschaltet sind (Bild 11.13).

Bemerkenswert ist die Anwendung der doppelten Negation. Im 74LS373

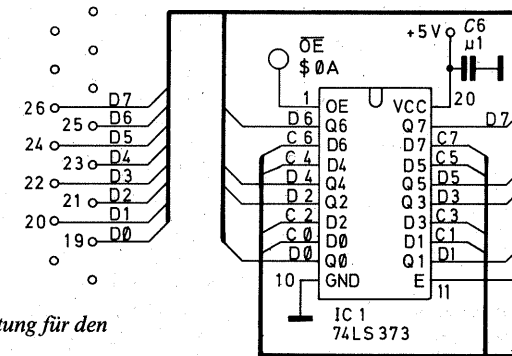


Bild 11.13 Portschtaltung für den AD-Wandler

invertieren die Puffer – daher sind sie nicht an die Q-Ausgänge, sondern an die Q-Ausgänge der Latches angeschlossen.

Die Latches übernehmen mit dem H-Impuls des Mono-Flop 1 den Zählerstand. Die Daten erscheinen aber erst dann auf dem Datenbus, wenn die Ausgänge durch ein \overline{OE} -Signal (aktiv L) geöffnet werden. Zum Lesen ist die Adresse $0A_{16}$ vorgesehen. Diese Adresse ist auf der Porteinheit decodiert. Zum Lesen des AD-Wandlers muß daher eine Verbindung zwischen den Lötträgeln $0A$ auf der Porteinheit und \overline{OE} auf dem AD-Wandler hergestellt werden.

Statt des Registers 74LS373 mit den transparenten Latches kann auch der Paralleltyp 74LS374 mit D-Flip-Flops verwendet werden. Zur Erinnerung: Ein Latch übernimmt so lange Informationen, die am D-Eingang stehen, wie der Takteingang $E=H$ ist. Gespeichert wird also immer der **letzte** Zustand am D-Eingang. Ein D-Flip-Flop übernimmt dagegen mit einer **Flanke** des Clock-Pulses CP die Daten des D-Eingangs. Ob es die ansteigende oder die abfallende Flanke ist, hängt jeweils vom Typ ab. Die D-Flip-Flops des 74LS374 werden mit der ansteigenden Flanke des Impulses (von Mono-Flop 1) gesetzt. Beim D-Flip-Flop spielt es keine Rolle, wenn sich der Zustand am D-Eingang während der Dauer eines H oder L am Takteingang CP ändert, denn gespeichert wird nur der Zustand während des Spannungssprungs.

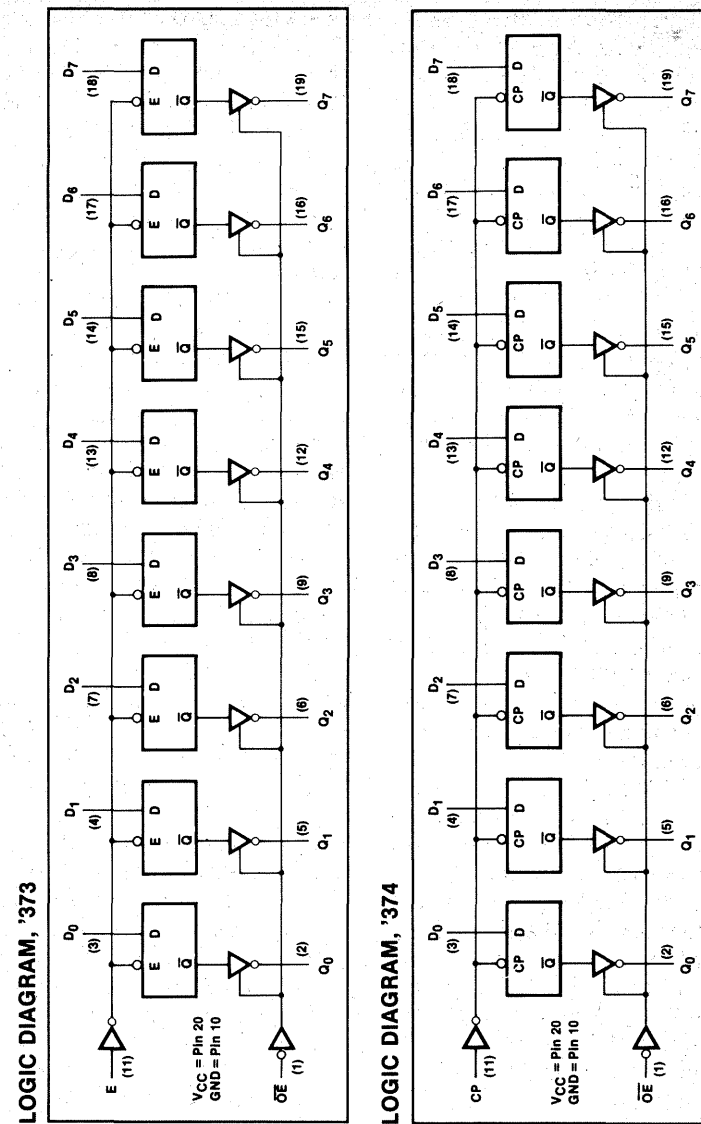
Beide Register sind anschlussgleich mit dem bereits behandelten Register 74LS273 (Bilder 10.7 und 10.8).

Der Vorverstärker

Die DA- und anschließende AD-Wandlung mit einem Widerstandsnetzwerk bedingt, daß der Meßbereich immer in einem bestimmten Verhältnis zur Bezugsspannung steht. Wir benutzen in unserem AD-Wandler die im ZN 425 E enthaltene Referenzspannung von 2,56 V. Damit liegt die **untere** Grenze des **Skalenendwertes** FSR (full scale range) fest. R5/R6 bilden einen Spannungsteiler. Wenn R5 auf $0\ \Omega$ gestellt ist, geht die Eingangsspannung **ungeteilt** auf den Eingang des Komparators. Der Skalenendwert liegt damit bei 2,55 V. Mit R5/R6 kann eine höhere Eingangsspannung geteilt werden, d.h. der Skalenendwert kann **nach oben** erweitert werden. Wählt man R5/R6 z. B. so, daß sich ein Spannungsteilverhältnis 1:1 ergibt, so wird die Eingangsspannung auf die Hälfte geteilt, ehe sie an den Spannungskomparator gelangt. Für den durch die Referenzspannung festgelegten Skalenendwert 2,55 V ist dann eine Eingangsspannung von 5,1 V nötig; der wirkliche Skalenendwert wurde – bezogen auf die Eingangsspannung – verdoppelt. Damit verdoppeln sich aber auch die „Treppenstufen“. Muß bei ungeteilter Eingangsspannung ein Spannungszuwachs von 10 mV entstehen (2,55 V: 255 = 10 mV), damit sich der Zähler um den Wert „1“ erhöht, so sind dazu bei einem durch Spannungsteilung verdoppelten Meßbereich 20 mV erforderlich.

Oft sind aber wesentlich kleinere Spannungen zu messen. Eine Treppe mit 10-mV-Stufen ist bei zahlreichen Messungen viel zu grob.

Ein Beispiel: Die Durchlaßspannung einer Siliziumdiode sinkt bei steigender Temperatur ziemlich linear um



2 mV/°C. Um das Bit-Muster des AD-Wandlers um den Wert „1“ zu erhöhen oder zu erniedrigen, wäre eine Temperaturänderung von 5°C erforderlich. Das Raster wäre außerordentlich grob.

Bild 11.14 Innenschaltungen der Register 74LS373 und 74LS374 (nach VALVO-Unterlagen)

Wenn man dagegen die Eingangsspannung auf ihren fünffachen Wert verstärkt, reichen dazu schon 2 mV; eine Temperaturänderung von 1°C würde also auch im Bit-Muster des AD-Wandlers eine Änderung um den Wert „1“ bewirken.

Daher ist dem eigentlichen AD-Wandler noch ein Gleichspannungsverstärker vorgeschaltet (Bild 11.16). Der Operationsverstärker ist als **nicht-invertierender** Verstärker geschaltet, die Eingangsspannung U_i wird auf den mit „+“ gekennzeichneten Eingang geleitet. Steigt die Eingangsspannung, so steigt durch die Ausgangsspannung U_{aus} ; sinkt U_i , so sinkt auch U_{aus} .

(Würde man die Eingangsspannung auf den invertierenden Eingang [–] leiten, so würde U_{aus} bei steigender U_i sinken und bei sinkender U_i ansteigen.)

Die Spannungsverstärkung V_u hängt von dem Widerstandsverhältnis der Gegenkopplung ab (Bild 11.16). Beim nichtinvertierenden Verstärker wird sie nach der Formel

$$V_u = \frac{R_a + R_b}{R_b} = 1 + \frac{R_a}{R_b}$$

berechnet. Je kleiner R_b im Verhältnis zu R_a ist, desto größer ist die Spannungsverstärkung V_u , und je größer R_b im Verhältnis zu R_a ist, desto kleiner ist V_u . Die absoluten Werte der Widerstände sind in einem sehr weiten Bereich frei wählbar und daher vergleichsweise unwichtig. Es kommt lediglich auf **das Verhältnis der Widerstände zueinander** an.

Man kann die Spannungsverstärkung in sehr weiten Bereichen ändern, wenn man **einen** der Widerstände veränderlich macht. Im allgemeinen ist es günstiger, dazu den der „Masse“ näheren Widerstand R_b zu nehmen.

In Bild 11.15 sind vier verschiedene umschaltbare Spannungsverstärkungen vorgesehen.

R_a (R7 in Bild 11.15) ist mit 270 kΩ frei gewählt. R_b wird umgeschaltet.

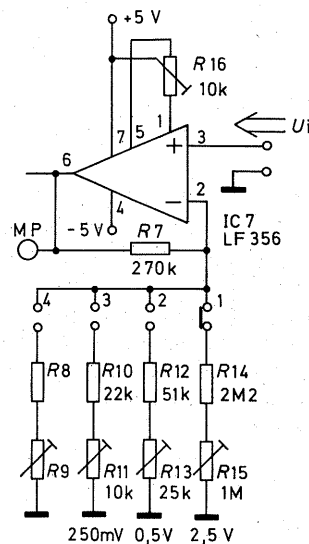


Bild 11.15 Stromlaufplan des Vorverstärkers für den AD-Wandler

Schalterstufe 1: Die Spannungsverstärkung soll etwas größer als 1 sein, etwa 1,08 bis 1,1. Wozu dieser „krumme“ Wert? Es soll erreicht werden, daß die Dualzahlen des AD-Wandlers in einem einfachen Verhältnis zur Eingangsspannung stehen. Die höchste Bitanzeige ist $FF_{16} \cong 255_{10}$. Dieser Zahl soll dann die Eingangsspannung 2,55 V entsprechen. Dann kann man die Ausgangszahl des AD-Wandlers unmittelbar als Spannungswert lesen, sei es als Dualzahl auf der Porteinheit, sei es mit Hilfe des Programms von Seite 283 als 3stellige Dezimalzahl. Nun kann es aufgrund von Fertigungstoleranzen vorkommen, daß die Referenzspannung etwas zu klein

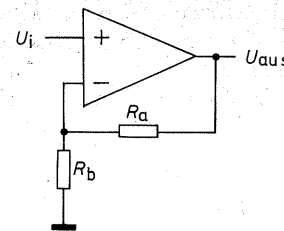


Bild 11.16 Gegenkopplung am Operationsverstärker mit R_a und R_b

oder zu groß ist, daß sich die Zahl 255₁₀ z.B. bei 2,48 V oder bei 2,61 V einstellt. Die geringe Vorverstärkung, verbunden mit der Möglichkeit, die Spannung anschließend durch R5/R6 auf das passende Maß herunterzuteilen, soll die Fertigungstoleranz in der Referenzspannungsquelle ausgleichen.

R_b ist nach der obigen – nach R_b umgestellten – Formel zu berechnen:

$$R_b = \frac{R_a}{V_u - 1}$$

Mit $R_a = 270 \text{ k}\Omega$ und $V_u = 1,09$ muß R_b 3 MΩ betragen. Um Toleranzen der Widerstände auszugleichen, ist R_b in einen Festwiderstand von 2,2 MΩ (R14) und einen Trimmerwiderstand von 1 MΩ (R15) aufgeteilt.

Die übrigen Meßbereiche sind auf die gleiche Weise berechnet.

Die **Schalterstellung 2** ist für eine fünffache Spannungsverstärkung vorgesehen, damit ist der Meßbereich 0,5 V. R_b beträgt rechnerisch 67,5 kΩ und ist in einen Festwiderstand von 51 kΩ (R12) und einen Trimmerwiderstand von 25 kΩ (R13) aufgeteilt.

Die **Schalterstellung 3** ist für eine zehnfache Spannungsverstärkung vorgesehen. Sie läßt es wieder zu, daß der Zahlenwert des AD-Wandlers ohne Umrechnung als Spannungswert gelesen werden kann, denn der Meß-

bereich beträgt nun 250 mV. Der rechnerische Wert für R_b beträgt nun 30 kΩ und ist in einen Festwiderstand von 22 kΩ (R10) und einen Trimmerwiderstand von 10 kΩ (R11) aufgeteilt.

Mit der **Schalterstellung 4** können Sie sich eine Spannungsverstärkung nach Ihren Bedürfnissen einstellen. Deswegen sind für R8 und R9 keine Widerstandswerte angegeben. Sie sollten aber keine zu hohe Spannungsverstärkung einstellen, weil bei der offenen Bauweise mit der Spannungsverstärkung auch die Anfälligkeiten für Störspannungen wächst.

Die Gesamtschaltung des AD-Wandlers

Bild 11.17 zeigt den Stromlaufplan des AD-Wandlers als Einheit. Hinzugekommen sind nur noch die Entkopplungskondensatoren für die positive und die negative Betriebsspannung. Die negative Versorgungsspannung für die Operationsverstärker ist deswegen nötig, damit diese auch an ihren Ausgängen 0 V erreichen können.

Für beide Operationsverstärker sind Trimmerwiderstände zum Offsetabgleich vorhanden (R4 und R16). Der **ideale** Operationsverstärker schaltet den Ausgang dann auf 0 V, wenn beide Eingänge die gleiche Eingangsspannung haben. Durch Fertigungstoleranzen gibt es aber immer gewisse Asymmetrien zwischen den Eingängen, so daß einige mV Spannungsdifferenz zwischen den Eingängen stehen müssen, damit der Ausgang genau 0 V erreicht. Diese geringe Spannungsdifferenz ist der **Offset** (engl. Versatz, versetzte Spannung). Man kann ihn bei sehr vielen Operationsverstärkern über besondere Steuer-

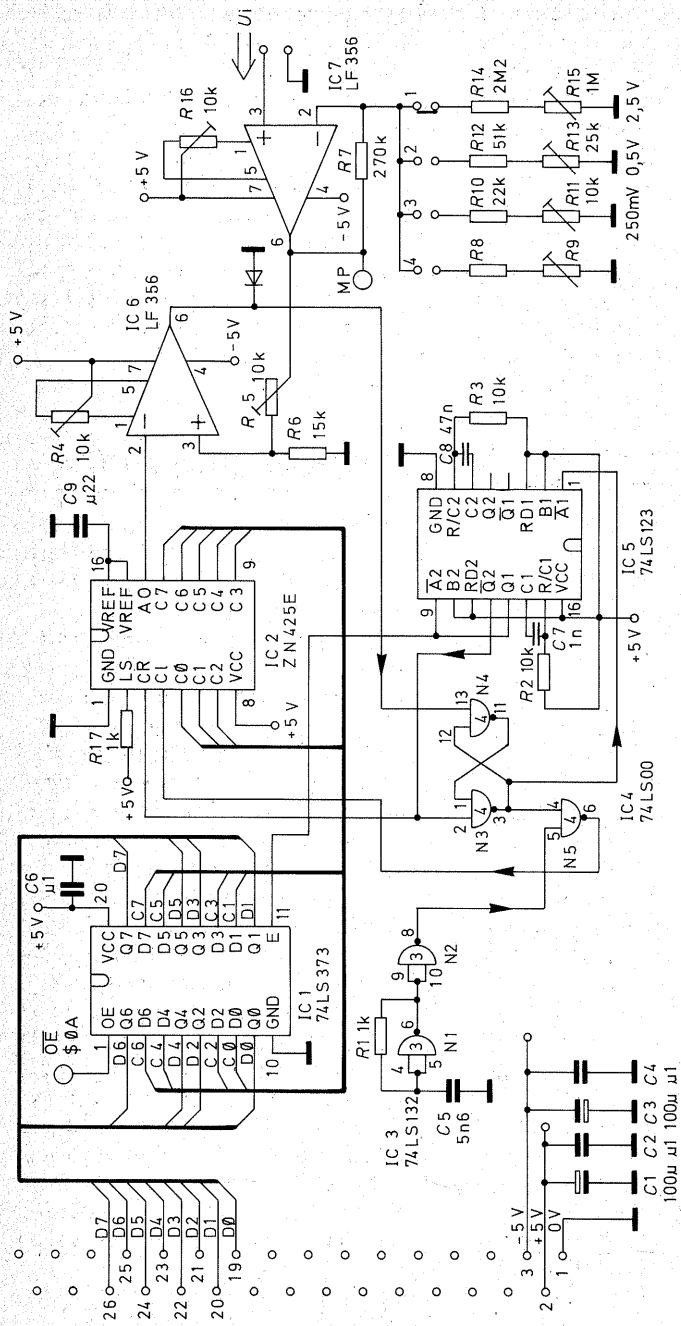


Bild 11.17 Vollständiger Stromlaufplan des AD-Wandlers

eingänge ausgleichen. Das geschieht beim LF 356 durch eine mehr oder weniger asymmetrische Spannungsversorgung der Steuereingänge (Pin 1 und 5). Der Trimmerwiderstand wird jeweils so verstellt, daß bei gleicher

Eingangsspannung am Ausgang möglichst genau 0V zu messen ist (siehe Seite 210).

Die Widerstände R4 und R16 wurden deshalb mit je 10 kΩ bemessen, damit Sie nicht zu viele verschiedene Werte kaufen müssen. Im Datenblatt empfehlen die Hersteller 25 kΩ. Falls Sie Trimmerwiderstände in dieser Größenordnung in Ihrer Bastelkiste haben, nehmen Sie diese. Es kommt aber wirklich nicht auf den genauen Wert von 25 kΩ an.

- 1 Leiterplatte
- 1 31polige Stiftleiste, Baureihe GdsW
- 1 Lötnagel RTM 1,3 mit Steckhülse
- 1 Cynch-Buchse für Printmontage (oder andere Eingangsbuchse nach Belieben) mit Stecker
- 2 Fassung DIL 8
- 2 Fassung DIL 14
- 2 Fassung DIL 16
- 1 Fassung DIL 20
- 1 DIL-Schalter, 4polig
- 1 74LS00
- 1 74LS123
- 1 74LS132
- 1 74LS373 oder 74LS374
- 2 LF 356 oder LF 357
- 1 ZN 425 E
- alle Widerstände möglichst in Metallfilm-Ausführung:
 - 2 Widerstand 1 kΩ
 - 2 Widerstand 10 kΩ
 - 1 Widerstand 15 kΩ
 - 1 Widerstand 22 kΩ
 - 1 Widerstand 270 kΩ
 - 1 Widerstand 51 kΩ
 - 1 Widerstand 2,2 MΩ
- alle Trimmerwiderstände möglichst als Spindeltrimmer oder geschlossene Bauform mit Staubschutzkappe
 - 4 Trimmerwiderstand 10 kΩ
 - 1 Trimmerwiderstand 25 kΩ
 - 1 Trimmerwiderstand 1 MΩ
- 2 Elektrolytkondensator 47 bis 100 µF/16V
- 3 keramischer Scheibenkondensator 0,1 µF
- 1 Folienkondensator 1 nF
- 1 Folienkondensator 5,6 nF
- 1 Folienkondensator 47 nF
- 1 Keramik- oder Folienkondensator 0,22 µF evtl. 1 keramischer Scheibenkondensator 1 nF

Hinweise zum Aufbau

Bild 11.19 zeigt die Kupferbahnen der Platine, Bild 11.20 die Bestückungsseite, Bild 11.21 die Drahtbrücken auf der Lötseite.

Sie beginnen die Bestückung mit der Montage der Eingangsbuchse. In der Stückliste ist eine Cynch-Buchse für Platinenmontage vorgeschlagen. Manche Elektronikläden führen diese Buchsen nicht, und wegen einer einzigen Buchse werden Sie auch nicht den Versandhandel bemühen. Jede andere Buchse ist geeignet, sofern sie nur den Anschluß einer abgeschirmten Leitung zuläßt, so z. B. eine Klinkenbuchse. Bei der Montage Ihrer Eingangsbuchse müssen Sie lediglich darauf achten, daß der Stecker **parallel** zur Leiterplatte eingesteckt wird. Nach der Montage messen Sie mit dem Ohmmeter (niedrigster Ohmbereich) nach, ob die Buchse auch einen guten Massekontakt hat. Anschließend setzen Sie die Stiftleiste und die Drahtbrücken auf der Bestückungsseite sowie die IC-Fassungen ein.

Fügen Sie dann R1, C5 und IC3 ein. Prüfen Sie, am besten mit einem Os-

Bild 11.18 Stückliste für den AD-Wandler

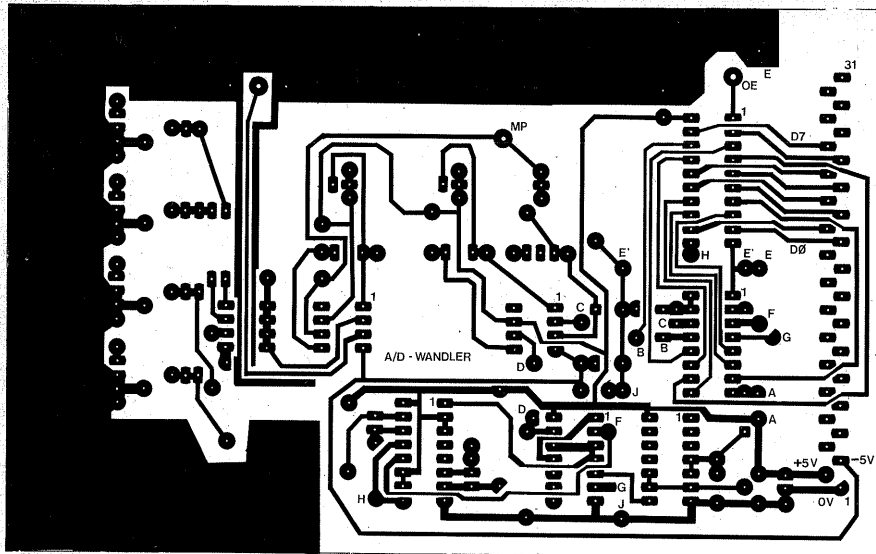


Bild 11.19 Leiterbahnbild für die AD-Wandler-Platine

zillographen, ob der Oszillator IC3 schwingt, indem Sie am Ausgang (Pin 8) messen. Wenn Ihnen kein Oszillograph zur Verfügung steht, löten Sie provisorisch auf der Unterseite der Leiterplatte einen Elko mit 100 oder 220 μF parallel zu C5 (Minuspol an Masse). Dann können Sie die Funktion des Oszillators mit dem Vielfachinstrument prüfen. Sie müssen am Pin 8 von IC3 eine pulsierende Spannung messen können. Wenn der Oszillator schwingt, entfernen Sie den Elko wieder und setzen die übrigen Bauelemente sowie die Drahtbrücken auf der Lötseite ein (Bild 11.21).

Die Trimmerpotentiometer verdienen Ihre Aufmerksamkeit: Wenn es Ihre Bastelkasse zuläßt, sollten Sie Spindeltrimmer verwenden. Wenn Sie einen Kompromiß schließen müssen, so leisten Sie sich wenigstens für R5

und R11 (Meßbereich 250 mV) die teuren Trimmer.

Die Platine ist für die unterschiedlichen Rastermaße von Trimmern („liegend“) eingerichtet. Wenn Sie „Normal“-Trimmer verwenden, achten Sie beim Kauf auf eine gekapselte Bauform.

Auch wenn Sie im allgemeinen Fassungen für ICs und Bauelemente mit IC-ähnlichen Anschlüssen bevorzugen, sollten Sie den DIL-Schalter unmittelbar in die Leiterplatte einlöten, weil er in einer Fassung nicht ausreichend sicher sitzt.

Bild 11.22 zeigt den fertigen Aufbau auf einer geätzten Leiterplatte.

Hinweise zum Aufbau auf einer Experimentierplatte

Sie beginnen die Arbeit wie üblich mit der Montage der Stiftleiste (Zählmarkierungen nicht vergessen), der Buchse und den IC-Fassungen. Orientierungshilfen können Sie Bild 11.23 entnehmen.

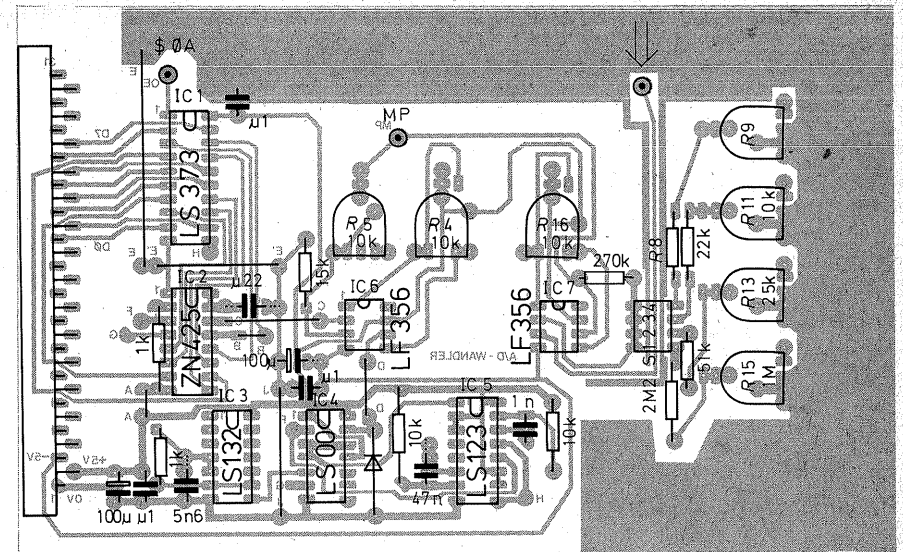


Bild 11.20 Bestückungsseite der AD-Wandler-Platine

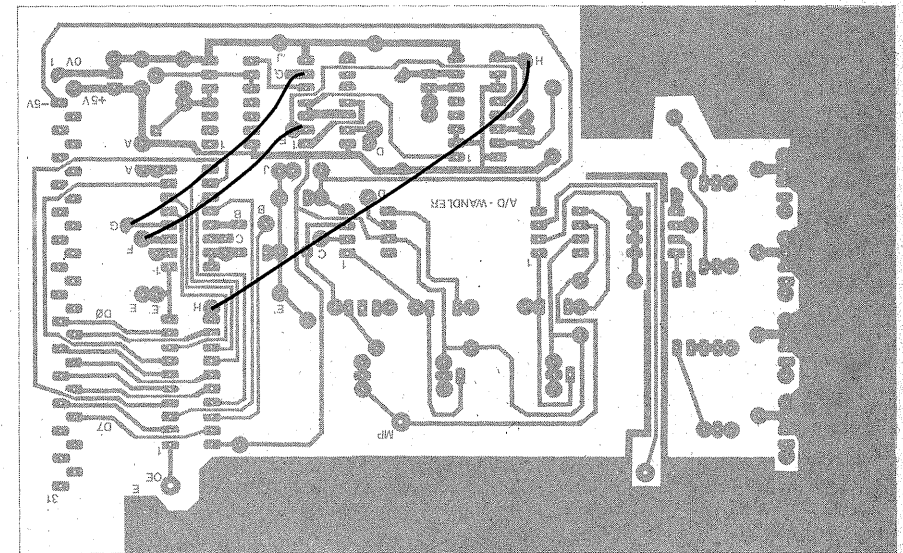


Bild 11.21 Drahtbrücken auf der Lötseite der AD-Wandler-Platine

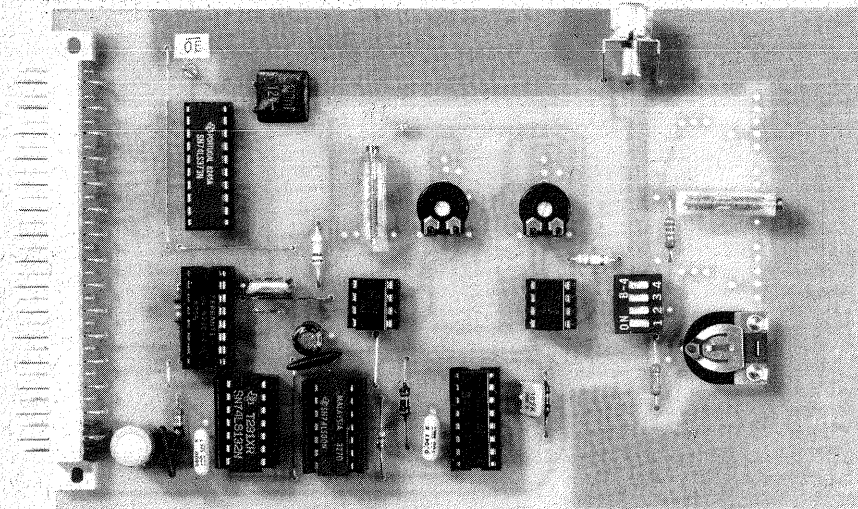


Bild 11.22 Der fertige AD-Wandler auf der Platine

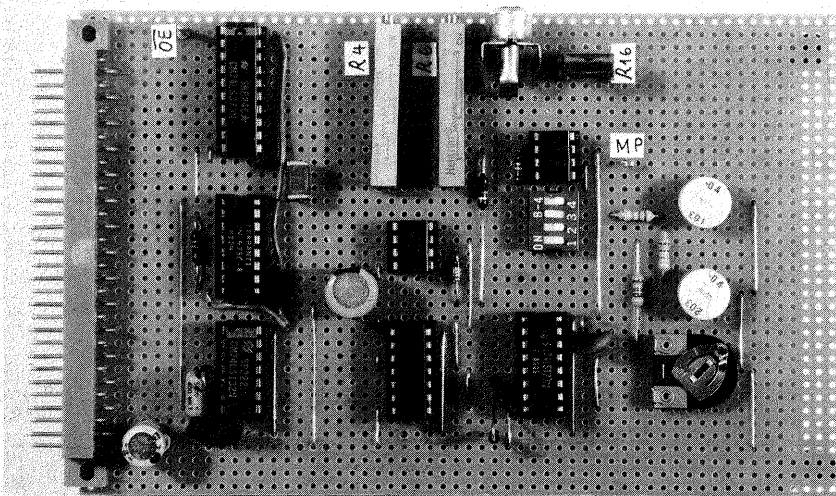


Bild 11.23 Der fertige AD-Wandler auf einer Experimentierplatte

Verlegen Sie zuerst die Leitungen für die Betriebsspannungen $+5\text{ V}$, 0 V und -5 V . Prüfen Sie mit dem Ohmmeter nach,

1. ob alle GND-Anschlüsse (Pin 10/IC1, Pin 1/IC2, Pin 7 bei IC3 und IC4, Pin 8/IC5) Durchgang zum Stift 1 der Stiftleiste (0 V) haben;
2. ob alle $+5\text{-V}$ -Anschlüsse Durchgang nach Stift 2 der Stiftleiste ($+5\text{ V}$) haben (Pin 20/IC1, Pin 8/IC2, Pin 14 bei IC3 und IC4, Pin 16 bei IC5 und Pin 7 bei IC6 und IC7,
3. ob die Anschlüsse 4 von IC6 und IC7 ($-U$) Durchgang zum Stift 3 (-5 V) der Stiftleiste haben.

Dann messen Sie von Stift 1 (0 V) den Widerstand nach Stift 2 ($+5\text{ V}$) und Stift 3 (-5 V). Das Ohmmeter muß $\infty\Omega$ anzeigen, weil sich ja noch keine Bauelemente auf der Platte befinden, es würde aber einen Kurzschluß der Betriebsspannungen anzeigen. Stecken Sie anschließend die Platte in eine der Busleisten, und messen Sie, ob an allen IC-Fassungen die Betriebsspannungen richtig anliegen. Beachten Sie, daß die Operationsverstärker $+5\text{ V}$ und -5 V gegen Masse (0 V) benötigen. Fehlt eine dieser Spannungen, so können die Operationsverstärker beschädigt werden. Deswegen sind die obigen Prüfungen so wichtig.

Als nächste Einheit verdrahten Sie den Oszillator (IC3) und prüfen ihn, wie oben beschrieben.

Den Rest der Schaltung können Sie in beliebiger Reihenfolge verdrahten.

Wenn alles funktioniert, sprühen Sie die Lötseite der Leiterplatte mit Plastiklack ein, um sie vor Oxidation zu schützen.

Inbetriebnahme und Abgleich des AD-Wandlers

Vorbereitungen

Sie benötigen ein abgeschirmtes Eingangskabel mit passendem Stecker für die Eingangsbuchse. Es braucht nicht besonders kapazitätsarm zu sein, so daß ein Mikrofon- oder Diodenkabel reicht. An dieses Kabel löten Sie ein Potentiometer mit linearer Kennlinie (R_{max} ca. 10 bis $47\text{ k}\Omega$) gemäß Bild 11.24 a sowie zwei kurze Litzen, am besten mit kleinen Krokodilklemmen japanischer Prüfkabel versehen, die Sie an eine Flachbatterie anklemmen können.

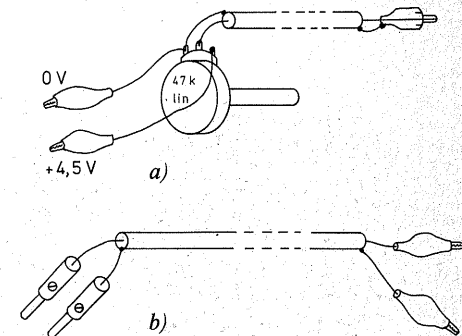


Bild 11.24 Hilfsmittel für den AD-Wandler

Außerdem benötigen Sie eine abgeschirmte Leitung, über die Sie Ihr Vielfachinstrument anschließen. Versetzen Sie ein Ende mit Bananensteckern für das Vielfachinstrument, das andere möglichst mit den bereits erwähnten kleinen Krokodilklemmen oder IC-Greifklemmen. Die Abschirmung ist für die Minusbuchse des

Vielfachinstruments vorgesehen, die „Seele“ für die Plusbuchse (Bild 11.24b).

Da Sie für den AD-Wandler öfter abgeschirmte Leitungen benötigen werden, lohnt es sich, die beiden erwähnten Kabel nicht nur provisorisch, sondern für den Dauergebrauch anzufertigen (Bild 11.24).

Die benötigten Bausteine

Stecken Sie die Speicherkarte und den Portbaustein auf die Busplatte. Schalten Sie den Speicher in seine Version „1“, so daß das RAM bei Adresse 0000 beginnt. Stecken Sie nun den AD-Wandler auf, wenn möglich, über den Busadapter (siehe Seite 90), dann können Sie mit dem AD-Wandler besser hantieren. Verbinden Sie den Anschluß \overline{OE} des AD-Wandlers mit dem Peripherieausgang 0A des Portbausteins. Schließen Sie den Eingang des AD-Wandlers mit einem Prüfkabel kurz.

Der Offsetabgleich von IC7

1. Schalten Sie mit dem DIL-Schalter 1 den größten Spannungsmeßbereich ein, das ist der Bereich mit dem größten Gegenkopplungswiderstand ($R_{14} + R_{15}$) gegen 0 V. Alle anderen Schalter müssen offen sein.
2. Messen Sie die Ausgangsspannung von IC7 am Meßpunkt MP (Meßbereich 5 bis 10 V -, Minuskabel an 0 V, Pluskabel an MP). Verstellen Sie R16 **langsam**, bis das Meßinstrument 0 V anzeigt. Die Ausgangsspannung von IC7 muß sich mit R16 leicht nach positiven wie auch nach negativen Werten gegenüber 0 V verstellen lassen.

Der Offsetabgleich von IC6

1. Laden Sie folgendes Programm in den Speicher:

Adr.					
0000	55	0A	REDE,0A	Lies den AD-Wandler in R1	
	02	D5	09	WRTE,09	Schreibe (R1) in den Port 09
	04	1F	00	00	BCTA
					Springe an den Anfang zurück

Lassen Sie dieses Schleifenprogramm mit der Clockfrequenz 1 MHz laufen. Noch wird der Port nichts anzeigen, weil der AD-Wandler ja noch keine Eingangsspannung erhält. Dieses Programm zeigt Ihnen aber bei allen folgenden Einstellungen die Ergebnisse des AD-Wandlers an.

2. Schließen Sie Ihr Vielfachinstrument an den Ausgang von IC6 an (Kathode der Diode zwischen IC4 und IC5). Verstellen Sie R4 **langsam**, bis Sie den Punkt erreichen, wo das Meßinstrument vom positiven Ausschlag zum negativen Ausschlag umkippt (oder umgekehrt). Sie werden nicht konstant 0 V einstellen können, weil der Operationsverstärker ohne Gegenkopplung und daher mit höchster Spannungsverstärkung arbeitet. Geringste Spannungsverschiebungen an seinem Eingang bringen den Ausgang zum Umkippen. Bei dieser Einstellung werden Sie bemerken, daß die LED D0 des Portbausteins flackert.

Die Einstellung des Grundmeßbereichs

Es wäre gut, wenn Sie für den folgenden Arbeitsgang zwei Vielfachinstrumente zur Verfügung hätten, eines, um die Eingangsspannung zu messen, das andere, um die Spannung an MP zu messen.

1. Lösen Sie den Kurzschluß am Eingang des AD-Wandlers, und schließen Sie über das vorbereitete Eingangskabel und das Potentiometer (Bild 11.24) die Flachbatterie an (Minus an 0 V). Zugleich messen Sie die Eingangsspannung über die vorbereitete abgeschirmte Leitung.

Achtung: Eingangs- und Meßkabel müssen unbedingt abgeschirmt sein, weil sonst HF-Einstrahlungen oder Brummspannungen das Ergebnis verfälschen können.

2. Stellen Sie mit dem Potentiometer eine Eingangsspannung von möglichst genau 2,5 V ein.

3. Messen Sie – möglichst mit einem zweiten Meßinstrument – die Spannung an MP (Ausgang des Vorverstärkers IC7). Es wäre gut, wenn Sie das erste Instrument nicht vom Eingang des AD-Wandlers abzutrennen brauchten, denn es belastet durch seine Stromaufnahme den Spannungsteiler. Die eingestellten 2,5 V Eingangsspannung gelten **nur mit** der Belastung durch den Instrumentenstrom. Wenn Sie das Meßinstrument abklemmen, steigt die Eingangsspannung je nach dem Innenwiderstand Ihres Vielfachinstruments mehr oder weniger an. Der Spannungszuwachs kann – abhängig auch vom Widerstand des Potentiometers – 0,1 V oder sogar etwas mehr betragen.

Verstellen Sie R15 so, daß Sie an MP eine Spannung von ca. 2,6 bis 2,7 V messen können. Der Ausgang von IC7 ist niederohmig genug, so daß seine Ausgangsspannung durch den Instrumentenstrom nicht über Gebühr beeinflusst wird.

4. Nun setzen Sie mit R5 den Skalenendbereich fest. Drehen Sie R5 zunächst auf einen mittleren Widerstandswert. Inzwischen wird der Port einen Spannungswert anzeigen. Ver-

stellen Sie R5 **langsam**, nach einem kleineren Widerstandswert. Dabei werden Sie beobachten, daß die Portanzeige immer höhere Dualzahlen anzeigt. Stellen Sie R5 so weit, daß im Port die Anzeige $FA_{16} (\cong 250_{10})$ erscheint. R5 sollte dann einen nur noch sehr kleinen Wert haben. R5 und R6 bilden zusammen einen Spannungsteiler, dessen Spannungsknoten die Eingangsspannung für den Komparator IC6 abgibt. Die von IC7 verstärkte Spannung soll bei diesem Meßbereich in nahezu voller Höhe an IC6 gelangen, damit bei der Einstellung der übrigen Meßbereiche die durch die Gegenkopplungswiderstände einstellbaren Verstärkungsfaktoren ausreichen.

Muß R5 auf einen recht großen Wert – deutlich mehr als 10% seiner Widerstandsbahn – eingestellt werden, so setzen Sie die Verstärkung von IC7 herab, indem Sie R15 auf einen größeren Wert stellen.

Können Sie die Anzeige FA_{16} auch dann nicht erreichen, wenn Sie R5 auf 0 Ω gestellt haben, so reicht die Verstärkung von IC7 nicht aus. Verstellen Sie in dem Fall R15 so, daß Sie an MP eine Spannung von 2,8 oder 2,9 V messen.

5. Kontrollieren Sie die Arbeit des AD-Wandlers, indem Sie die Eingangsspannung auf möglichst genau 1,25 V halbieren, das Bitmuster der Portanzeige muß sich dabei um eine Stelle nach rechts verschieben ($D7_{16} \cong 125_{10}$). Ob sich genau der Wert $D7_{16}$ einstellt, hängt in erster Linie davon ab, wie gut es Ihnen gelungen ist, die Spannungen 2,5 V bzw. 1,25 V einzustellen. Der Port muß aber einen Wert nahe dieser Größe anzeigen. Sollten die Abweichungen sehr groß sein, wiederholen Sie den Offsetabgleich von IC6.

Mit dem Erreichen der Anzeige FA₁₆ bzw. des halben Werts ist der Abgleich des Grundmeßbereichs beendet. Heften Sie kleine Aufklebepunkte auf R4, R5 und 16, damit Sie diese Trimmerwiderstände nicht versehentlich wieder verstellen.

Der Abgleich der übrigen Meßbereiche

Zum Abgleich der übrigen Meßbereiche brauchen Sie nur noch die Gegenkopplungen an IC7 einzustellen. Stellen Sie als nächsten Bereich den Schalter 3 auf „ein“ (und alle anderen auf „aus“). Geben Sie auf den Eingang eine Spannung von 250 mV und stellen Sie R11 so ein, daß sich auf dem Port die Anzeige FA₁₆ ergibt. Damit ist der 250-mV-Bereich eingestellt.

In gleicher Weise können Sie sich die beiden übrigen Meßbereiche je nach Bedarf einstellen.

Die Anwendung des AD-Wandlers

Mit dem AD-Wandler verfügen Sie über ein **Digital-(Milli-)Voltmeter**. Es hat einen hochohmigen Eingang und kann bei entsprechender Wahl des Verstärkungsfaktors sehr empfindlich sein. Falls sich der Ortssender mit HF-Einstrahlungen störend bemerkbar machen sollte, löten Sie einen keramischen Scheibenkondensator mit 1 nF direkt vom „heißen“ Ende der Eingangsbuchse gegen die Massefläche, um HF-Spannungen kurzzuschließen.

Mit dem AD-Wandler können Sie unmittelbar die Spannungen messen,

die von Sensoren geliefert werden. Sie können z. B. einen LDR oder einen NTC-Widerstand mit einem Festwiderstand zu einem Spannungsteiler in Reihe schalten und die den Beleuchtungs- oder Temperaturänderungen entsprechenden Spannungsänderungen messen und auswerten. Wegen des hochohmigen Eingangs können Sie die Sensoren auch in eine Widerstandsbrücke einfügen und mit dem AD-Wandler die Spannungsänderungen in der Brückendiagonale messen. Die Brückenschaltung läßt es zu, daß Sie Anfang und Ende des Meßbereichs bestimmen. Sie können auch ein Potentiometer als Spannungsteiler schalten und damit Bewegungen oder Drehwinkel in Spannungsänderungen umsetzen usw.

Die Anwendungen des AD-Wandlers sind ein sehr weites Feld. Sie können aus Platzmangel im Rahmen dieser Bauanleitung nicht weiter behandelt werden. Für sie ist ein ganzer Band vorgesehen, der verschiedene Sensoren und Aktuatoren, deren Anwendungen und dazu nützliche Auswertungs- bzw. Umrechnungsprogramme enthält. Die Software wird auch als EPROM lieferbar sein.

Kapitel 12

Tastatur und Anzeige (Baugruppe 9)

Der Baustein Tastatur und Anzeige leistet im Prinzip das, was Daten- und Adreßbaustein (Baugruppe 3 und 5) auch leisten, nur sehr viel komfortabler. Der Baustein erspart Ihnen viel von der mühseligen Eingabe- und Ablesearbeit. Das Gesetz von der Erhaltung der Arbeitsmenge gilt auch hier: Die Arbeit, die Sie nicht erledigen, muß aber doch von jemandem erledigt werden, und dieser „jemand“ ist Ihr Computer.

Er benötigt dazu ein **Betriebsprogramm**, das **Monitorprogramm**, kurz den Monitor. Dieses umfangreiche Programm (2 Kbyte), das u. a. auch sehr viele für andere Zwecke nützliche Unterprogramme („Routinen“) enthält, ist die Voraussetzung zur Inbetriebnahme des Tastatur- und Anzeigebausteins. Es muß beim Einschalten der Betriebsspannung bereits zur Verfügung stehen und daher in einem Festwertspeicher (siehe Seite 56) vorhanden sein. Sie können den Monitor als EPROM bei der auf Seite 12 genannten Firma beziehen. Eine ausführliche Dokumentation ist in diesem Band aus Platzmangel nicht möglich. Sie werden eine detaillierte Beschreibung im „Programmierkur-

sus“ finden. Hinweise zur Benutzung der Routinen finden Sie aber bereits in diesem Band auf Seite 253.

Die Anzeigeeinheit

Die Siebensegmentanzeige und die Zeichendecodierung

Eine einfache Form der alphanumerischen Anzeige ist die sogenannte Siebensegmentanzeige (Bild 12.1 a). Sie besteht aus acht (!) Segmenten, in denen sich jeweils eine LED (bei großen Segmenten auch mehrere) befindet. Sieben Segmente bilden den Rahmen für die darzustellenden Zeichen; das achte Segment ist der Dezimalpunkt. Die Bezeichnung der Segmente mit den Buchstaben a bis f und dp für den Dezimalpunkt ist genormt und ermöglicht so eine Verständigung. Die LEDs in den Segmenten sind mit jeweils einem ihrer Anschlüsse intern zusammengefaßt, entweder mit den Kathoden („gemeinsame Kathode“, gK) oder mit den Anoden („gemeinsame Anode“, gA).

In unserer Anzeigeeinheit verwenden

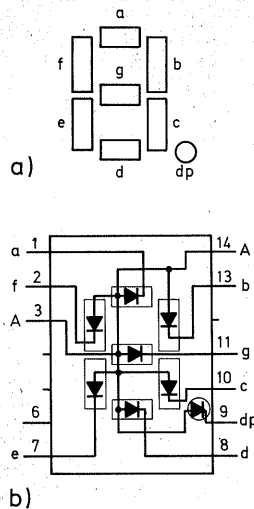


Bild 12.1 Siebensegmentanzeige
 a) Bezeichnung der Segmente,
 b) Anschlußbelegung und Lage der Dioden in den Anzeigen hp 5082-7611 bzw. HDSP 3531

wir Bausteine mit gemeinsamen Anoden (Bild 12.1.b). Der Anodenanschluß wird über einen Schalttransistor an +5 V gelegt (Bild 12.4). Die Segmente, die leuchten sollen, müssen über Vorwiderstände an 0 V geschaltet werden (bei Anzeigen mit gK wäre die Beschaltung genau entgegengesetzt). Die aufleuchtenden Segmente bilden dann das Zeichen, die nicht an 0 V geschalteten Segmente bleiben dunkel und sind hinter einem Farbfilter unsichtbar.

Zwischen dem darzustellenden Datenwort und der Siebensegmentanzeige ist ein Decoder erforderlich, der das Bitmuster, z. B. das einer Dualzahl, in das Bitmuster umsetzt, das diejenigen Segmente der Anzeige aktiviert, die zusammen die entsprechende Zehner- oder Hex-Zahl zeigen.

Es gibt verschiedene BCD zum Siebensegmentdecoder (BCD: binär codiertes Dezimalsystem). Sie wurden – wie die Siebensegmentanzeige selbst – zur Darstellung von **Ziffern** entwickelt. Sie decodieren jeweils 4 Bit, was zur Anzeige einer dekadischen Stelle ausreicht, und stellen die Ziffern „0“ bis „9“ sowie einige Sonderzeichen dar.

Wir könnten solche Decoder verwenden, denn 4 Bit sind ja gerade wieder 1 Nibble (siehe Seite 95). Mit zwei Decodern und zwei Siebensegmentanzeigen ließe sich also 1 Byte in hexadezimaler Schreibweise darstellen – freilich nicht ganz: Die Ziffern „0“ bis „9“ würden richtig erscheinen, statt der Buchstaben „A“ bis „F“ aber irgendwelche andere Zeichen, nur nicht die Buchstaben. „F₁₆“ würde bei manchen Decodern das Verdunkeln sämtlicher Segmente bedeuten.

Andere Decoder, z. B. der F9368 (Fairchild), können alle Hex-Zeichen von „0“ bis „F“ binär zu Siebensegment decodieren. Sie sind aber sehr teuer und lassen außer den 16 Zeichen keine weiteren Darstellungen zu.

Daher ist es praktischer, die Decodierung dem Computer per Software (Monitor) zu übertragen. Es paßt ja ausgezeichnet, daß der Computer über acht Datenleitungen und die Anzeige über ebenfalls acht Segmente verfügt. Also wird jedem Segment eine Datenleitung zugeordnet:

- D0 $\hat{=}$ Segment a
- D1 $\hat{=}$ Segment b
- D2 $\hat{=}$ Segment c
- D3 $\hat{=}$ Segment d
- D4 $\hat{=}$ Segment e
- D5 $\hat{=}$ Segment f
- D6 $\hat{=}$ Segment g
- D7 $\hat{=}$ Segment dp

Die Zuordnung ist willkürlich. Jede andere wäre ebenso möglich. Diese ist aber leicht zu behalten, weil sie der Reihe nach geht.

Über die Datenleitungen lassen sich mit Hilfe des Schreibbefehls WRTE alle Segmente in beliebigen Zusammenstellungen zum Aufleuchten bringen. Dabei ergibt sich eine Vielzahl von Möglichkeiten, und mit viel Phantasie kann man nicht nur die Ziffern „0“ bis „9“ darstellen, sondern das ganze Alphabet dazu. Für den Buchstaben „H“ z. B. sind die Segmente b, c, e, f und g zu aktivieren (Bild 12.2). Dazu müssen die Datenleitungen D1, D2, D4, D5 und D6 logisch 1 sein ($0111\ 0110 \hat{=} 76_{16}$). Den vollen Zeichensatz finden Sie in Bild 12.3. Die Zeichen können über die Routine CONVert (engl. umwandeln; Einsprungbefehl BB 85 und indizierte Adressierung über R3; siehe Laufschriftprogramm Seite 264) abgerufen werden.

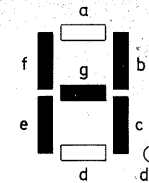


Bild 12.2 Buchstabe „H“ in Siebensegmentdarstellung

Der Hex-Code in Bild 12.3 gibt jeweils die Stellung des Zeichens in der Tabelle an. Diese Zahl ist der Index, unter dem das Zeichen zu finden ist. Das Alphabet ist freilich wegen der geringen Anzahl der Segmente sehr grob: Klein- und Großschreibung gehen durcheinander, einige Zeichen sind doppelt belegt ($1 \hat{=} i$; $5 \hat{=} S$; $2 \hat{=} Z$). Beim Lesen der Zeichen hel-

Bild 12.3 Im Monitor untergebrachter Siebensegmentzeichensatz

Zeichen	Hex-Code	7-Seg-Code	Zeichen	Hex-Code	7-Seg-Code	Zeichen	Hex-Code	7-Seg-Code	Zeichen	Hex-Code	7-Seg-Code	Zeichen	Hex-Code	7-Seg-Code	
0	00	3F	A	0A	77	K	14	75	U	1E	1C		28	41	
1	01	06	B	0B	7C	L	15	38	V	1F	3C		29	48	
2	02	5B	C	0C	39	M	16	37	W	20	3E		2A	80	
3	03	4F	D	0D	5E	N	17	54	X	21	64		2B	40	
4	04	66	E	0E	79	O	18	5C	Y	22	6E		leer	2C	00
5	05	6D	F	0F	71	P	19	73	Z	23	5B				
6	06	7D	G	10	7D	Q	1A	67		24	63				
7	07	07	H	11	76	R	1B	50		25	49				
8	08	7F	I	12	06	S	1C	6D		26	22				
9	09	6F	J	13	0E	T	1D	78		27	14				

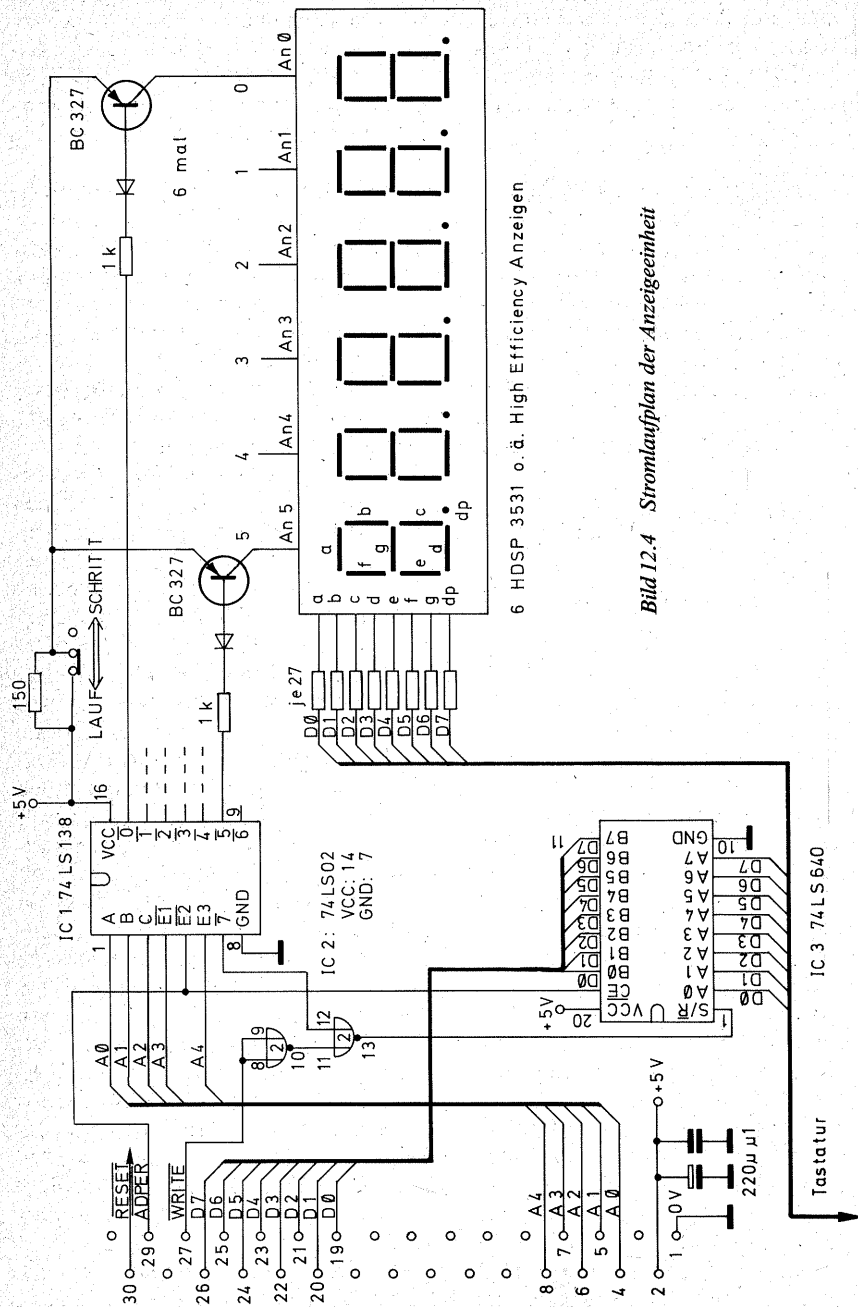


Bild 12.4 Stromlaufplan der Anzeigeeinheit

fen entweder Phantasie oder Übung – am besten ist es, man hat beides. Nach kurzer Zeit gewöhnen Sie sich aber gewiß an die etwas seltsame Darstellung.

Die Schaltung der Anzeigeeinheit

Die Anzeigeeinheit (Bild 12.4) läßt sich am besten mit dem Portbaustein aus Kapitel 10 vergleichen. Dort wurden 1×8 LEDs zum Leuchten gebracht, hier sind es 6×8 LEDs, denn die Anzeigeeinheit hat sechs Stellen („Digits“). Jede Stelle hat ihre eigene Portadresse:

Digit 0 (ganz rechts):	10 ₁₆
1:	11
2:	12
3:	13
4:	14
5:	15.

Es wurde bereits erwähnt, daß sich die Behandlung von Ports prinzipiell nicht von der Behandlung von Speicherstellen unterscheidet.

Bei der Zusammenschaltung von Speicherbausteinen werden die Daten- und Adreßleitungen der einzelnen Bausteine parallelgeschaltet. Der jeweils benutzte Speicherbaustein wird von einem Adreßdecoder über den \overline{CS} -Eingang aktiviert.

So ist es auch hier: Die Segmente (also gewissermaßen deren Datenleitungen) sind parallelgeschaltet und einschließlich ihrer Vorwiderstände über einen Bustransceiver mit Tri-State-Ausgängen (IC2) an den Datenbus angeschlossen. Ein Adreßdecoder (IC1, 74LS138, vgl. Bild 9.7 bis 9.10) decodiert die Adreßleitungen A0 bis A4, wobei der Enable-Eingang E1 quasi als vierter Adreßeingang benutzt wird. So ist der Decoder nur

dann freigegeben, wenn A4=H ist, und daher kann es nicht zu Konflikten mit dem Adreßdecoder der Porteinheit kommen; der wird nur dann freigegeben, wenn A4=L ist (Bild 10.4).

Die \overline{CS} -Eingänge der Anzeigen sind die Basen der Schalttransistoren an den gemeinsamen Anoden der Digits. Der Übersichtlichkeit zuliebe wurden in Bild 12.4 nur der erste und der letzte Schalttransistor gezeichnet. Jede Anzeige erhält jedoch einen eigenen Transistor, der von je einem Decoderausgang (aktiv L für pnp-Transistoren) geöffnet oder gesperrt wird.

Die Anzeigen werden nacheinander aktiviert. Es soll z. B. „HALLO.“ in die 6stellige Anzeige geschrieben werden:

1. Schritt: Der Prozessor schreibt mit einem WRTE-Befehl in die Portadresse 15₁₆ das „H“ (0111 0110, Bild 12.2). Der **invertierende** Bustransceiver kehrt das Datenwort um und schaltet D1, D2, D4, D5 und D6 auf L, die übrigen Datenleitungen auf H. Damit liegen die Kathoden der zu aktivierenden Segmente b, c, e, f und g am Minuspol der Betriebsspannung. Wichtig ist an dieser Stelle das fan out des Bustransceivers, weil jeder Ausgang ja den gesamten Strom des zugeordneten Segments aufnehmen können muß. Der 74LS640 (Anschlußbelegung und Datenfluß wie beim 74LS245, Bild 10.10, nur invertierend) kann pro Ausgang – 24 mA aufnehmen, der Paralleltyp 74LS640-1 sogar – 48 mA, das reicht. Ein Inverter, der keinen so starken Strom schalten könnte, würde die Segmente nicht ausreichend hell zum Leuchten bringen.

Der Adreßdecoder schaltet in diesem Augenblick **nur** den Ausgang 5 auf L. Daher wird **nur** der pnp-Transistor

für Digit 5 leitend. Es können deswegen **nur** die Segmente von Digit 5 aufleuchten, denn alle anderen sind ja durch die Schalttransistoren von +5V abgetrennt. Diese Stellenauswahl erlaubt es, alle gleichnamigen Segmente parallelzuschalten.

2. *Schritt*: Nun schreibt der Prozessor ein „A“ in die Portadresse 14₁₆. Wieder werden alle zu „A“ gehörenden Segmente an L geschaltet, aber nur Digit 4 erhält über seinen Schalttransistor – aktiviert vom Decoderausgang 4–Strom. Also leuchtet das „A“ nur in Digit 4 auf.

3. *Schritt*: Der Prozessor schreibt ein „L“ in Adresse 13₁₆. Die zu „L“ gehörenden Segmente werden an 0V geschaltet, aber nur Digit 3 erhält Strom.

4. *Schritt*: Wieder schreibt der Prozessor ein „L“, aber diesmal in die Portadresse 12₁₆. Der Transistor von Digit 2 wird von Decoderausgang 2 durchgeschaltet und versorgt Digit 2 mit Strom.

5. *Schritt*: Der Prozessor schreibt ein „O“ in die Portadresse 11₁₆ und schaltet damit nur den Transistor für Digit 1 durch.

6. *Schritt*: Der Prozessor schreibt einen „.“ in die Portadresse 10₁₆. Nun wird der Transistor für Digit 0 durchgeschaltet, und der Punkt erscheint nur dort.

Danach beginnt der Zyklus von vorn. Er muß sich in einer Schleife rasend schnell wiederholen, damit sich eine Daueranzeige ergibt. Sonst würden alle Digits einmal aufblitzen, und man hätte nichts gesehen.

Die Anzeigen werden **nacheinander** zum Aufleuchten gebracht, und zwar so schnell, daß das Auge den Wechsel nicht mehr erkennt und den Eindruck hat, als leuchteten alle Anzeigen gleichzeitig. Dieses Verfahren heißt

(Zeit-) **Multiplex-Betrieb** (multiplex, lat. vielfältig).

Die Aufleuchtzeiten sind sehr kurz. Sie betragen jeweils nur die Dauer zweier CLOCK-Pulse. Bei einer CLOCK-Frequenz von 1 MHz sind das gerade 2 µs. Dann geschieht eine ganze Weile gar nichts, bis die nächste Anzeige aufblitzt – das Programm muß ja abgearbeitet werden: Die Zeichen müssen geholt, und die Schreibbefehle müssen ausgeführt werden. Sie können ja später einmal den Anzeigeyklus durchtasten und zählen, wie viele Dunkelzeiten einer einzigen Hellzeit einer Anzeige gegenüberstehen, wie viele Maschinenzyklen vergehen müssen, bis eine Anzeige wieder einmal aufleuchtet.

Daher müssen die Anzeigen besonders hell aufleuchten. Das erreicht man einerseits durch eine **starke Überlastung**. Sie ist zulässig, weil der Stromfluß nur so kurze Zeit dauert. Die Vorwiderstände in Bild 12.4 haben einen Wert von je 27 Ω. Zum Dauerleuchten wären Widerstände von mindestens 180 Ω erforderlich.

Achtung: Wenn Sie die Anzeige oder ein anderes Programm im Singlestep oder gar Einzel-CLOCK durchtasten, so müssen Sie den Schalter „LAUF-SCHRITT“ auf „SCHRITT“ schalten. Dann liegt ein gemeinsamer Vorwiderstand von 150 Ω in der Versorgungsleitung der Anzeigeeinheit. Er begrenzt den Anzeigenstrom auf ein Maß, das die Anzeigen dauernd ertragen. **In der Stellung „LAUF“ ist der Vorwiderstand kurzgeschlossen. Ein Dauerstrom, der sich beim Durchtasten eines Programms ergeben würde, wäre dann so groß, daß er die Anzeigen beschädigen könnte.**

Trotz der Überlastung wäre eine „normale“ Siebensegmentanzeige meist nicht hell genug, um angesichts

der sehr kurzen Lichtblitze und der langen Dunkelzeiten eine zufriedenstellende Gesamthelligkeit zu erzeugen. Daher sollten in dieser Schaltung sogenannte **High-Efficiency-Anzeigen** (high efficiency, engl. hoher Wirkungsgrad) verwendet werden. Sie leuchten einige zimal heller als normale Anzeigen; geeignet sind z. B. die Typen hp 5082-7611 oder HDSP 3531 o. ä.

Leider sind diese Anzeigen nicht überall erhältlich und etwa doppelt so teuer wie normale. Wie wäre es möglich, ohne sie auszukommen? Man müßte das Datenwort jeder Anzeige in einem Auffangregister zwischenspeichern, so wie dies in der Portanzeige (siehe Seite 178) geschieht. Für die sechs Anzeigen wären sechs Auffangregister nötig. Für die Ersparnis bei den Anzeigen handelt man sich mindestens ebenso hohe Kosten für die Register und einen komplizierteren Aufbau ein. Es lohnt also nicht.

Abschließend noch ein Wort zu dem Elektrolytkondensator an der Plusleitung: Die angegebenen 220 µF bilden die untere Grenze. Beim Aufblitzen einer Anzeige fließt ein kräftiger Stromstoß, der sich auf der Plusleitung als Spannungseinbruch bemerkbar machen kann. Ein geladener Kondensator kann Stromstöße mit geringem eigenem Innenwiderstand abgeben, also Spannungseinbrüche reduzieren, und zwar um so besser, je größer seine Kapazität ist. Wenn Sie einen räumlich kleinen Elko mit größerer Kapazität haben, z. B. 470 µF, den Sie an dieser Stelle unterbringen können, so sollten Sie es auch tun.

Die Tastatur

Die Arbeitsweise der Tastatur

Die Tastatur soll die Bitmuster erzeugen, die für die Eingabe von Daten, Adressen und Befehlen nötig sind. Sie übernimmt die Funktion der Schieberegister auf der Daten- und Adreßeingabe, nur mit dem Unterschied, daß die Bedienung viel weniger Mühe macht.

Wenn man sich wünscht, daß ein Tastendruck gleich ein ganzes Byte erzeugen soll, so benötigt man für die 256 Möglichkeiten auch 256 Tasten. Ob die Bedienung eines so umfangreichen Tastenfeldes noch eine Erleichterung der Eingabe ist, sei dahingestellt. Zumindest würde dieses Tastenfeld sehr groß und sehr teuer.

Beschränkt man sich hingegen darauf, daß ein Tastenfeld nur 1 Nibble erzeugt, so reichen 16 Tasten. Dazu kommen dann noch einige Control-Tasten, mit denen sich bestimmte Steuerfunktionen eingeben lassen.

Zu beiden Tastenfeldern gehört freilich wieder eine Menge Software, die finden Sie ausführlich dokumentiert im Programmierband. Fürs erste mag es reichen, das Arbeitsprinzip zu kennen:

Der Prozessor fragt zwischen den Anzeigeyklen immer wieder das Tastenfeld ab, ob eine Taste gedrückt wurde. Wenn nein, wiederholt er den Anzeigeyklus und fragt danach das Tastenfeld erneut ab. Ob eine Taste gedrückt wurde, merkt er am Signal **STROBE** (D7=H). Wenn eine Taste gedrückt wurde, unterbricht er den Anzeigeyklus und fragt, welche Taste gedrückt wurde. War es eine Control-Taste, so führt er die Steuerfunktion aus, war es eine Datentaste, so

liest er das 1. Nibble, das beim Tastendruck entsteht, in ein Arbeitsregister; dort erscheint es als „unteres“ (wertniedriges) Nibble. Durch viermaliges Rotieren wird es nach links verschoben und nimmt die Stelle des „oberen“ Nibbles ein. Der Prozessor merkt sich, daß er erst 1/2 Byte erhalten hat. Mit dem zweiten Tastendruck liest er das 2. Nibble, und es erscheint wieder als unteres Nibble. Diesmal steht es aber an der richtigen Stelle im Byte. Die Rotation nach links entfällt. Das untere Nibble wird zum oberen (dem ersten) addiert. Jetzt ist ein ganzes Byte vorhanden. Es wird durch die Control-Funktion NEXT (Taste NXT) im Speicher abgelegt, und zugleich wird der externe Programmzähler (PC, program counter) um eins weitergezählt.

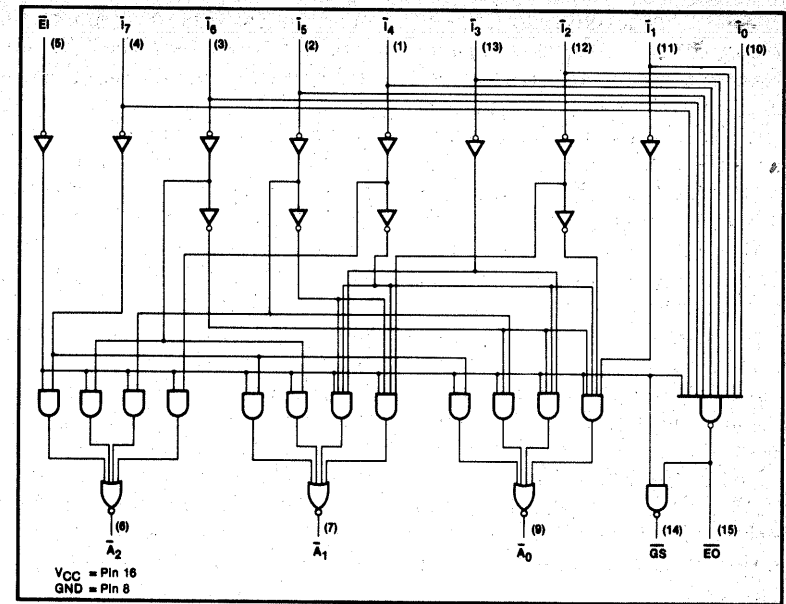
Der externe Programmzähler besteht aus zwei RAM-Speicherstellen. Die 2 Bytes können eine vollständige Adresse aus 4 Nibbles aufnehmen. So wie eben das Laden eines Datenbytes dargestellt wurde, können auch diese beiden Bytes mit einer Adresse geladen werden, und in dieser Adresse wird das eingegebene Datenbyte durch die Funktion NXT abgelegt. Da im Anschluß an den Speichervorgang die Adresse des externen Programmzählers automatisch um eins erhöht wird, brauchen Sie nicht, wie beim Laden des Speichers mit den Schiebeschaltern, bei jedem Datenbyte auch noch zusätzlich eine Adresse einzustellen.

Die Schaltung der Tastatur

Um 1 Nibble zu erzeugen, benötigt man einen **Encoder** „1-aus-16 zu 4 Bit“, also einen Baustein, der beim Aktivieren eines von 16 Eingängen an vier Ausgängen das dazugehörige

duale 4-Bit-Muster erzeugt (Tabelle 6.1). Solch einen Encoder gibt es leider nicht als TTL-Standardbaustein, sondern nur Encoder „1-aus-8 zu 3 Bit“, nämlich den Typ 74(LS)148. Bild 12.5 zeigt den Logikplan der Innenschaltung. Die Anschlüsse \bar{I}_0 bis \bar{I}_7 sind die Eingänge. Die Negation drückt aus, daß ein Eingang, der aktiviert werden soll, auf L zu schalten ist. Die Anschlüsse \bar{A}_0 ($\cong 2^0$), \bar{A}_1 ($\cong 2^1$) und \bar{A}_2 ($\cong 2^2$) sind die Dualausgänge. Auch sie sind aktiv L.

Der Encoder ist als Prioritätsencoder geschaltet: Werden zwei oder mehrere Eingänge zugleich aktiviert, so codiert er den werthöchsten Eingang ohne Berücksichtigung der anderen. Die Prioritätsschaltung wird durch die Inverter an den Eingängen und die UND-Gatter realisiert und macht das Prinzip der Codierung etwas schwer durchschaubar. Die eigentliche Codierung wird in den NOR-Gattern geleistet. Am leichtesten ist das Prinzip am Ausgang \bar{A}_2 zu erkennen. Die vier UND-Gatter sollen nur den Eingang $\bar{E}\bar{I}$ (enable input, engl. Eingangsfreigabe) verknüpfen. Nur, wenn dieser Steuereingang L ist, kann die UND-Bedingung erfüllt werden (Invertierung des Eingangssignals zwischen $\bar{E}\bar{I}$ und den UND-Gattern beachten!). Sie können sich die UND-Gatter erst einmal wegdenken. Wann soll der Ausgang \bar{A}_2 ($\cong 2^2$) aktiv werden? Nur dann, wenn der Eingang \bar{I}_4 ($\cong 100$) ODER \bar{I}_5 ($\cong 101$) ODER \bar{I}_6 ($\cong 110$) ODER \bar{I}_7 ($\cong 111$) angesprochen werden. Da die ODER-Verknüpfung in einem NOR-Gatter geschieht, ist der Ausgang aktiv L (siehe Seite 49 und Funktionstafel 12.6). Außer dem Steuereingang $\bar{E}\bar{I}$, der es ermöglicht, den Baustein in seiner Gesamtheit zu aktivieren ($\bar{E}\bar{I}=L$) oder abzuschalten ($\bar{E}\bar{I}=H$), verfügt



der Encoder noch über zwei Steuereingänge, $\bar{G}\bar{S}$ (group signal, engl. Bausteinsignal) und $\bar{E}\bar{O}$ (enable output, engl. Freigabesignal des Ausgangs, und zwar für weitere Bausteine).

Die Funktionstafel 12.6 zeigt das Verhalten des Bausteins:

Der Ausgang $\bar{G}\bar{S}$ gibt immer dann ein aktiv-L-Signal ab, wenn irgendein Eingang aktiviert (eine Taste gedrückt) wurde. Das $\bar{G}\bar{S}$ -Signal trägt in

Bild 12.5 Logikplan des Encoders (74(LS)148 (nach VALVO-Unterlagen)

Bild 12.9 die Bezeichnung **STROBE** (engl. Steuersignal) und wird als D7 auf den Datenbus gegeben. Da der Bustransceiver alle Signale invertiert, erscheint es als H auf dem Datenbus.

Bild 12.6 Funktionstafel des Encoders 74(LS)148 (nach VALVO-Unterlagen)

FUNCTION TABLE

$\bar{E}\bar{I}$	INPUTS							OUTPUTS					
	\bar{I}_0	\bar{I}_1	\bar{I}_2	\bar{I}_3	\bar{I}_4	\bar{I}_5	\bar{I}_6	\bar{I}_7	$\bar{G}\bar{S}$	\bar{A}_0	\bar{A}_1	\bar{A}_2	$\bar{E}\bar{O}$
H	X	X	X	X	X	X	X	X	H	H	H	H	H
L	H	H	H	H	H	H	H	H	H	H	H	H	L
L	X	X	X	X	X	X	X	L	L	L	L	L	H
L	X	X	X	X	X	X	L	H	L	L	L	L	H
L	X	X	X	X	L	H	H	H	L	L	L	L	H
L	X	X	X	L	H	H	H	H	L	L	L	L	H
L	X	X	L	H	H	H	H	H	L	L	L	L	H
L	L	H	H	H	H	H	H	H	L	H	H	L	H

H = HIGH voltage level
L = LOW voltage level
X = Don't care

zeugt HHH, so daß das ganze Nibble LHHH (auf dem Datenbus $1000 \cong 8_{16}$) entsteht.

Auf die gleiche Art erhalten alle Encodereingänge eine Doppelfunktion (Bild 12.9).

Die Datentasten erzeugen nur ein „unteres“ Nibble, nämlich auf den Leitungen D0 bis D3. Es wäre schade, die Möglichkeit, auch ein „oberes“ Nibble zu erzeugen, einfach ungenutzt zu lassen.

D7 ist ja bereits für $\overline{\text{STROBE}}$ vergeben, es bleiben noch die 3 Bits D4, D5 und D6. Sie werden von einem zweiten Encoder (IC4 in Bild 12.9) belegt. Diese Bits werden dazu genutzt, den Prozessor in verschiedene Unterprogramme springen zu lassen, die immer wieder benötigte Steuerfunktionen ausführen. Die Speicherfunktion NXT wurde bereits erwähnt, die übrigen siehe Seite 234.

Auch der zweite Encoder erzeugt ein $\overline{\text{GS}}$ -Signal. Beide $\overline{\text{GS}}$ -Signale werden über ein NOR-Gatter aus zwei Dioden und einem Widerstand zu einem gemeinsamen $\overline{\text{STROBE}}$ -Signal zusammengefaßt, denn der Prozessor soll ja zunächst erkennen, ob **überhaupt** eine Taste gedrückt wurde. Erkennt er dann Daten im oberen Nibble (das untere Nibble hat dann 0000), weiß er, daß eine Control-Taste gedrückt wurde, und führt die entsprechende Funktion aus. Erkennt er Daten im unteren Nibble (das obere Nibble hat dann 1000), so weiß er, daß eine Datentaste gedrückt wurde. Damit es keine Konflikte mit den Tasten „0“ geben kann, ist der $\overline{\text{I}_0}$ -Eingang des Control-Tasten-Encoders nicht belegt. Das Control-Nibble hat immer einen Wert zwischen 1001 und 1111, den Wert 1000 gibt es nicht. Byte 1000 0000 repräsentiert damit automatisch das Datennibble 0000.

Die von den Encodern erzeugten Bits müssen nun auf den Datenbus gelangen. Dazu gehörten eigentlich invertierende Puffer mit Tri-State-Ausgängen. Eine zugegebenermaßen grobe Methode ist es, die Encoderausgänge einfach mit den A-Ein-/Ausgängen des Bustransceivers (IC3) zu verbinden. „Grob“ aus folgendem Grund: Wenn der Prozessor in die Anzeigen schreibt, sind die A-Ausgänge des Transceivers aktiv (L). In dem Augenblick sind die Ausgänge der Encoder aber H. Während der Schreibzeiten sind also zwei Ausgänge parallelgeschaltet, und das widerspricht der Regel von der Zusammenschaltung mehrerer TTL-Bausteine (siehe Seite 61). Die Transceiverausgänge erzeugen während der Schreibzeiten Kurzschlüsse an den Encoderausgängen. Da die Schreibzeiten aber sehr kurz sind, nämlich nur etwa $2 \mu\text{s}$, und der Transceiver „stärker“ ist als der Encoder, funktioniert die Zusammenschaltung trotzdem, zumal in den Ausgängen der Encoder ja Schutzwiderstände stecken (siehe auch TTL-Ausgang, Bild 2.29). Einige -zig Tastaturen arbeiten seit geraumer Zeit zuverlässig mit dieser Art der Zusammenschaltung. Besser ist es aber, zwischen die Encoderausgänge und den Transceiver je einen Widerstand zwischen 470Ω und $1 \text{ k}\Omega$ zu schalten. Dann ist der Kurzschluß praktisch beseitigt. Die Ausgänge verhalten sich wie ein Wired-NOR (siehe Seite 62), und wenn die A-Anschlüsse des Transceivers zum Lesen der Tastatur als Eingänge geschaltet sind, werden sie, weil sie LS-Eingänge sind, auch über einen Widerstand bis maximal $1 \text{ k}\Omega$ noch zuverlässig auf L geschaltet.

In der Gesamtschaltung der Tastatur und Anzeige haben sich die übrigen

auch billigeren Standardencoder 74148 (also keine LS-Typen) gut bewährt. Bei einer einzigen Tastatur (unter sehr vielen, die mit LS-Encodern aufgebaut waren) zeigte sich, daß die Störimpulse, welche durch das Multiplexen der Anzeigen auf der VCC-Leitung entstehen können, bei einem LS-Baustein Störimpulse auf dem GS-Ausgang ($\overline{\text{STROBE}}$) erzeugten. Damit arbeitete die Tastatur unzuverlässig. Ein Kondensator zwischen $4,7$ und 10 nF von der $\overline{\text{STROBE}}$ -Leitung nach 0 V eingelötet, beruhigte die $\overline{\text{STROBE}}$ -Leitung und sicherte die Funktion der Tastatur. Dieser Kondensator ist im Bestückungsplan der Lötseite (Bild 12.15) mit „hohlen“ Platten zwischen der Leitung B7 und dem Punkt C eingezeichnet. Im Normalfall wird er entbehrlich sein.

Aufbau der Tastatur- und Anzeigeeinheit

Die Tastatur- und Anzeigeeinheit ist der handwerklich schwierigste Teil, in den sich wegen der bisweilen engen Leiterbahnführung leicht Fehler durch Kurzschlüsse einschleichen. **Lassen Sie daher keinen Zwischentest aus!** Bei den Zwischenprüfungen sind eventuelle Fehler leicht zu finden und zu beseitigen, im fertigen Werk ist das sehr viel schwieriger, besonders dann, wenn der Fehler in der kleinen Anzeigenplatte zu suchen ist.

Aufbau der Anzeigenplatte

1. Bestücken Sie zuerst die kleine Anzeigenplatte (Bilder 12.11 und 12.12). Auch wenn Sie Ihren Computer auf

2	Leiterplatte
1	31polige Stiftleiste, Baureihe GdsW
1	kleiner Schiebeschalter
24	Shadow-Digitaster „Mini“, Typ REK
1	Fassung DIL 14 (Wenn auch die Anzeigen in Fassungen gesetzt werden sollen, dann 7 Stück)
3	Fassung DIL 16
1	Fassung DIL 20
24	Silizium-Allzweckdiode, z. B. 1 N 4148
6	pnp-Transistor BC 327
6	High-Efficiency-Anzeige mit gemeinsamer Anode, z. B. hp 5082-7611 oder HDSP 3531
1	74LS02
1	74LS138
2	74148
1	74LS640 oder 74LS640-1
1	mit dem Monitorprogramm programmiertes EPROM 2716
8	Widerstand 27Ω
1	Widerstand 150Ω
14	Widerstand $1 \text{ k}\Omega$
1	Elektrolytkondensator $\geq 220 \mu\text{F}/16 \text{ V}$
1	keramischer Scheibenkondensator $0,1 \mu\text{F}$
1	keramischer Scheibenkondensator $4,7$ bis 10 nF

Bild 12.10 Stückliste für die Tastatur- und Anzeigeeinheit

Experimentierplatten aufbauen, sollten Sie dieses kleine Platinchen mit den High-Efficiency-Anzeigen mitbestellen. Die geringe Ausgabe lohnt sich. Wenn Sie aber doch gern eine Experimentierplatte verwenden wollen, so löten Sie für die Anzeigen Fassungen DIL 14 auf ein Stück Experimentierplatte mit **Lötaugen** (nicht Streifen), und verbinden Sie die gleichnamigen Segmente untereinander mit Kupferlackdraht („Fädel-draht“; der Lack schmilzt in der Lötwärme). Das ist bequemer, als an jedem Segment eine „Doppelleitung“ anzulöten.

2. Setzen Sie die sechs Anzeigen ein. Die Dezimalpunkte zeigen zu den

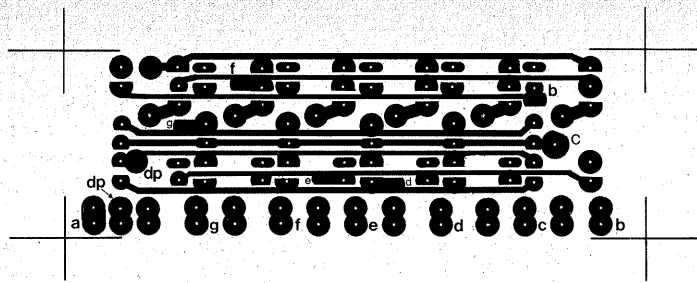


Bild 12.11 Leiterbahnbild der Anzeigenplatte

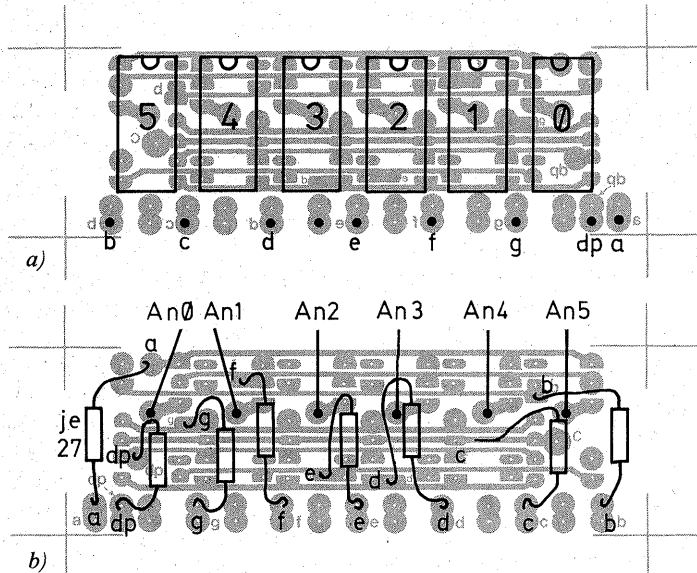


Bild 12.12 a) Bestückungsseite, b) Widerstände auf der Lötseite der Anzeigenplatte

Lötaugen für die Segmentanschlußdrähte.

3. Drehen Sie die Anzeigenplatte vorsichtig um und drücken sie leicht auf eine ebene Fläche (z. B. Tischplatte), so richten sich die Anzeigen in einer Ebene aus. Löten Sie die Anzeigen erst an je einem Stift fest, dann können Sie sie gegebenenfalls noch ausrichten (Zinn verflüssigen, sonst reißt das Lötage ab).

4. Wenn alle Anzeigen in einer Ebene in der Leiterplatte stecken, können Sie sämtliche Stifte anlöten.

5. Löten Sie nun die acht Segmentwiderstände ein. Biegen Sie vorher die Anschlußdrähte in etwa zurecht, schieben Sie Isolierschlauch darüber, und löten Sie einen Widerstand zuerst am unteren großen Lötage an. Richten Sie ihn nun aus und löten ihn auf die entsprechende Leiterbahn.

6. Löten Sie an die Segment- und die Anodenanschlüsse etwa 6 bis 8 cm lange, nicht zu dicke Drähte und schneiden sie stufenweise (analog zu

den Schaltern) ab (Bild 6.13). Dadurch läßt sich die Anzeigeeinheit später in die Grundplatte leichter einfädeln.

7. **Zwischentest:** Lassen Sie diesen **Zwischentest unter keinen Umständen aus**, denn wenn Sie später einen Fehler suchen oder beseitigen wollen, müssen Sie die kleine Platte auslöten. Sie können für diesen Test Ihr Ohmmeter nehmen (Meßbereich $R \times 1$ oder $R \times 10$), sofern die eingebaute Batterie für diesen Meßbereich eine Spannung von mindestens 3 V zur Verfügung stellt (z. B. zwei Babyzellen). Wenn Sie nicht sicher sind, schalten Sie an den Minuspol einer Flachbatterie (4,5 V) einen Widerstand von ca. 470 Ω .

Verbinden Sie den Pluspol der Batterie (er entspricht bei dem als Ohmmeter geschalteten Vielfachinstrument in der Regel dem Minuskabel) mit dem Anodendraht An 0. Tippen Sie nun mit dem freien Ende des Widerstandes die Segmentdrähte der Reihe nach an. Es müssen die zugehörigen Segmente in der Anzeige 0 aufleuchten.

Fehlermöglichkeiten: Wenn zwei Segmente zugleich aufleuchten, besteht zwischen den Segmentleitungen ein Kurzschluß, wahrscheinlich durch Lötzinn (mit Löttauglitze entfernen) oder eine beim Ätzen übriggebliebene Kupfernadel (mit einem Messer nachkratzen).

Wenn ein Segment nicht aufleuchtet, gibt es mehrere Möglichkeiten:

- Kalte Lötstelle (Segmentwiderstand, Anzeige),
- Leiterbahnunterbrechung (z. B. vom Ätzen); Beseitigung durch Überlöten eines feinen Drahtes, siehe Bild 1.23;
- Kurzschluß mit dem Anodenanschluß (Pin 3 oder 14); gefährdet

sind die Segmente a und f durch Pin 14 oder b und g durch Pin 3.

Wenn die Anzeige 0 richtig reagiert, klemmen Sie den Pluspol der Batterie an den Anodendraht An 1, und überprüfen Sie alle Segmente in gleicher Weise, ob sie richtig aufleuchten. Ebenso verfahren Sie mit den Anzeigen 2 bis 5.

8. Wenn Sie **ganz sicher** sind, daß alle Anzeigen bei Ansteuerung (Plus an den zugehörigen An-Anschluß, Minus an den jeweiligen Segmentanschluß) richtig aufleuchten, sprühen Sie die Lötseite der kleinen Platine mit Plastiklack ein. Nun können Sie die Grundplatte bestücken.

Aufbau der Grundplatte

1. Bohren Sie alle Löcher für die Taster (auch für deren Führungsstifte) mit 1,3 mm \varnothing .

Löten Sie die sechs Drahtbrücken ein, die auf der **Bestückungsseite** liegen (A-A, A'-A', A''-A'', B-B, C-C, D-D). Löten Sie alle Taster ein. Achten Sie dabei darauf, daß deren Gelenk zur Anzeigenseite hin gerichtet ist. Die Taster müssen fest auf der Platine aufsitzen (Bild 1.1).

2. Löten Sie die acht Tasterdioden ein, welche die Taster „8“ bis „F“ an die Taster „0“ bis „7“ koppeln (Bild 12.16). Deren Anschlußdrähte sollten Sie isolieren, am besten mit Rüsenschlauch (Leinengewebeschlach). Die Abstände zwischen den Lötaugen für die acht Dioden betragen 30 mm. Da die Diodenkörper 3 mm lang sind, benötigen Sie knapp 27 mm lange Isolierstücke:

- Markieren Sie sich auf einem Brettchen o. ä. mit dünnen Strichen einen Abstand von 26 mm.
- Schneiden Sie mit einem Abbrech-

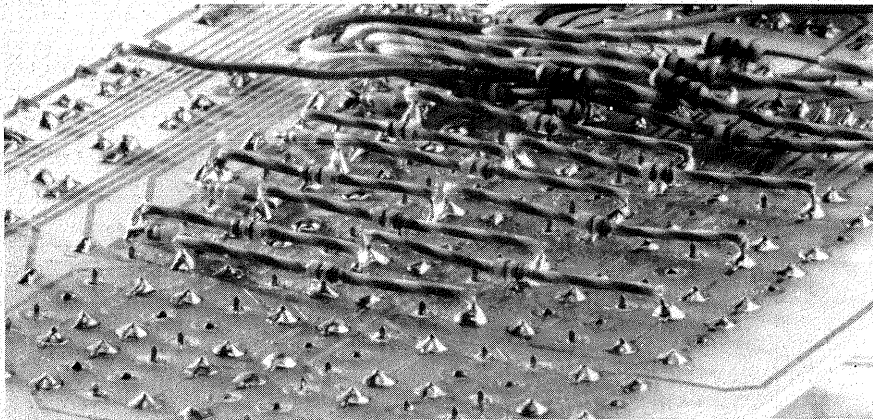


Bild 12.16 Lage der Dioden unter den Tastern

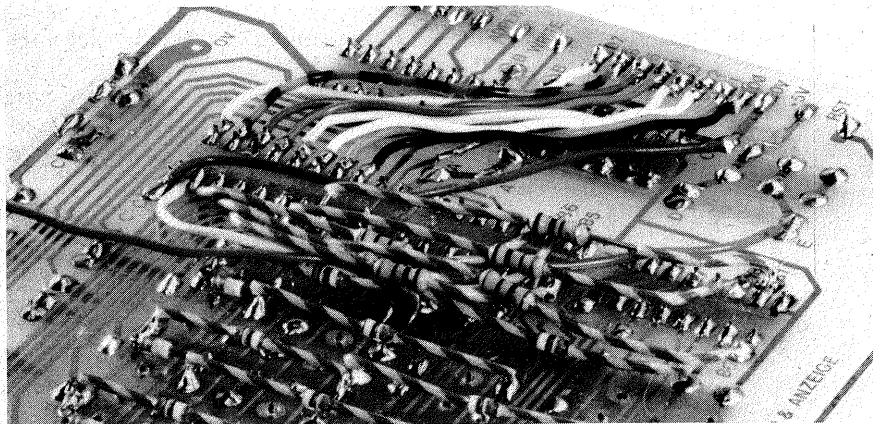


Bild 12.17 Lage der Widerstände zwischen den Encodern und dem Bustransceiver

auf die eingebaute Batterie. In der Regel sind Multimeter so geschaltet, daß bei der Widerstandsmessung der Minuspol der eingebauten Batterie an der Plusklemme des Multimeters erscheint, der Pluspol an der Minusklemme).

Mit der Plusleitung kontrollieren Sie den Schaltvorgang an den zu den Ta-

stern gehörigen IC-Anschlüssen.

Klemmen Sie eine Stecknadel an die Meßleitung, und tippen Sie damit den IC-Kontakt an. Bei offenem Taster muß das Ohmmeter $\infty \Omega$ anzeigen, bei geschlossenem Taster Durchgang. Bei den Control-Tastern und den Datentastern „0“ bis „7“ zeigt es dann (nahe) 0Ω an, bei den über Dioden geschalteten Tastern „8“ bis „F“ den bei Siliziumdioden üblichen geringeren Zeigerausschlag.

4. Die Control-Taster

Zu den Control-Tastern „C1“ bis „C7“ gehören die Anschlüsse 11, 12, 13, 1, 2, 3 und 4 von IC4:

C1 (NXT):	Pin 11
C2 (GOTO):	Pin 12
C3 (PC):	Pin 13
C4 (RUN):	Pin 1
C5 (MON):	Pin 2
C6 (CMD):	Pin 3
C7 (F):	Pin 4

5. Die Datentaster

Zu den Datentastern „0“ bis „7“ gehören unmittelbar die Anschlüsse 10, 11, 12, 13, 1, 2, 3 und 4 von IC5:

Taster 0:	Pin 10
1:	Pin 11
2:	Pin 12
3:	Pin 13
4:	Pin 1
5:	Pin 2
6:	Pin 3
7:	Pin 4

Da die Taster „8“ bis „F“ die entsprechenden IC-Anschlüsse über Dioden schalten, wird das Ohmmeter beim Drücken eines dieser Taster wegen der Diodenschwellenspannung einen gewissen Widerstandswert (statt 0Ω) anzeigen:

Taster 8:	Pin 10
9:	Pin 11
A:	Pin 12
B:	Pin 13
C:	Pin 1
D:	Pin 2
E:	Pin 3
F:	Pin 4

Setzen Sie nun das Prüfkabel an den Meßpunkt MP1 (Leiterbahn B3). Beim Drücken eines jeden Tasters „8“ bis „F“ muß das Ohmmeter Durchgang anzeigen, beim Drücken der Taster „0“ bis „7“ dagegen $\infty \Omega$.

6. Nun löten Sie die acht Drahtbrücken B0 bis B7 auf der Unterseite der Platine ein. Es sind die Drähte, die an

die Anschlüsse 2 bis 9 von IC3 führen.

Achtung: Der Übersichtlichkeit halber sind die Verbindungen B0-B0, B1-B1, B2-B2 sowie B4-B4, B5-B5 und B6-B6 als einfache Leitungen gezeichnet. In Wirklichkeit bestehen sie aus Widerständen 470Ω bis $1 k\Omega$, über deren Anschlußdrähte Isolierschlauch gezogen wurde (Bild 12.17). Löten Sie zuerst alle Drähte bzw. Widerstände bei IC3 an.

Beim Anlöten der Drähte an die übrigen Punkte beginnen Sie mit B3. Diese Drahtbrücke darf wegen ihrer Kürze eine kleine Schlaufe bilden. Achten Sie besonders auf die Umgebung des Lötunktes B7 (bei IC4) und den Punkt „A“ zwischen IC3 und IC4. Dort stellten sich wegen der engen Leiterbahnführung bei Schülerarbeiten gern Kurzschlüsse ein.

Außerdem ist die Drahtbrücke vom Anschluß 0 V an Punkt C einzulöten, damit die ICs mit Spannung versorgt werden.

- Löten Sie Anschlußdrähte an 0 V und +5 V an, damit Sie die Platine mit Spannung versorgen können.
- Setzen Sie die beiden Tastaturencoder IC4 und IC5 (74148) ein.
- Messen Sie, ob die Tastaturencoder richtig arbeiten. Sie können die Pegel mit der „Stecknadleitung“ an den noch offenen Anschlüssen der Fassung für IC3 messen. Dabei haben Sie sofort eine Kontrolle darüber, ob die Pegel beim Bustransceiver IC3 (74LS640) richtig ankommen. Schalten Sie Ihr Multimeter auf Spannungsmessung um (Meßbereich $5 \dots 10 V$).
- Versorgen Sie die Platine mit +5 V.
- Messen Sie, ob zwischen den Anschlüssen 8 (-) und 16 (+) der ICs 4 und 5 die Versorgungsspannung steht.

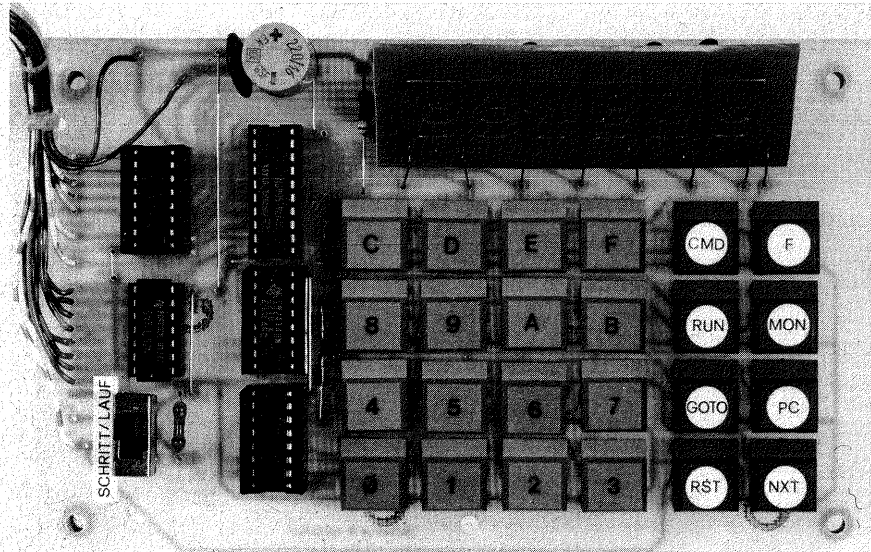


Bild 12.18 Die fertige Tastatur auf der Platine

7. Messen Sie STROBE am Meßpunkt MP2 (die beiden Anoden der Dioden neben IC4). Beim Drücken eines jeden Control- oder Datentasters muß STROBE von H auf L springen, beim Loslassen wieder von L auf H.

STROBE muß auch an Pin 9 von IC3 zu messen sein.

8. Messen Sie B3 an Pin 5 von IC3. Dort müssen Sie in Ruhestellung der Taster H-Signal messen können, das beim Drücken eines jeden Tasters „8“ bis „F“ auf L (ca. 0,6 V) sinkt. Durch Drücken der Taster „0“ bis „7“ ist es nicht zu beeinflussen (Dauer-H).

9. Prüfen Sie die Control-Pegel an B4, B5 und B6:

Taster	Pin 8 A2=B6	Pin 7 A1=B5	Pin 6 A0=B4
C1:	H	H	L
C2:	H	L	H
C3:	H	L	L
C4:	L	H	H
C5:	L	H	L
C6:	L	L	H
C7:	L	L	L

Wenn sich diese Tabelle **nicht** einstellt, liegt wahrscheinlich ein Kurzschluß an den Eingängen des Encoders vor. Denken Sie daran, daß der 74148 ein Prioritätsencoder ist. Werden z. B. zwei Tasten gleichzeitig gedrückt (also zwei Eingänge mit L beschaltet), müßte der Encoder in einen Verhaltenskonflikt geraten, denn er kann ja nicht zugleich zwei Bitmuster anbieten. Daher ist er intern so geschaltet, daß er automatisch den höheren Eingang decodiert und den anderen ignoriert. Beispiel: Zwischen

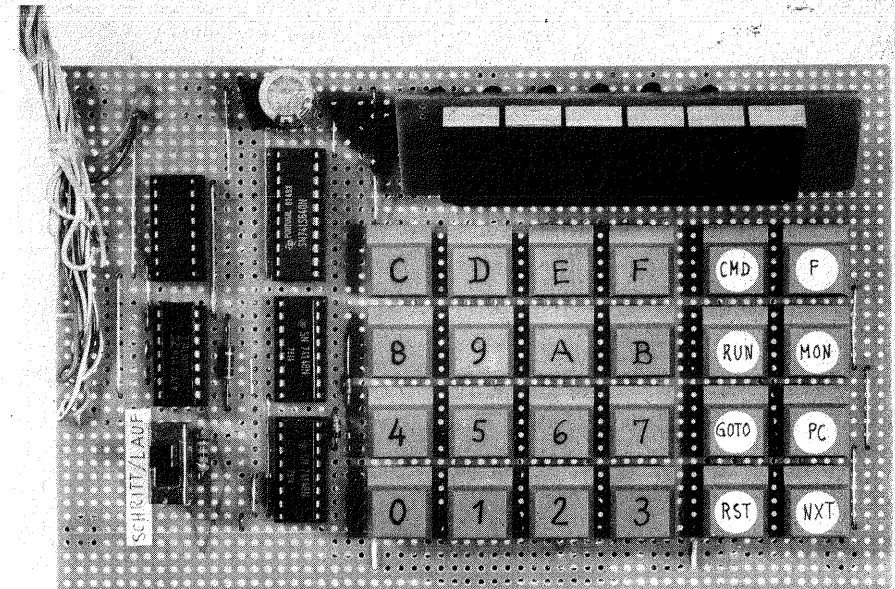


Bild 12.19 Die fertige Tastatur auf einer Experimentierplatte

den Eingängen $\bar{11}$ und $\bar{12}$ war ein Kurzschluß. Beim Drücken des Tasters „C1“ ergab sich das Bitmuster, das zum Taster „C2“ gehörte; statt H-H-L erschien H-L-H. An solchen Bitmusterfehlern können Sie erkennen, wo Sie den Fehler zu suchen haben. Wenn Sie allerdings eine der Leitungen, die an IC3 führen, verwechselt haben sollten, hilft das Nachdenken über das Verhalten eines Prioritätsencoders nicht weiter. Beziehen Sie daher diese Fehlermöglichkeit in Ihre (hoffentlich überflüssigen) Suchaktionen ein.

10. Prüfen Sie nun die Datentaster. Das Bit B3 haben Sie bereits geprüft (vgl. Seite 232), so daß Sie jetzt jeweils nur die Bits B0 bis B2 zu prüfen brau-

chen. Je zwei zusammengehörige Taster, „0“ und „8“, „1“ und „9“ usw. ergeben das gleiche Muster in B0 bis B2:

Taster	Pin 4 A2=B2	Pin 3 A1=B1	Pin 2 A0=B0
0/8:	H	H	H
1/9:	H	H	L
2/A:	H	L	H
3/B:	H	L	L
4/C:	L	H	H
5/D:	L	H	L
6/E:	L	L	H
7/F:	L	L	L

11. Wenn soweit alles richtig funktioniert, können Sie die restlichen Brücken einlöten (E-E und C) die acht Datenleitungen) sowie die Stifteleiste anschließen (siehe Seite 90). Ehe Sie die geprüfte Anzeigeneinheit einsetzen, vergewissern Sie sich, daß die Dioden vor den Basen der Treibertransistoren richtig gepolt sind.

Fädeln Sie zuerst die Segmentdrähte, danach die Anodendrähte ein. Löten Sie erst die Segmentanschlüsse dp und b an, richten Sie die Anzeigenplatte aus, und biegen Sie sie in einen Winkel von etwa 45° zur Grundplatte. Danach können Sie alle Drähte anlöten.

Wenn Sie nun die restlichen ICs einsetzen und die Speicherkarte so schalten, daß das EPROM mit dem Monitor bei Adresse 0000 beginnt, muß sich Ihr Computer nach dem Einschalten mit „HALLO.“ melden.

Die Bedienung der Tastatur

Mit den **Datentasten** geben Sie jeweils Adressen oder Daten-(Befehls-)bytes ein. Sie geben 1 Byte in 2 Nibbles ein, wie es der hexadezimalen Schreibweise entspricht, zuerst das obere Nibble, danach das untere. Z. B. A7: zuerst „A“, dann „7“, abschließend NXT, damit das Byte A7 in den Speicher übernommen wird.

Der frei verfügbare RAM-Bereich beginnt bei der Adresse 0900 und endet, wenn Sie nur einen einzigen RAM-Baustein 6116 eingesetzt haben, mit der Adresse 0FFF. Sie haben in der kleinsten Ausbaustufe 1791₁₀ Bytes zur freien Verfügung.

Die **Control-Tasten** haben folgende Funktionen:

Taste	Anzeige	Funktion
RST	HALLO.	Die gerade laufende Funktion wird unterbrochen, und der Prozessor wird auf die Adresse 0000, die Startadresse des Monitorprogramms, zurückgesetzt.
MON	HALLO.	Diese Taste kann vom Programm abgefragt werden (Tastenwert 50 ₁₆) und setzt den Prozessor auf die Adresse 0000, die Startadresse des Monitorprogramms, zurück.
PC	PCnnn	Auf der Anzeige erscheinen die Buchstaben PC (für program counter) und die Adresse des externen Programmzählers. Es kann dann eine neue Adresse eingegeben werden. Sie muß vollständig, also vierstellig hexadezimal eingegeben werden. Bei den drei ersten Stellen erscheint als „Merker“ jeweils zu der eingegebenen Ziffer der Dezimalpunkt. Wenn die vierte Stelle eingegeben ist, verschwinden die Punkte, und nun ist eine vollständige Adresse eingegeben, die der Prozessor übernehmen kann. Danach bestehen folgende Möglichkeiten: - NEXT: Sprung auf den Inhalt der PC-Adresse, - MON: Rücksprung auf die Startadresse des Monitors, - GOTO: Sprung zur PC-Adresse und Abarbeitung des dort beginnenden Programms.

N(E)XT nnnn XX

Auf den vier linken Anzeigen erscheint die Adresse des externen Programmzählers, auf den beiden rechten Anzeigen der unter dieser Adresse gespeicherte Hex-Wert. Dieser Wert kann mit den Datentasten (Hex-Tastatur) überschrieben werden. Es muß immer ein vollständiges Byte eingegeben werden. Mit der Eingabe des oberen Nibbles erscheint zusätzlich ein Dezimalpunkt als Merker, mit der Eingabe des unteren Nibbles verschwindet dieser Punkt. Dann ist ein vollständiges Byte eingegeben, das übernommen werden kann. Danach bestehen folgende Möglichkeiten:

- NEXT: Zugleich wird der externe Programmzähler um eins weitergezählt, so daß das nächste Byte wie beschrieben eingegeben werden kann.
- CMD: Der Inhalt der angezeigten Adresse wird gelöscht. Alle Bytes in den darüber liegenden Adressen bis 3FFF rutschen zur nächst niedrigeren Adresse (Delete-Funktion; delete, lat./engl. löschen).
- F (Control-Taste Function): Von der angezeigten Speicherstelle bis zur Adresse 3FFE werden alle Bytes um einen Speicherplatz nach oben verschoben (Insert-Funktion; insert, lat./engl. einfügen). Die nunmehr freie Speicherstelle hat den Wert 00 und kann mit einem anderen Wert geladen werden.
 Achtung: Nach Programmänderung durch Delete oder Insert müssen die Zieladressen der Sprungbefehle gegebenenfalls angepaßt werden!
- MON: Rücksprung auf die Startadresse des Monitorprogramms.

RUN leer Sprung zur Adresse 0900 und Abarbeitung des dort beginnenden Programms. Durch die Funktion RUN werden alle Variablen auf 0 gesetzt.

GOTO leer Sprung zur im externen Programmzähler gespeicherten Adresse und Abarbeitung des dort beginnenden Programms. Bei der Funktion GOTO bleiben alle Variablen erhalten.

CMD CMD= Aufruf der COMMAND-Funktion. Hier stehen folgende Funktionen zur Verfügung:

- A: Ask for Register Contents. Abfrage der Registerinhalte nach einem Programmabbruch über Break-Point. Beim Drücken einer beliebigen Datentaste erscheinen der Reihe nach die Registerinhalte:

R0 = Register 0	R5 = Register 2' (Bank 1)
R1 = Register 1 (Bank 0)	R6 = Register 3' (Bank 1)
R2 = Register 2 (Bank 0)	R7 = PSL
R3 = Register 3 (Bank 0)	R8 = PSU
R4 = Register 1' (Bank 1)	

- B: Break-Point. Eingabe einer vierstelligen Hex-Adresse, an der ein Break-Point im Programm gesetzt werden soll. Die Adresse muß auf das 1. Byte eines Befehls zeigen und muß im RAM-Bereich liegen.
- C: Clear Break-Point. Hiermit kann ein einmal gesetzter Break-Point vor Programmablauf wieder gelöscht werden, wenn z. B. das Setzen irrtümlich erfolgte oder eine falsche Adresse gewählt wurde. Beim Erreichen eines Break-Points im Programm wird dieser automatisch gelöscht, so daß das Programm beim nächsten Durchlauf nicht angehalten wird.
- D: Dump to Tape. Abspeichern von Programmen oder Daten auf Tonbandkassette. Es werden jeweils Blöcke von 256 Bytes gespeichert. Nach der Eingabe von „D“ erscheint in der Anzeige ein „A“ (Anfang). Hier wird die Anfangsadresse des ersten zu speichernden Blocks (2stellig Hex) eingegeben, z. B. für die Anfangsadresse 0900 der Wert 09. Danach erscheint in der Anzeige ein „E“ (Ende). Hier wird die Endadresse eingegeben, bis zu der gespeichert werden soll, z. B. Endadresse 0F00 \triangleq 0F.
Danach Starten der Aufnahmefunktion des Kassettenrecorders und Aussteuerung mit dem anfangs vorhandenen Dauerton. Die besten Ergebnisse werden mit leichter Übersteuerung erreicht. Zum Starten der Datenübertragung Druck auf eine beliebige Datentaste. Während des Speichervorgangs erscheint die Adresse des gerade gesendeten Blocks auf der Anzeige. Nach beendetem Speichervorgang erscheint HALLO.
- E: Examine Tape File. Die Routine dient zur Überprüfung, ob eine Datei einwandfrei auf die Kassette überspielt wurde. Nach der Eingabe von „E“ erscheint auf der Anzeige ein „A“. Nun wird die Anfangsadresse eingegeben, an der das zu prüfende Programm im Speicher steht (2stellig Hex). Entspricht das Programm auf der Kassette dem Speicherinhalt, so antwortet der Monitor mit „HALLO.“ Andernfalls erscheint auf der Anzeige „ERROR2“. Der Speichervorgang ist dann zu wiederholen. Es empfiehlt sich, an jeden DUMP-Vorgang ein EXAMINE anzufügen, um sicher zu sein, daß das Programm fehlerfrei auf der Kassette vorhanden ist.
- F: Fetch from Tape. Laden von Programmen oder Daten von der Tonbandkassette in den Speicher. Drücken Sie zuerst CMD, dann starten Sie die Wiedergabefunktion Ihres Kassettenrecorders, und wenn Sie den hohen Ton hören, drücken Sie schnell die Datentaste „F“, ehe das „Klingeln“ des Programms beginnt.

Das ordnungsgemäße Laden kann über die LED auf der Porteinheit überwacht werden. Während des Ladevorgangs erscheint die Adresse des gerade gelesenen Blocks auf der Anzeige. Die gelesenen Adressen und Daten werden aufaddiert und mit einer auf dem Band gespeicherten Prüfwahl verglichen. Bei Abweichungen erscheint auf der Anzeige „ERROR“. Der Ladevorgang ist dann zu wiederholen. Bei korrekt abgeschlossenem Ladevorgang erscheint auf der Anzeige „HALLO.“

Bei den Arithmetikprogrammen (14.22 bis 14.24) haben die Tasten folgende zusätzlichen Bedeutungen:

RUN: +
GOTO: -
PC: x
NXT: :

Beispiel für die Eingabe eines Programms

Das „=“ hat keine eigene Taste. Das Ergebnis erscheint sofort nach dem Loslassen der letzten eingegebenen Ziffer.

Sie wollen die Zahlen 02_{16} und 05_{16} addieren; das Ergebnis soll sowohl in die Port-LEDs (Adresse 09_{16}) als auch in die Speicheradresse 0960_{16} geladen werden.

Zeile	Taste	Anzeige	Bemerkungen
1	MON oder RST	HALLO.	Sprung ins Monitorprogramm
2	PC	PCnnnn	Zufälliger Inhalt des externen Programmzählers erscheint
3	0	PC0.nnn	Anfangsadresse 0900 eingeben; der Punkt zeigt an,
4	9	PC0.9.nn	welche Stelle Sie eingegeben haben. Mit der Eingabe
5	0	PC0.9.0.n	der vierten Stelle ist eine vollständige Adresse
6	0	PC0900	geladen. Die Punkte verschwinden. Nun erst kann der externe Programmzähler die Adresse übernehmen
7	NXT	0900XX	Zufälliger Speicherinhalt der Adresse 0900
8	0	09000.X	Nun erfolgt die Eingabe 04 02
9	4	090004	(LODI, R0, 02)
10	NXT	0901XX	Jede Dateneingabe wird mit NXT abgeschlossen. Dabei geht der externe Programmzähler einen Schritt weiter, und in den beiden letzten Digits erscheint der zufällige Inhalt dieser Speicherstelle. Sie können unmittelbar das folgende Byte laden.
11	0	09010.X	
12	2	090102	
13	NXT	0902XX	

14	8	09028.X	Addierbefehl ADDI, R0, 05 eingeben
15	4	090284	
16	NXT	0903XX	
17	0	09030.X	
18	5	090305	
19	NXT		
20	D	0904D.X	Schreibbefehl für \$ 09 eingegeben, WRTE, R0, 09
21	4	0904D4	
22	NXT	0905XX	
23	0	09050.X	
24	9	090509	
25	NXT	0906XX	
26	C	0906C.X	Speicherbefehl für Adresse 0960 eingeben; STRA, R0, 0960
27	C	0906CC	
28	NXT	0907XX	
29	0	09070.X	
30	9	090709	
31	NXT	0908XX	
32	6	09086.X	
33	0	090860	
34	NXT	0909XX	
35	4	09094.X	HALT-Befehl eingeben
36	0	090940	
37	NXT	090AXX	Auch die letzte Eingabe muß mit NXT abgeschlossen werden.
38	MON	HALLO.	Rücksprung ins Monitorprogramm
39	RUN	leer	Wenn der Prozessor ein Programm abarbeitet, springt er aus der Anzeigeroutine, daher werden die Anzeigen dunkel. Das Ergebnis erscheint in diesem Fall in den Port-LEDs. Aus dem WAIT-Zustand kann der Prozessor nur durch RESET befreit werden. Danach kann man das in der Adresse 0960 abgelegte Ergebnis in die Anzeige bringen.
40	RST	HALLO.	Das Ergebnis verschwindet durch RESET aus den Port-LEDs. Der Prozessor springt ins Monitorprogramm zurück. Nun muß die Adresse 0960 geladen werden.
41	PC	PCnnn	
42	0	PC0.nnn	
43	9	PC0.9.nn	
44	6	PC0.9.6.n	
45	0	PC0960	Die Adresse ist jetzt vollständig und kann mit NXT übernommen werden.
46	NXT	0960 07	Adresse und Inhalt (Ergebnis)

Wenn das Ergebnis dieses Programmes stimmt, so ist es der pure Zufall, weil in diesem Programm auf das Initialisieren des Programm-Status-Worts verzichtet wurde. Falls als Ergebnis 08 erscheint, wurde ein zufällig vorhandener Übertrag dazugerechnet.

Beim Rückholen des Prozessors aus dem WAIT-Zustand verschwand die Anzeige in den Port-LEDs. Jetzt ist es empfehlenswert, die RESET-Leitung auf dem Portbaustein zu entfernen oder abschaltbar zu machen. Lötungen zur Montage eines Schalters sind auf der Portplatine vorhanden.

Falls Sie sich bei der Eingabe eines Bytes geirrt haben und dies vor der Betätigung von NXT bemerken, so geben Sie das richtige Byte sogleich anschließend ein und betätigen NXT, wenn das Byte richtig in der Anzeige steht. Bemerken Sie Ihren Irrtum erst nach der Betätigung von NXT, so gehen Sie über MON in die Monitorroutine zurück, drücken PC und laden die zu korrigierende Adresse, drücken dann NXT und laden das richtige Byte. Den Ladevorgang schließen Sie mit NXT ab.

Sie sollten jedes eingegebene Programm „nachlesen“. Dazu laden Sie die Anfangsadresse und betätigen NXT. Dann erscheinen die erste Adresse sowie ihr Inhalt in der Anzeige. Danach drücken sie wieder NXT, und die nächste Adresse erscheint samt Inhalt. Durch Weitertasten mit NXT können Sie das gesamte Programm schnell überprüfen.

Anschließend wird das folgende Byte in das Arbeitsregister geladen, und so geht es weiter, bis das ganze Programm als Tonfolge gespeichert ist. Zwischen den Bytes wird jeweils noch ein Start- bzw. Stop-Bit übertragen, damit der Prozessor beim Lesen weiß, wann ein Byte beginnt und wann es endet.

2. Ein Byte wird gelesen.

Wenn man das Tonband wieder abspielt, empfängt man nur tiefe (1200 Hz) oder hohe (2400 Hz) Töne. Mit den Tonspannungen kann der Prozessor noch nichts anfangen. Die Töne müssen in L- bzw. H-Pegel umgeformt werden. Das geschieht im „Empfänger“. Die kleinen Tonspannungen werden erst verstärkt und begrenzt (IC1 in Bild 13.1) und dann auf ein digitales Filter geleitet (IC2 und IC3), dessen Ausgang (SENSE) bei einer empfangenen Tonspannung von 1200 Hz auf L schaltet, bei einer Frequenz von 2400 Hz dagegen auf H (siehe auch „High-Fenster“, Bild 13.7).

Der Ausgang des Filters steuert den SENSE-Eingang des Mikroprozessors. Das SENSE-Bit (Bit 7 im PSU) gibt immer den augenblicklichen Zustand am SENSE-Eingang wieder. Das SENSE-Bit kann direkt als Bit 7 in das Arbeitsregister übertragen werden. Anschließend wird es durch Rotation nach rechts um eine Stelle verschoben, so daß der Platz (Bit 7) für das folgende Bit wieder frei ist. Nach achtmaligem Empfang eines Bits und achtmaliger Rotation nach rechts steht das Byte wieder im Arbeitsregister und kann in den Speicher transportiert werden.

Die Zuverlässigkeit des Kassetteninterfaces hängt nicht zuletzt von der Qualität der frequenzbestimmenden

Bauelemente ab. Die Kondensatoren am Oszillator (C_4) und an den beiden Mono-Flops (C_2, C_3, C_1) sollten unbedingt Folienkondensatoren sein. Keramische Kondensatoren haben nicht nur eine viel zu große Fertigungstoleranz, sondern meist auch einen sehr großen Temperaturbeiwert. Die höchstzulässige Toleranz bei den genannten Kondensatoren beträgt $\pm 10\%$. Die Trimmerwiderstände R_1/R_2 sollten nach Möglichkeit Spindeltrimmer sein.

Auch der Kassettenrecorder verdient einige Aufmerksamkeit: Er muß nicht gerade zur HiFi-Spitzenklasse zählen, ein billiges Modell reicht auch, aber **es sollte über die Möglichkeit der Handaussteuerung verfügen**. Damit hat es folgende Bewandnis: Die Tonspannungen mit den verschiedenen Frequenzen gelangen zwar mit der gleichen Amplitude auf den Eingang des Kassettenrecorders, werden aber im Aufnahmeverstärker nicht linear verstärkt. Die hohen Töne verhalten sich wie etwas schwächere Eingangssignale. Die Regelautomatik versucht, diese Schwankungen auszugleichen, und kaum beginnt sie, sich dem Wechsel anzupassen, ändert sich die Eingangsfrequenz, und die Regelautomatik beginnt, den Regelvorgang umzukehren. So kann es unter (sehr) ungünstigen Verhältnissen zu Regelschwingungen kommen, die Fehler in der Datenaufzeichnung verursachen. Bei einem Recorder mit der Möglichkeit der Handaussteuerung wird die Regelautomatik abgeschaltet, und dann erzielt man die besten Ergebnisse, wenn man den Recorder leicht übersteuert.

Leider sind nahezu alle Billigeräte nur mit einer Regelautomatik ausgestattet. Das heißt aber nicht, daß Sie sie deshalb nicht benutzen können.

Sie können sie gut verwenden, wenn Sie bereit sind, ein Programm eventuell zweimal auf die Kassette zu überspielen. Durch die EXAMINE-Routine (siehe Seite 236) können Sie sich vergewissern, daß Sie das Programm richtig gespeichert haben.

Eine weitere nützliche Einrichtung ist ein Bandzählwerk, so daß Sie mehrere Programme auf einer Kassette leicht wiederfinden, wenn Sie dazwischen ausreichend Platz lassen.

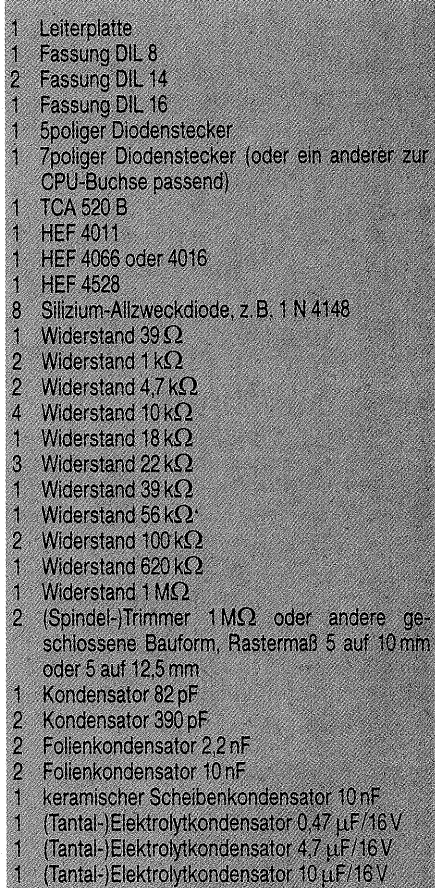
Normale Eisenoxiddassetten haben sich in der Datenaufzeichnung gut bewährt. Es gibt spezielle Kassetten mit einer Spieldauer von 10 oder 15 Minuten. Sie erleichtern sich die Ordnung in Ihrem Programmarchiv mit diesen kleinen Kassetten erheblich und können sich viel Sucharbeit ersparen.

Aufbau und Inbetriebnahme des Kassetteninterfaces

Bild 13.3 zeigt die Leiterbahnen der Platine, Bild 13.4 die Bestückungsseite. Sie beginnen die Bestückung am besten mit dem Oszillator (IC4 und umgebende Bauelemente). Die Platine ist für Trimmerwiderstände verschiedener Rastermaße eingerichtet. Es ist nicht wichtig, ob Schleifer oder Endanschluß eines Trimmers mit dem entsprechenden IC-Anschluß zusammengeschaltet werden. Es kommt lediglich darauf an, daß jeweils ein Endanschluß eines Trimmers frei bleibt.

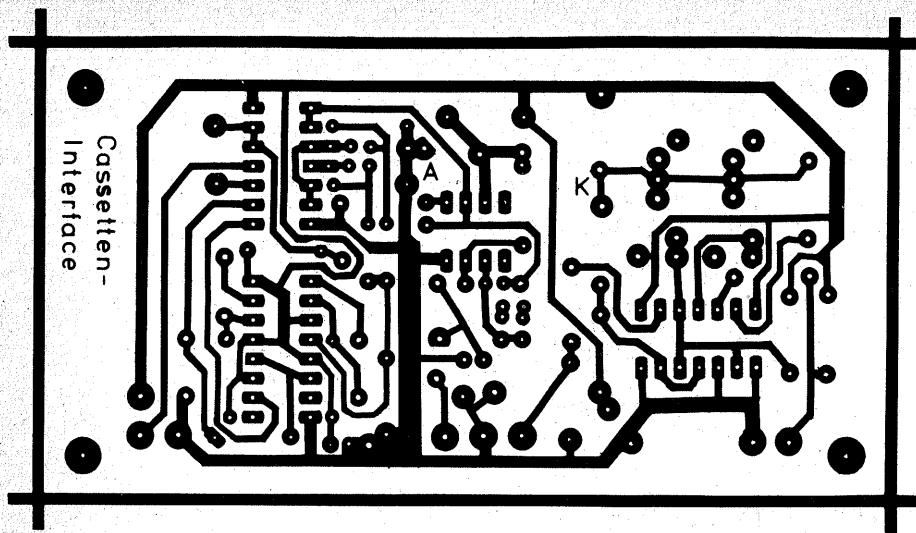
Falls Ihre Bastelkasse es hergibt, sollten Sie unbedingt Spindeltrimmer verwenden. Sofern Sie normale Trim-

mer mit einem Drehwinkel von 270° einsetzen, müssen diese wenigstens eine geschlossene Bauform und von guter Qualität sein (Bild 13.5). Die Frequenzeinstellung des Oszillators ist mit Spindeltrimmern leicht, mit normalen Trimmern dagegen recht knifflig. Die Diode D_1 wird auf der Leiterbahnseite der Platine eingelötet (Anode an A, Kathode an K).



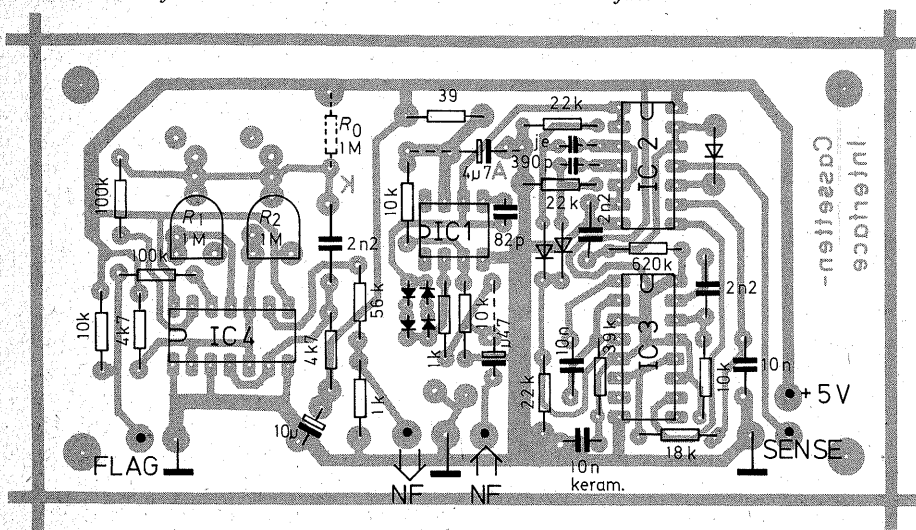
1	Leiterplatte
1	Fassung DIL 8
2	Fassung DIL 14
1	Fassung DIL 16
1	5poliger Diodenstecker
1	7poliger Diodenstecker (oder ein anderer zur CPU-Buchse passend)
1	TCA 520 B
1	HEF 4011
1	HEF 4066 oder 4016
1	HEF 4528
8	Silizium-Allzweckdiode, z. B. 1 N 4148
1	Widerstand 39 Ω
2	Widerstand 1 k Ω
2	Widerstand 4,7 k Ω
4	Widerstand 10 k Ω
1	Widerstand 18 k Ω
3	Widerstand 22 k Ω
1	Widerstand 39 k Ω
1	Widerstand 56 k Ω
2	Widerstand 100 k Ω
1	Widerstand 620 k Ω
1	Widerstand 1 M Ω
2	(Spindel-)Trimmer 1 M Ω oder andere geschlossene Bauform, Rastermaß 5 auf 10 mm oder 5 auf 12,5 mm
1	Kondensator 82 pF
2	Kondensator 390 pF
2	Folienkondensator 2,2 nF
2	Folienkondensator 10 nF
1	keramischer Scheibenkondensator 10 nF
1	(Tantal-)Elektrolytkondensator 0,47 μ F/16 V
1	(Tantal-)Elektrolytkondensator 4,7 μ F/16 V
1	(Tantal-)Elektrolytkondensator 10 μ F/16 V

Bild 13.2 Stückliste für das Kassetteninterface



Oben:
Bild 13.3 Leiterbahnbild für das Kassetteninterface

Unten:
Bild 13.4 Bestückungsplan für das Kassetteninterface



Ehe Sie die Platine weiterbestücken, prüfen Sie den Oszillator. Am besten wäre es, Sie hätten einen Oszillographen und einen Frequenzzähler. Sie schließen zuerst den Oszillographen am NF-Ausgang an. Vor dem Ein-

schalten der Betriebsspannung bringen Sie beide Trimmer in eine mittlere Stellung. Außerdem verbinden Sie den FLAG-Eingang über ein kurzes Prüfkabel mit 0 V. Wenn Sie nun die Betriebsspannung einschalten, muß

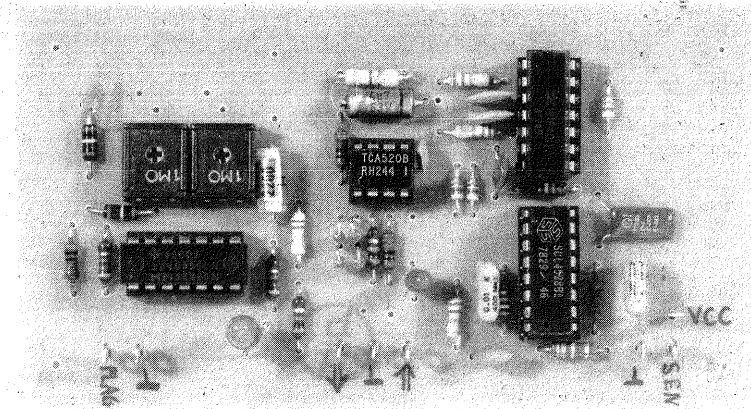


Bild 13.5 Das fertige Kassetteninterface auf der Platine

Ihnen der Oszillograph eine Rechteckspannung in der Größenordnung von 100 mV anzeigen. Nun schließen Sie den Frequenzzähler an. Dabei kann es – je nach Eingangswiderstand des Zählers – vorkommen, daß die Rechtecksignale fürchterlich verbogen mit abgeschrägten Dächern aussehen. Das macht aber nichts. Sollte der Frequenzzähler nichts anzeigen, so ist die Eingangsspannung für ihn zu niedrig. Immerhin liegt ja ein hochohmiger Spannungsteiler von 56 k Ω : 1 k Ω am NF-Ausgang. In dem Fall klemmen Sie das Eingangskabel des Frequenzzählers direkt an das „heiße“ Ende des Widerstands 56 k Ω (\cong Pin 9/IC4).

Nun stellen Sie mit R₂ eine Frequenz von 1200 Hz ein. Es kommt nicht auf ± 10 Hz an, aber so ungefähr sollte die Frequenz doch stimmen.

Danach lösen Sie die Masseverbindung am FLAG-Eingang, damit wird der Eingangswert H. Nun stellen Sie mit R₁ eine Oszillatorfrequenz von

2400 Hz ein. Auch hier ist eine geringe Toleranz zulässig.

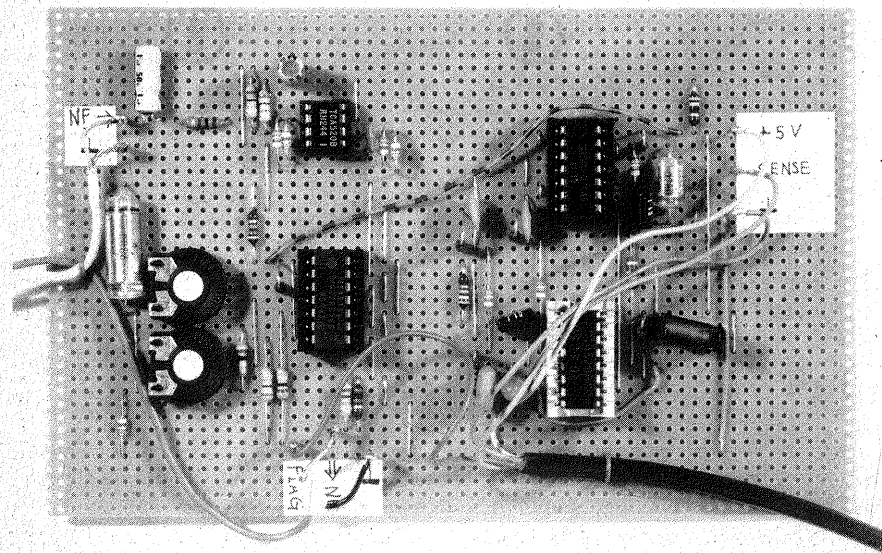
Da sich die Einstellungen von R₁/R₂ gegenseitig beeinflussen können, schalten Sie den FLAG-Eingang wieder auf L und kontrollieren bzw. korrigieren die Frequenzeinstellung von 1200 Hz mit R₂. Anschließend wiederholen Sie die Einstellung von 2400 Hz mit R₁ bei H am FLAG-Eingang.

Nun klemmen Sie den Frequenzzähler ab und sehen sich das Rechtecksignal auf dem Oszillographen genauer an. Es muß ein etwa ausgeglichenes Tastverhältnis haben, d.h. H- und L-Zeiten müssen ungefähr gleich lang sein. Falls sich die Zeiten **erheblich** voneinander unterscheiden, können Sie sie mit R₀=1 M Ω einander angleichen. Das war aber in keinem Fall der nachgebauten Oszillatoren erforderlich. Falls Sie aber R₀ einlöten, wird eine geringfügige Frequenzkorrektur erforderlich sein.

Sie können den Oszillator auch ohne die genannten Meßgeräte ausreichend genau einstellen, wenn Ihnen ein gestimmtes Musikinstrument

(Klavier, Akkordeon, elektronische Orgel o. ä.) zur Verfügung steht. Der Frequenz von 1200 Hz entspricht ein Ton zwischen d''' und dis''' . Sie schließen jetzt schon das Verbindungskabel zum Kassettenrecorder an und schalten den FLAG-Eingang an und schalten den FLAG-Eingang auf L. Wenn Sie nun beide Geräte einschalten, müssen Sie einen Ton hören können – vorausgesetzt, Ihr Kassettenrecorder verfügt über eine Mithörmöglichkeit, auch da gibt es ja allerlei Überraschungen. Wenn nicht, so benutzen Sie irgendeinen NF-Verstärker. Sie können sogar einen hochohmigen Kopfhörer oder Lautsprecher, z. B. eine Telefonhörkapsel, direkt an den Ausgang anschließen. Nun vergleichen Sie den Ton Ihres Oszillators mit dem d''' Ihres Musikinstruments und verstellen R_2 , bis der

Bild 13.6 Das fertige Kassetteninterface auf einer Experimentierplatte



Ton Ihres Oszillators etwas höher klingt als d''' , aber noch nicht so hoch wie dis''' . Sie können die Töne am besten miteinander vergleichen, wenn sie etwa gleich laut klingen.

Anschließend öffnen Sie den FLAG-Eingang, so daß dieser H wird. Nun verstellen Sie R_1 , bis der Ton genau eine Oktave höher klingt als bei L am FLAG-Eingang. Sie können leicht von L nach H und umgekehrt umschalten, indem Sie den FLAG-Eingang mit einem mit 0 V verbundenen Prüfkabel antippen.

Wenn Sie das Kassetteninterface auf einer Experimentierplatte aufbauen, so können Sie Bild 13.6 unmittelbar als Bestückungsplan benutzen. Die einzige Drahtbrücke auf der Lötseite ist eine Verbindung von Pin 9/IC4 zum Ausgangsspannungsteiler $56\text{ k}\Omega / 1\text{ k}\Omega$. Außerdem müssen Sie freilich auf die notwendigen Leiterbahnunterbrechungen achten.

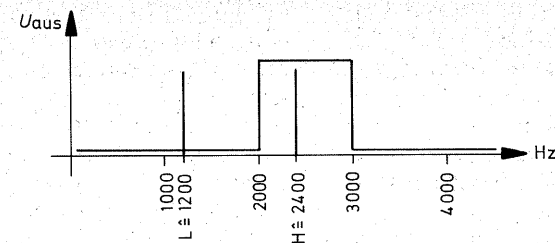


Bild 13.7 Das „High“-Fenster des Kassetteninterface

Zum Prüfen des Empfängers wäre ein durchstimmbarer Tongenerator (ca. 500 Hz bis 4 kHz) sehr nützlich. Sie schließen an den Ausgang (SENSE) ein Vielfachinstrument (oder einen Oszillographen, Einstellung für Gleichspannungsmessung) an, um die Ausgangsspannung zu messen. Sie speisen die Wechselfrequenz des Tongenerators (ca. 300 mV Sinus) in den NF-Eingang ein. Beim Durchstimmen des Tongenerators muß der Ausgang ein „High-Fenster“ zeigen (Bild 13.7). In den Frequenzbereichen 0 Hz bis etwa 2 kHz muß der Ausgang L sein, zwischen etwa 2 kHz und 3 kHz muß er auf H schalten, darüber wird er wieder auf L fallen. Letzteres ist aber uninteressant, weil dieser Frequenzbereich ja überhaupt nicht benutzt wird. Die absoluten Grenzen des High-Fensters sind nicht so wichtig, es kommt lediglich darauf an, daß zur High-Frequenz 2400 Hz ein gewisser Toleranzbereich nach oben wie nach unten vorhanden ist.

Wenn Ihnen kein Tongenerator zugänglich ist, so speisen Sie den Ausgang des Oszillators in den NF-Eingang des Empfängers ein. Dazu müssen die Frequenzen des Oszillators freilich schon eingestellt sein. Wenn der FLAG-Eingang H ist, muß auch der Empfängerausgang auf H schalten, bei L am FLAG-Eingang muß

der Ausgang des Empfängers ebenfalls auf L schalten. Mit dieser Prüfung erhalten Sie zwar keinen Überblick über die Toleranzbreite des High-Fensters, wenn sich der Empfängerausgang richtig verhält, können Sie aber das Kassetteninterface als funktionstüchtig ansehen.

Falls sich das High-Fenster so weit verschieben sollte, daß es die Frequenz 2400 Hz nicht mehr sicher erfaßt, haben Ihnen wahrscheinlich die Bauteiltoleranzen, besonders die der Kondensatoren an den Mono-Flops (C_2 , C_3) einen Streich gespielt. Auch die Umschaltsschwellen der ICs gehen in die Frequenzbestimmung mit ein. Sie unterliegen deutlichen Toleranzen.

Für die untere Grenze des High-Fensters (sie ist die eigentlich wichtige) ist das erste Mono-Flop ($\frac{1}{2}$ IC 3 mit $C_2/39\text{ k}\Omega$) zuständig. Wenn Sie keine HEF-Typen verwenden, sondern Vergleichstypen anderer Hersteller (mit anderer Technologie als LOCMOS), dann wird das High-Fenster wahrscheinlich zu hoch liegen, z. B. zwischen 2,5 und 4 kHz oder gar 3 und 5 kHz. In dem Fall vergrößern Sie entweder C_2 von 10 nF auf 12 oder gar 15 nF, oder Sie vergrößern den zeitbestimmenden Widerstand von 39 k Ω auf 47 oder 56 oder gar 68 k Ω . Am besten finden Sie den richtigen Wert dadurch, daß Sie

– den Widerstand 39 k Ω durch einen Trimmer 100 k Ω ersetzen,

- ein Signal mit einer Frequenz zwischen 1,9 und 2,0 kHz einspeisen,
- den Trimmer so verstellen, daß der Ausgang des Interfaces (SENSE) gerade von H nach L oder umgekehrt umschaltet,
- den Widerstand des benutzten Trimmerteils messen und
- statt seiner einen Festwiderstand mit dem nächsten Normwert einlöten.

Das zweite Mono-Flop ($\frac{1}{2}$ IC 3 mit $C_3/10 \text{ k}\Omega$) und der Tiefpaß ($C_1/18 \text{ k}\Omega$) dienen zur Unterdrückung von Störimpulsen und brauchen daher nicht so genau eingestellt zu werden.

Während aller Einstellarbeiten dürfen FLAG und SENSE des Mikroprozessors nicht angeschlossen sein. Zur Bedienung der Tastatur für Aufnahme und Wiedergabe sei auf die Beschreibung der COMMAND-Funktionen (CMD, siehe Seite 235) verwiesen.

Sie drücken - wenn der Prozessor mit „HALLO.“ anzeigt, daß er sich in der Monitorroutine befindet - zuerst CMD - Anfangs- und Endadresse, dann starten Sie die Aufnahmefunktion Ihres Kassettenrecorders, und wenn Sie den Dauerton einige Sekunden lang aufgenommen haben, drücken Sie eine beliebige Datentaste und beginnen damit die Datenübertragung. Sie können die Datenübertragung auch auf dem Portbaustein beobachten. Zugleich werden Sie auch die FLAG-LEDs flackern sehen, ein sichtbares Zeichen dafür, daß die Daten den Prozessor über den FLAG-Ausgang verlassen.

Versäumen Sie nicht, sich im Anschluß an die Datenübertragung mit Hilfe der EXAMINE-Routine zu vergewissern, daß das Programm Byte

für Byte richtig auf der Kassette gespeichert ist!

Geben Sie CMD-E-Anfangsadresse ein; dann starten Sie die Wiedergabefunktion Ihres Kassettenrecorders, und wenn Sie einen hohen Ton hören, drücken Sie eine beliebige Datentaste, ehe das „Klingeln“ des Programms beginnt. Auf der Anzeige erscheinen E (Examine) und der jeweilige Block, der verglichen wird.

Falls Sie bei der Wiedergabe eines Programms von der Kassette Schwierigkeiten haben sollten, kann es am Ausgangspegel Ihres Kassettenrecorders liegen. Der Empfänger benötigt eine Eingangsspannung von etwa $0,3-0,5 V_{\text{eff}}$. Die Ausgangsspannungen der Kassettenrecorder unterscheiden sich oft erheblich voneinander. Sie können auch den Kopfhörer- oder Lautsprecherausgang zur Steuerung des Empfängers benutzen. Dann stellen Sie mit dem Lautstärkeregler die optimale Ausgangsspannung ein und kennzeichnen sich diese Einstellung.

Weiter ist wichtig, daß die Wiedergabe mit dem aufgenommenen hohen Dauerton beginnt. Sie drücken also zuerst CMD, dann starten Sie die Wiedergabefunktion des Kassettenrecorders, und wenn Sie den Dauerton hören, dann drücken Sie schnell auf F, ehe das „Klingeln“ des Programms beginnt.

Beim Tausch von Programmkassetten kann es vorkommen, daß der Computer das Programm von der geliehenen Kassette nicht annimmt. Dafür gibt es zwei mögliche Ursachen, von denen eine allein u.U. schon den Mißerfolg bewirken kann:

1. Die CLOCK-Frequenzen der Computer weichen deutlich voneinander ab. Es ist daher für den Cassettentausch wichtig, die CLOCK-Fre-

quenz möglichst genau auf 1 MHz abzugleichen.

2. Die Geschwindigkeiten der Kassettenlaufwerke können sehr stark voneinander abweichen. Nahezu alle Kassettengeräte haben einen Trimmer zur Einstellung der Laufgeschwindigkeit. Damit können Sie Ihr Gerät genau dem Gerät Ihres Tauschpartners angleichen (lassen). Bitten Sie ihn, einen Stimmgabelton von 440 Hz aufzunehmen, und wenn Sie diese Kassette auf Ihrem Gerät abspielen, vergleichen Sie dessen Ton mit dem Ton einer gleichen Stimmgabel. Leichter geht es freilich, wenn Ihnen ein Tongenerator und ein Frequenzzähler zur Verfügung stehen.

Falls Sie eine Programmkassette kopieren wollen, sollten Sie sich die Zeit nehmen, das Programm in Ihren eigenen Computer zu laden und von dort mit Ihrem eigenen Kassettenrecorder aufzunehmen. Dann werden Sie mit Ihrer Programmkassette keine Schwierigkeiten bekommen. Überspielen Sie dagegen Programme wie Musikstücke von Recorder zu Recorder, so addieren sich die Zeitfehler, und dann werden Übertragungsfehler wahrscheinlich.

Kapitel 14

Programmbeispiele

Allgemeine Hinweise zur Programmierung und Anwendung der Programme

Die folgenden Programme sollen Sie in die Lage versetzen, sogleich mit Ihrem Computer etwas zu tun. Alle Befehle sind erläutert, so daß Sie beim Verfolgen der Programme auch verstehen können, was sich da ereignet. Das wird Sie ermutigen, auch eigene Ideen in Programme umzusetzen. Verändern Sie zuerst die vorhandenen Programme, und beobachten Sie, welche Folgen sich daraus ergeben, dann beginnen Sie mit eigenen kurzen Sequenzen.

Es hat sich bewährt, eigene Programme nach dem Muster von Bild 14.1 auf Formblätter zu schreiben. Sie brauchen nicht unbedingt gedruckte Formulare. Es genügt, einen Bogen DIN A4 quer in Spalten für

- die Zeilennummer,
- Adresse,
- Plätze für 3 Bytes (ein Befehl kann ja bis zu 3 Bytes lang sein);
- Label (engl. Etikett; gemeint ist der Name eines Programms oder Programmteils);

- den Befehl in mnemonischer Abkürzung;
- den oder die beiden Operanden (Daten, Adressen) und
- den Kommentar aufzuteilen.

Das Blatt in Bild 14.1 enthält am unteren Rand eine Umrechnungstabelle für die relative Adressierung. Die Reihe „N“ gibt die Sprungweite (Anzahl der zu überspringenden Bytes) dezimal an, die Reihe „+“ den entsprechenden Versatz hexadezimal für einen Vorwärtssprung; die Reihe „-“ entsprechend für einen Rückwärtssprung; die beiden letztgenannten Reihen geben also das 2. Byte für einen Befehl mit direkter relativer Adressierung an; bei indirekter relativer Adressierung muß zum 2. Byte jeweils der Wert 80_{16} addiert werden, weil dann das Bit 7 ja immer den Wert 1 haben muß.

Man beginnt beim Programmieren mit dem mnemonischen Code, z. B. LODI, R3 und codiert die Maschinsprache erst, wenn die Abfolge der Befehle fertig ist. Die besonders zeitraubende Arbeit der relativen Adressierung kann Ihnen der Computer abnehmen. Laden Sie dazu gleichzeitig die Programme HEX-DEZ-HEX-Wandlung (siehe Seite 274) und HEX-Rechnen (siehe Seite

PROGRAMM	DIVISION			VERSION	A	STARTADR.	TEILPROGRAMM		NAME/DATUM	08.03.84/EL
	BESCHREIBUNG	UND ZEIT	DAS IN R ₀ DURCH DEN WERT IN R ₂				BLATT 4	VON		
ZEILE	ADRESSE	B0	B1	B2	LABEL	BEFEHL	OPERAND(EN)	KOMMENTAR	ZYKL.	
1	0000	75	08		START	CPSL	WC	NEBERTRAG ABSCHALTEN		
2	0002	05	00			LODI, R1	00	R1 AUF 0		
3	0004	04	FE			LODI, R0	FE	R0 MIT DIVIDEND LADEN		
4	0006	06	08			LODI, R2	08	R2 MIT DIVISOR LADEN		
5	0008	CE	00	50		STRA, R2	0050	DIVISOR SICHERN		
6	000B	A2			LOOP	SUBZ, R2		DIVISOR VON DIVIDEND ABZIEHEN		
7	000C	EC	00	50		COMA, R0	0050	REST MIT DIVISOR VERGLEICHEN		
8	000F	1A	04			BCTR, LT	SHOW	REST KLEINER, DANN ERGEBN. ZEIGEN		
9	0011	85	01			ADDI, R1	01	A ZUM ERGEBNIS ADDIEREN		
10	0013	1B	76			BCTR, UN	LOOP	WEITER ABZIEHEN		
11	0015	D5	09		SHOW	WRITE, R1	09	ERGEBNIS ANZEIGEN		
12	0017	40				HALT		PROZESSOR STÖPPEN		
13					**	HOECHSTER	VERT DES DIVIDENDEN	UND DES DIVISORS IST FF		
14										
15										
16										
17										

+	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	
N	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
-	7F	7E	7D	7C	7B	7A	79	78	77	76	75	74	73	72	71	70	6F	6E	6D	6C	6B	6A	69	68	67	66	65	64	63	62	61		
+	20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F	30	31	32	33	34	35	36	37	38	39	3A	3B	3C	3D	3E	3F	
N	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
-	60	5F	5E	5D	5C	5B	5A	59	58	57	56	55	54	53	52	51	50	4F	4E	4D	4C	4B	4A	49	48	47	46	45	44	43	42	41	40

DIRECT RELATIVE ADDRESSING - SECOND BYTE INDIRECT RELATIVE ADDRESSING: ADD 80₁₆ TO DISPLACEMENT

Bild 14.1

276). Setzen Sie den Programmcounter auf 0A00.

Die Rechenregel heißt: Zieladresse minus 1. Byte des der Startadresse folgenden Befehls.

1. Beispiel: Sie wollen von Adresse 000F zur Adresse 0015 verzweigen.

Eingabe: RUN, 0015, - (GOTO-Taste) 000F

Ausgabe: 0006 = Sprungweite 06₁₆.

2. Beispiel: Sie wollen von Adresse 0013 zur Adresse 000B verzweigen.

Eingabe: RUN, 000B, - (GOTO-Taste), 0013

Ausgabe: FFF6 = Sprungweite 76: Bei relativer Adressierung interessiert nur das „untere“ Byte; daher wird das „obere“ Byte nicht betrachtet.

Das Bit 7 des unteren Bytes gehört nicht zur Sprungweite, sondern es gibt an, ob **direkt** relativ (Bit 7 = 0) adressiert werden soll oder **indirekt** relativ (Bit 7 = 1). Es muß zunächst aus dem errechneten Wert gestrichen werden, und so wird aus dem Rechenergebnis F6 die Sprungweite 76. Anschließend wird je nach gewünschter Adressierungsart das Bit 7 mit 0 oder 1 belegt.

3. Beispiel: Sie wollen von Adresse 0013 8 Bytes (dezimal) zurück verzweigen.

Eingabe: GOTO, D, + (RUN-Taste), 00008

Ausgabe: FFF6 (= 76, siehe 2. Beispiel)

Eingabe: RESET, RUN, 0013, + (GOTO-Taste), FFF6

Ausgabe: 000B = Zieladresse.

Die Beispiele sind dem Programm DIVISION entnommen. Vor dem Schreiben eigener Programme sollten Sie die hexadezimale Adreßberechnung anhand der Musterprogramme üben. An diesen können Sie ja kontrollieren, ob Ihre Berechnungen stimmen.

Auch die Benutzung der **im Monitor enthaltenen Benutzerroutinen** sollten Sie sich über kurz oder lang **anhand der Musterprogramme einprägen**. Dort ist auch genau gezeigt, welche Eingaben diese Routinen erwarten und welche Ausgaben sie erzeugen. Auf Seite 234 wurde bereits aufgelistet, welche Monitorfunktionen der Benutzer direkt aufrufen kann. Darüber hinaus enthält der Monitor viele

Liste der nutzbaren Monitorroutinen

NAME	BEFEHL	ZWECK	EINGABE	AUSGABE	BEN. REGISTER
HEXDEZ	BB 83	UMSETZ. EINER 4-STELLIGEN BINÄRZAHLEN IN EINE 5-STELLIGE DEZIMALZAHLEN.	082B 082C	082E 082F 0830	R0, R1, R2, R3 PSL
CONV	BB 85	UMWANDL. EINES HEX-WERTES IN 7-SEGMENT-DARSTELLUNG FÜR DIE ANZEIGE AUF DEM DISPLAY.	R3	R3	R3
SEP	BB 87	WANDELT EIN DATENBYTE IN 2 BYTES UM.	0801	0801 0802	R3
COMB	BB 89	WANDELT DIE 4 RECHTEN BIT VON 2 BYTES IN EIN DATENBYTE UM.	0801 0802	0801 0802	R3
DIS	BB 8B	ZEIGT DEN INHALT DER SPEICHERSTELLEN 080D BIS 0812 AUF DEM DISPLAY. ANFRAGE DIE TASTATUR AB, BIS TASTE GEDRÜCKT WIRD.	-	-	R0
KIN	BB 8D	ZWISCHENDURCH ANZEIGE ÜBER DIE DIS ROUTINE.	-	0800	R0, R3
INIT	BB 8F	SETZT ALLE REGISTER AUF 0	-	-	-
SAVE	BB 91	SICHERT ALLE REGISTER	-	-	-
RECAL	BB 93	REGENERIERT DEN LETZTEN INHALT ALLER REGISTER	-	-	-
CDIS	BB 95	LOESCHT DEN ANZEIGESPEICHER 080D-0812	-	-	R0, R1
LODAT	BB 99	ERLAUBT EINGABE EINER 2-, 4- ODER 5STELLIGEN ZAHLEN ÜBER TASTATUR UND ZUSAMMENFASSUNG IN 3 BYTES.	082D	081A 081B 081C	R0, R1, R2, R3
ADDSUB	BB 9B	ADDITION/SUBTRAKTION ZWEIER NEG./POS. 4STELLIGER BINÄRZAHLEN (2BYTES). DAS COM-BIT IM PSL (BIT1) GIBT AN, OB SUBTRAKT(0) ODER ADDITION(1) GEWÜNSCHT IST	0827 0828 0829 082A PSL	082B 082C	R0, R1, PSL
MULT	BB 9D	MULTIPLIK. ZWEIER 2-STELL. POS./NEG. BINÄRZAHLEN JE 1 BYTE LANG	0827 0829	082B 082C	R0, R1, R2, R3 PSL
DIV	BB 9F	DIVISION ZWEIER BINÄRZAHLEN 4-STELL. DIVIDEND (2BYTES) 2-STELL. DIVISOR (1BYTE) POS./NEG. DIVIDEND GEHT VERLOREN, NEG. DIVISOR WIRD POSITIV	0827 0828 0829	082B 082C R1=REST	R0, R1, R2, R3 PSL
DEZHEX	BB A1	UMSETZ. EINER 5-STELLIGEN DEZ. ZAHLEN MIT VORZEICHEN ZW. +32767 U. -32768 IN EINE 4-STELL. BINÄRZAHLEN. NEG. BINÄRZAHLEN WERDEN ALS ZWEIERKOMPLEMENT DARGESTELLT	082E 082F 0830	082B 082C	R0, R1, R2, R3 PSL

Routinen, die für interne Zwecke benötigt werden, jedoch dem Anwender zur Verfügung stehen. Eigene Anwenderprogramme lassen sich damit erheblich einfacher schreiben. Die Benutzung dieser Routinen ist in den folgenden Programmen demonstriert (ZBSR-Befehle). Die folgende Liste gibt einen Überblick über alle nutzbaren Routinen und deren Adressen.

Die Registerinhalte der aufgeführten „benutzten Register“ gehen bei der Abarbeitung der Monitorroutinen verloren. Sie müssen deshalb gegebenenfalls vor dem Anspringen der Routinen in Hilfsspeichern gesichert und anschließend wieder geladen werden.

HEXD	BB A5	ANZEIGE EINER 4-STELLIGEN BINAERZAHL ALS 5-STELLIGE DEZIMALZAHL MIT VORZEICH.	082B 082C	R0,R1,R2,R3 PSL
DHEX	BB A7	ANZEIGE EINER 5-STELLIGEN DEZIMALZAHL MIT VORZEICHEN ALS 4-STELLIGE BINAERZAHL	082E 082F 0830	R0,R1,R2,R3 PSL
HEX	BB A9	ANZEIGE EINER 4-STELLIGEN BINAERZAHL	082B 082C	R0,R1,R2,R3
DEZ	BB AB	ANZEIGE EINER 5-STELLIGEN DEZIMALZAHL MIT VORZEICHEN	082E 082F 0830	R0,R1,R2,R3

AN DIE LETZTEN 4 ROUTINEN IST EINE ANZEIGE-ROUTINE WIE 'DIS' ODER 'KIN' ANZUSCHLIESSEN.

Vom Monitor benutzte Speicherstellen

DATA1	0800	EIN- UND AUSGABE- SPEICHER FUER UMWANDLUNGS- ROUTINEN
DATA2	0801	
DATA3	0802	
KINBUF	0803	INTERNER PUFFER DER KIN ROUTINE
RSAVE	0804-080C	REGISTER-SICHERUNG Z.B. FUER DIE BREAKPOINT-ROUTINE
DISBUF	080D-0812	ANZEIGESPEICHER, DIE IN DIESEM SPEICHERBEREICH ENTHALTENEN DATEN WERDEN DURCH DIE DIS-ROUTINE ANGEZEIGT
	0813-0814	INTERNE SPEICHER
WBUF	0815-081C	INTERNE ARBEITSSPEICHER
PCH	081D	ENTHAELT DIE ADRESSE DES PROGRAMM-COUNTERS
PCL	081E	
ADRS	081F-0821	ENTHAELT ANFANGS-ADRESSE UND BLOCKLAENGE DER CASSETTEN- ROUTINE
SUM	0822	SUMCHECK BYTE DER CASSETTEN-ROUTINE
BPADR	0823-0824	ENTHAELT DIE BREAKPOINT-ADRESSE
BPINH	0825-0826	ENTHAELT DEN AN DER BREAKPOINT-ADRESSE GESPEICHERTEN BEFEHL
OPR1	0827-0828	EINGABESPEICHER FUER ARITHMETISCHE OPERATIONEN
OPR2	0829-082A	EINGABESPEICHER FUER ARITHMETISCHE OPERATIONEN
RSLT	082B-082C	AUSGABESPEICHER FUER ARITHMETISCHE OPERATIONEN
STAT	082D	STATUS-BYTE FUER ARITHMETISCHE OPERATIONEN UND FUER DIE DATENEINGABE
BCDD	082E-0830	SPEICHER FUER DEZIMALZAHLEN
	0831-08FF	FREI FUER ANWENDERZWECKE

Der Bereich 0800-08FF kann nicht auf Cassette gespeichert werden. Das Abspeichern von Programmen darf deshalb erst bei 0900 beginnen.

Zu Beginn aller Programme ist jeweils die benutzte Hardware angegeben. „Grundversion“ bedeutet: CPU, Daten- und Adreßein- und -ausgabe, RAM-Speicher und Portbaustein, ohne EPROM und ohne Tastatur- und Anzeigeeinheit. In diesen Fällen ist der Schalter auf der Speicherkarte in Stellung „1“ zu bringen, so daß der RAM-Bereich bei der Adresse 0000 beginnt.

Wenn Sie die Tastatur- und Anzeigeeinheit benutzen, muß der Schalter auf der Speicherkarte in Stellung „2“ stehen, so daß das EPROM am Anfang des Adressierbereichs liegt. Dann können Sie die Daten- und Adreßein- und -ausgabeeinheiten von

der Busplatte abtrennen. Sie können sie aber auch angeschlossen lassen. In dem Fall müssen der Datenshalter Da in Stellung DAFLOT und der Adreßschalter Adr in Stellung AD-FLOT stehen, sonst läuft nichts! Diese beiden Baustufen nützen Ihnen dann, wenn Sie ein Programm im Single-STEP oder Einzel-CLOCK durchtakteten; dann muß der Schalter SCHRITT-LAUF auf der Tastatur in Stellung SCHRITT stehen, damit die Siebensegmentanzeigen im Falle ihrer Aktivierung nicht überlastet werden. Den Programmfortschritt können Sie dann auf der Daten- und Adreßein- und -ausgabe beobachten.

440-Hz-Programm zum Abgleich des CLOCK-Oszillators

LINE	ADDR	B1	B2	B3	B4	LABEL	OPCODE	OPERAND	COMMENTS	PAGE	0002
0001	0000						** PROGRAMM: OSZILLATORABGLEICH				
0002	0000						** HARDWARE: GRUNDVERSION -1MHZ				
0003	0000						** 150 OHM LAUTSPRECHER AN DEN PORT 09 ANSCHLIESSEN				
0004	0000						**				
0005	0000						** FUNKTION: DAS PROGRAMM ERZEUGT EINEN TON, DER BEI GENAUER				
0006	0000						** EINSTELLUNG DES OSZILLATORS AUF DER CPU-PLATINE EXAKT				
0007	0000						** 440 HZ (ALSO KAMMERTON A) BETRAEGT. MIT HILFE EINER				
0008	0000						** STIMMGABEL KANN ALSO DER OSZILLATOR DES RECHNERS GENAU AUF				
0009	0000						** 1MHZ ABGEGlichen WERDEN (AUF SCHWEBUNGSNULl)				
0010	0000						ORG 0000				
0011	0000	05	00			START	LODI,R1	00	R1 AUF 0		
0012	0002	06	FF				LODI,R2	FF	R2 AUF FF		
0013	0004	D6	09			TON	WRTE,R2	09	PORT EINSCHALTEN		
0014	0006	3F	00	10			BSTA,UN	DELAY	TON ERZEUGEN		
0015	0009	D5	09				WRTE,R1	09	PORT AUSSCHALTEN		
0016	000B	3F	00	10			BSTA,UN	DELAY	VERZOEGERN		
0017	000E	1B	74				BCTR,UN	TON	ZURUECK ZUM ANFANG		
0018	0010	04	2C			DELAY	LODI,R0	2C	R0 AUF 2C (TONWERT FUER 440 HZ)		
0019	0012	07	01			DELAY1	LODI,R3	01	R3 AUF 1		
0020	0014	FB	7E				BDRR,R3	\$	R3 AUF 0 ZAEHLEN		
0021	0016	FB	7A				BDRR,R0	DELAY1	R0 AUF 0 ZAEHLEN		
0022	0018	17					RETC,UN	ZURUECK	INS HAUPTPROGRAMM		

Portadressierung

```

LINE  ADDR B1 B2 B3 B4 LABEL  OPCODE  OPERAND  COMMENTS  PAGE 0001
0001  0000          ** PROGRAMM: PORT-ADRESSIERUNG
0002  0000          ** HARDWARE: GRUNDVERSION - SINGLESTEP ODER 1 HZ
0003  0000          **
0004  0000          ** FUNKTION: DIE ZAHL H'01' WIRD IN DAS REGISTER R1 GELADEN UND ZUM
0005  0000          ** PORT 09 TRANSPORTIERT. DANN WIRD DER PROZESSOR ANGEHALTEN.
0006  0000          ORG      0000
0007  0000 05 01  START  LODI,R1  01
0008  0002 05 09          WRTE,R1  PORT  INH.VON R1 ZUM PORT SENDEN
0009  0004 40          HALT      PROGRAMM BEENDEN
0010  0005          ** VERSUCHEN SIE ES AUCH MIT ANDEREN WERTEN IN R1

```

Einfache Addition

```

LINE  ADDR B1 B2 B3 B4 LABEL  OPCODE  OPERAND  COMMENTS  PAGE 0001
0001  0000          ** PROGRAMM: ADDITION
0002  0000          ** HARDWARE: GRUNDVERSION - SINGLESTEP ODER 1 HZ
0003  0000          **
0004  0000          ** FUNKTION: ZUNAECHST WIRD DER UEBERTRAG ABGESCHALTET, DAMIT BEI DER
0005  0000          ** ERSTEN ADDITION NICHT EIN ZUFUELLIG VORHANDENER UEBERTRAG MIT
0006  0000          ** ADDIERT WIRD. DANN WIRD REGISTER R1 AUF 0 GESETZT UND JEWEILS UM
0007  0000          ** 1 ERHOEHET. DER JEWEILIGE REGISTERINHALT WIRD AUF DEM PORT 09 ANGEZEIGT
0008  0000          ORG      0000
0009  0000 75 08  START  CPSL    WC      UEBERTRAG ABSCHALTEN
0010  0002 05 00          LODI,R1  00      WERT 0 NACH R1
0011  0004 85 01  LOOP  ADDI,R1  01      WERT IN R1 UM 1 ERHOEHEN
0012  0006 05 09          WRTE,R1  PORT  INH.VON R1 ZUM PORT SENDEN
0013  0008 1F 00 04          BCTA,UN  LOOP  SCHLEIFE AB LOOP WIEDERHOLEN
0014  000B          ** ACHTUNG! DAS PROGRAMM LAEUFT EWIG WEITER.
0015  000B          ** WOLLEN SIE ES BEENDEN, SO BETAETIGEN SIE
0016  000B          ** RESET UND HOLD

```

Einfache Subtraktion

```

LINE  ADDR B1 B2 B3 B4 LABEL  OPCODE  OPERAND  COMMENTS  PAGE 0001
0001  0000          ** PROGRAMM: SUBTRAKTION
0002  0000          ** HARDWARE: GRUNDVERSION - SINGLESTEP ODER 1 HZ
0003  0000          **
0004  0000          ** FUNKTION: PROGRAMM SUBTRAHIERT EINEN WERT VOM INHALT VON
0005  0000          ** R1 UND ZEIGT DAS ERGEBNIS AUF DEM AUSGANGSPORT
0006  0000          ORG      0000
0007  0000 75 08  START  CPSL    WC      UEBERTRAG ABSCHALTEN
0008  0002 05 FF          LODI,R1  FF      1.WERT IN R1 LADEN

```

```

0009  0004 A5 55          SUBI,R1  55      2.WERT VON R1 SUBTRAHIEREN
0010  0006 D5 09          WRTE,R1  PORT  INH.VON R1 ZUM PORT SENDEN
0011  0008 40          HALT      PROGRAMM BEENDEN
0012  0009          ** VERSUCHEN SIE ES AUCH MIT ANDEREN WERTEN IN R1
0013  0009          ** UND IM SUBTRAKTIONSBEFEHL.

```

Einfache Multiplikation

```

FILE 'PROG19' AS ASSEMBLED BY SYSTEM ON 07-03-84          PAGE 0002
LINE  ADDR B1 B2 B3 B4 LABEL  OPCODE  OPERAND  COMMENTS
0001  0000          ORG      0000
0002  0000          ** PROGRAMM: MULTIPLIKATION
0003  0000          ** HARDWARE: GRUNDVERSION - SINGLE STEP, 1HZ ODER 1 MHZ
0004  0000          **
0005  0000          ** FUNKTION: DAS PROGRAMM MULTIPLIZIERT DIE BEIDEN WERTE IN
0006  0000          ** R1 UND R2 INDEM ES DEN INHALT VON R1 R2* IN DAS REGISTER
0007  0000          ** R0 ADDIERT.
0008  0000 75 08  START  CPSL    WC      UEBERTRAG ABSCHALTEN
0009  0002 20          EORZ    RO      RO AUF 0
0010  0003 05 07          LODI,R1  07      R1 MIT MULTIPLIKATOR LADEN
0011  0005 06 08          LODI,R2  08      R2 MIT MULTIPLIKAND LADEN
0012  0007 81          LOOP  ADDZ,R1          ADDIERE MULTIPLIKATOR
0013  0008 FA 7D          BDRR,R2  LOOP  MULTIPLIK.-1, WENN UNGL.0 WEITER
0014  000A D4 09          WRTE,RO  09      ERGEBNIS AUF DEN PORT SCHREIBEN
0015  000C 40          HALT
0016  000D          ** MULTIPLIKATOR UND MULTIPLIKAND MUESSEN SO GEWAHLT WERDEN,
0017  000D          ** DASS DAS ERGEBNIS NICHT GROESSER ALS FF IST.

```

Einfache Division

```

FILE 'PROG20' AS ASSEMBLED BY SYSTEM ON 09-03-84          PAGE 0002
LINE  ADDR B1 B2 B3 B4 LABEL  OPCODE  OPERAND  COMMENTS
0001  0000          ORG      0000
0002  0000          ** PROGRAMM: DIVISION
0003  0000          ** HARDWARE: GRUNDVERSION - SINGLE STEP, 1HZ ODER 1MHZ
0004  0000          **
0005  0000          ** FUNKTION: DAS PROGRAMM DIVIDIERT DEN WERT IN R0 DURCH
0006  0000          ** DEN WERT IN R2 INDEM ES DEN WERT IN R2 SO OFT VON R0
0007  0000          ** ABZIEHT, BIS R0 KLEINER R2 IST. DIE ZAHL DER SUBTRAK-
0008  0000          ** TIONEN WIRD IN R1 GEZAELT UND ALS ERGEBNIS ANGEZEIGT.
0009  0000 75 08  START  CPSL    WC      UEBERTRAG ABSCHALTEN
0010  0002 77 02          PPSL    COM     PSL AUF LOGICAL COMPARE
0011  0004 05 00          LODI,R1  00      R1 AUF 0
0012  0006 04 FE          LODI,RO  FE      R0 MIT DIVIDEND LADEN
0013  0008 06 08          LODI,R2  08      R2 MIT DIVISOR LADEN

```



```

0014 000A CE 00 50      STRA,R2  0050  DIVISOR SICHERN
0015 000D EC 00 50      COMA,RO  0050  DIVIDEND KLEINER DIVISOR?
0016 0010 1A 0A          BCTR,LT  SHOW   WENN JA, 0 ANZEIGEN
0017 0012 A2             LOOP    SUBZ,R2  DIVISOR VON DIVIDEND ABZIEHEN
0018 0013 85 01          ADDI,R1  01     1 ZUM ERGEBNIS ADDIEREN
0019 0015 EC 00 50      COMA,RO  0050  REST MIT DIVISOR VERGLEICHEN
0020 0018 1A 02          BCTR,LT  SHOW   REST KLEINER, DANN ERGEBN.ANZEIGEN
0021 001A 1B 76          BCTR,UN  LOOP   WEITER ABZIEHEN
0022 001C D5 09          SHOW    WRTE,R1  09     ERGEBNIS ANZEIGEN
0023 001E 40             HALT
0024 001F                ** HOECHSTER WERT DES DIVIDENDEN UND DES DIVISORS IST FF.

```

Verkehrssampel 1

```

LINE ADDR B1 B2 B3 B4 LABEL OPCODE OPERAND COMMENTS PAGE 0001
0001 0000                ** PROGRAMM: VERKEHRSSAMPEL 1
0002 0000                ** HARDWARE: GRUNDVERSION - SINGLESTEP ODER 1 HZ
0003 0000                **
0004 0000                ** FUNKTION: DAS PROGRAMM SIMULIERT EINE VERKEHRSSAMPEL. DIE
0005 0000                ** OBEREN 3 PORT-LED STELLEN DIE HAUPTSTRASSEN-
0006 0000                ** AMPEL DAR, DIE UNTEREN 3 DIE QUERSTRASSENAMPEL
0007 0000                DRG     0000
0008 0000 04 24          START  LODI,RO  24     HAUPTSTR.GRUEN,QUERSTR.ROT
0009 0002 D4 09          WRTE,RO  PORT   INH.VON RO ZUM PORT SENDEN
0010 0004 04 42          LODI,RO  42     GELBPHASE 1
0011 0006 D4 09          WRTE,RO  PORT   INH.VON RO ZUM PORT SENDEN
0012 0008 04 81          LODI,RO  81     HAUPTSTR.ROT,QUERSTR.GRUEN
0013 000A D4 09          WRTE,RO  PORT   INH.VON RO ZUM PORT SENDEN
0014 000C 04 42          LODI,RO  42     GELBPHASE 2
0015 000E D4 09          WRTE,RO  PORT   INH.VON RO ZUM PORT SENDEN
0016 0010 1B 6E          BCTR,UN  START  ZURUECK ZUM ANFANG
0017 0012                ** DAS PROGRAMM WIRKT REALISTISCHER, WENN AN DEN PORT-
0018 0012                ** STECKER ROTE,GELBE UND GRUENE LED ANGESCHLOSSEN WER-
0019 0012                ** DEN. WER ES KOMFORTABLER WUENSCHT, SOLLTE PROGRAMM
0020 0012                ** VERKEHRSSAMPEL 2 VERSUCHEN.

```

Blinklicht

```

LINE ADDR B1 B2 B3 B4 LABEL OPCODE OPERAND COMMENTS PAGE 0001
0001 0000                ** PROGRAMM: BLINKLICHT
0002 0000                ** HARDWARE: GRUNDVERSION - 1MHZ
0003 0000                **
0004 0000                ** FUNKTION: IN DIESEM PROGRAMM BLINKEN ALLE LED IN DER PORT-
0005 0000                ** ANZEIGE. DA DAS PROGRAMM MIT 1 MHZ LAEUFT, WUERDEN DIE LED
0006 0000                ** SO SCHNELL BLINKEN, DASS SIE FUER DAS TRAEGE AUGE DAUERND
0007 0000                ** LEUCHTEN WUERDEN. DESHALB IST IN DAS PROGRAMM EINE VERZOE-

```

```

0008 0000                ** GERUNGSSCHLEIFE EINGEBAUT, DIE DIE EIN- UND AUSZEITEN VER-
0009 0000                ** LAENGERT.
0010 0000                DRG     0000
0011 0000 05 FF          START  LODI,R1  FF     FF NACH R1=ALLE LED EIN
0012 0002 D5 09          WRTE,R1  PORT   INH.VON R1 ZUM PORT SENDEN
0013 0004 3F 00 11      BSTA,UN  DELAY  VERZOEГ.SCHLEIFE ANSPRINGEN
0014 0007 05 00          LODI,R1  00     00 NACH R1=ALLE LED AUS
0015 0009 D5 09          WRTE,R1  PORT   INH.VON R1 ZUM PORT SENDEN
0016 000B 3F 00 11      BSTA,UN  DELAY  VERZOEГ.SCHLEIFE ANSPRINGEN
0017 000E 1F 00 00      BCTA,UN  START  ZURUECK ZUM START
0018 0011                ** VERZOEGERUNGSSCHLEIFE:
0019 0011 06 FF          DELAY  LODI,R2  FF     VERZOEГ.ZEIT EINSTELLEN
0020 0013 07 FF          DELAY2 LODI,R3  FF     VERZOEГ.ZEIT EINSTELLEN
0021 0015 FB 7E          DELAY3 BDRR,R3  DELAY3 R3 BIS 0 ZURUECKZAEHLEN
0022 0017 FA 7A          BDRR,R2  DELAY2 R2 BIS 0 ZURUECKZAEHLEN
0023 0019 17            RETC,UN
0024 001A                ** WENN SIE DEN WERT VON R2 UND/ODER R3 VERAENDERN,
0025 001A                ** WIRD DIE BLINKFREQUENZ HOECHER. MERKEN SIE SICH DIE
0026 001A                ** VERZOEGERUNGSRoutine 'DELAY' GUT! SIE WIRD IMMER
0027 001A                ** WIEDER GEBRAUCHT.

```

Wuerfel

```

LINE ADDR B1 B2 B3 B4 LABEL OPCODE OPERAND COMMENTS PAGE 0001
0001 0000                ** PROGRAMM: WUERFEL
0002 0000                ** HARDWARE: GRUNDVERSION - 1MHZ
0003 0000                ** SENSE ANSCHLUSS(PIN 6 DER 7-POL-BUCHSE AUF CPU)
0004 0000                ** MIT 50 HZ AUSGANG DER NETZPLATINE VERBINDEN
0005 0000                ** FUNKTION: DAS PROGRAMM ERZEUGT EINE ZUFALLSZAHL ZWISCHEN 1 UND 6
0006 0000                ** UND ZEIGT SIE AUF DEM PORT AN. MAN KANN DAMIT ALSO
0007 0000                ** EINEN WUERFEL SIMULIEREN
0008 0000                DRG     0000
0009 0000 3B 03          START  BSTR,UN  RANDOM ZUFALLSZAHL ERZEUGEN
0010 0002 D7 09          WRTE,R3  PORT   ZUM PORT SENDEN
0011 0004 40             HALT
0012 0005 12            RANDOM  SPSU     INHALT VON PSU NACH RO LADEN
0013 0006 44 80          ANDI,RO  80     NUR HOECHSTES BIT BETRACHTEN
0014 0008 CC 07 15      STRA,RO  0715   SENSE ZUSTAND SPEICHERN
0015 000B 07 00          NEW     LODI,R3  00     AUGEN-ZAEHLER SETZEN
0016 000D 87 01          LOOP    ADDI,R3  01     AUGEN-ZAEHLER UM 1 ERHOEHEN
0017 000F E7 07          COMI,R3  07     WUERFEL WERT GROESSER 6?
0018 0011 18 78          BCTR,EQ  NEW     VON VORN ZAEHLEN
0019 0013 12            SPSU     INHALT VON PSU NACH RO LADEN
0020 0014 44 80          ANDI,RO  80     NUR HOECHSTES BIT BETRACHTEN
0021 0016 EC 07 15      COMA,RO  0715   SENSE ZUSTAND VERAENDERT?
0022 0019 18 72          BCTR,EQ  LOOP   WENN NEIN, WEITERZAEHLEN
0023 001B 17            RETC,UN  ZURUECK INS HAUPTPROGRAMM
0024 001C                ** DIE RANDOM ROUTINE LAESST SICH IMMER DANN VERWENDEN, WENN
0025 001C                ** ZUFALLSZAHLEN ERZEUGT WERDEN SOLLEN. DURCH AENDERUNG DER
0026 001C                ** SPEICHERSTELLE 0010 LASSEN SICH AUCH ANDERE ZAHLEN ERZEUGEN

```

Metronom

LINE	ADDR	B1	B2	B3	B4	LABEL	OPCODE	OPERAND	COMMENTS	PAGE
0001	0000								** PROGRAMM: METRONOM	0001
0002	0000								** HARDWARE: GRUNDVERSION - 1MHZ	
0003	0000								**	
0004	0000								** FUNKTION: AN DEN AUSGANGSPORT DER PORT-PLATINE (PORT 09)	
0005	0000								** MUSS AN DEN ANSCHLUSS DO EIN 150 OHM LAUTSPRECHER	
0006	0000								** GEGEN MASSE ANGESCHLOSSEN WERDEN	
0007	0000						ORG	0000		
0008	0000	06	00			START	LODI,R2	00	REGISTER 2 MIT 0= PORT AUS LADEN	
0009	0002	07	FF				LODI,R3	FF	REGISTER 3 MIT 1= PORT EIN LADEN	
0010	0004	D7	09			BEAT	WRTE,R3	09	PORT AUSSCHALTEN	
0011	0006	04	FF				LODI,R0	FF	VERZOEGERUNGSZEIT EINSTELLEN	
0012	0008	F8	7E				BDRR,R0	\$	R0 AUF 0 ZAEHLEN	
0013	000A	D6	09				WRTE,R2	09	PORT EINSCHALTEN	
0014	000C	3B	02				BSTR,UN	DELAY	BIS ZUM NAECHSTEN BEAT WARTEN	
0015	000E	1B	74				BCTR,UN	BEAT	NAECHSTEN BEAT ERZEUGEN	
0016	0010	04	02			DELAY	LODI,R0	02	VERZOEGERUNGSZEIT EINSTELLEN	
0017	0012	05	FF			DELAY1	LODI,R1	FF	VERZOEGERUNGSZEIT EINSTELLEN	
0018	0014	06	FF			DELAY2	LODI,R2	FF	VERZOEGERUNGSZEIT EINSTELLEN	
0019	0016	FA	7E			DELAY3	BDRR,R2	\$	R2 BIS 0 HERUNTERZAEHLEN	
0020	0018	F9	7A				BDRR,R1	DELAY2	R1 BIS 0 HERUNTERZAEHLEN	
0021	001A	F8	76				BDRR,R0	DELAY1	R0 BIS 0 HERUNTERZAEHLEN	
0022	001C	17					RETC,UN		ZURUECK ZUM HAUPTPROGRAMM	
0023	001D								** DURCH VERAENDERUNG DER SPEICHERSTELLE 0011,0013 ODER 0015	
0024	001D								** KANN DIE TAKTFREQUENZ VERAENDERT WERDEN	

Lauflicht

LINE	ADDR	B1	B2	B3	B4	LABEL	OPCODE	OPERAND	COMMENTS	PAGE
0001	0000								** PROGRAMM: LAUFLICHT	0001
0002	0000								** HARDWARE: GRUNDVERSION - 1MHZ	
0003	0000								**	
0004	0000								** FUNKTION: DIESES PROGRAMM VERWANDELT DEN COMPUTER	
0005	0000								** IN EIN LAUFLICHT. DIE 0000 0001 IM REGISTER R1 WIRD BEI JEDEM	
0006	0000								** DURCHLAUF EINE STELLE WEITER NACH LINKS GESCHOBEN, ALSO	
0007	0000								** AUF 0000 0010, 0000 0100 USW. UND DANN AUF DEM PORT ANGE-	
0008	0000								** ZEIGT.	
0009	0000						ORG	0000		
0010	0000	75	08			START	CPSL	WC	UEBERTRAG ABSCHALTEN	
0011	0002	05	01				LODI,R1	01	BINAER 00000001 NACH R1	
0012	0004	D5	09			LOOP	WRTE,R1	PORT	INH.VON R1 ZUM PORT SENDEN	
0013	0006	3F	00	0D			BSTA,UN	DELAY	VERZOE.-SCHLEIFE EINFUEGEN	
0014	0009	D1					RRL,R1		DIE 1 IM BIN.WERT R1 NACH LINKS	
0015	000A	1F	00	04			BCTA,UN	LOOP	INH.VON R1 ANZEIGEN	
0016	000D	06	FF			DELAY	LODI,R2	FF	VERZOE.-ZEIT EINSTELLEN	
0017	000F	07	FF			DELAY2	LODI,R3	FF	VERZOE.-ZEIT EINSTELLEN	
0018	0011	FB	7E			DELAY3	BDRR,R3	DELAY3	R3 BIS 0 ZURUECKZAEHLEN	

0019	0013	FA	7A				BDRR,R2	DELAY2	R2 BIS 0 ZURUECKZAEHLEN	
0020	0015	17					RETC,UN			
0021	0016								** WENN SIE DEN WERT VON R2 UND/ODER R3 VERAENDERN,	
0022	0016								** WIRD DIE LAUFGESCHWINDIGKEIT HOEHER.	

Voltmeter mit dualer Anzeige

LINE	ADDR	B1	B2	B3	B4	LABEL	OPCODE	OPERAND	COMMENTS	PAGE
0001	0000								** PROGRAMM: VOLTMETER	0001
0002	0000								** HARDWARE: GRUNDVERSION + A/D-WANDLER	
0003	0000								**	
0004	0000								- 1 MHZ	
0005	0000								** FUNKTION: PROGRAMM FRAGT DEN JEWEILIGEN WERT	
0006	0000								** DES A/D-WANDLERS AB UND ZEIGT IHN	
0007	0000								** HEXADEZIMAL AUF DEM AUSGANGSPORT AN	
0008	0000						ORG	0000		
0009	0000	55	0A			START	REDE,R1	A/D	WERT VON A/D-WANDLER HOLEN	
0010	0002	D5	09				WRTE,R1	PORT	A/D-WERT ZUM PORT SENDEN	
0011	0004	1B	7A				BCTR,UN	START	ZUM ANFANG VERZWEIGEN	

Denkzeitbegrenzer

LINE	ADDR	B1	B2	B3	B4	LABEL	OPCODE	OPERAND	COMMENTS	PAGE
0001	0000								** PROGRAMM: DENKZEITBEGRENZER	0001
0002	0000								** HARDWARE: GRUNDVERSION - 1 MHZ	
0003	0000								**	
0004	0000								** FUNKTION: PROGRAMM KANN ALS DENKZEITBEGRENZER DIENEN.	
0005	0000								** ERLISCHT DIE LETZTE LED, SO IST DIE ZEIT	
0006	0000								** ABGELAUFEN	
0007	0000						ORG	0000		
0008	0000	75	08			START	CPSL	WC	UEBERTRAG ABSCHALTEN	
0009	0002	77	02				PPSL	COM	PSL AUF LOGISCHEN VERGLEICH	
0010	0004	04	FF				LODI,R0	FF	FF NACH R0=ALLE LED EIN	
0011	0006	D4	09			SUB	WRTE,R0	PORT	INH.VON R0 ZUM PORT SENDEN	
0012	0008	3F	00	15			BSTA,UN	DELAY	VERZOE.-SCHLEIFE ANSPRINGEN	
0013	000B	D0					RRL,R0		EINSEN IN R0 NACH RECHTS	
0014	000C	A4	01				SUBI,R0	01	OBERSTE LED AUS	
0015	000E	E4	00				COMI,R0	00	ALLE LED AUS?	
0016	0010	1C	00	22			BCTA,EQ	END	WENN JA, ENDE	
0017	0013	1B	71				BCTR,UN	SUB	ZURUECK IN SCHLEIFE	
0018	0015								** VERZOEGERUNGSSCHLEIFE, DIE AUS 3 KREISEN BESTEHT UND	
0019	0015								** ENTSPRECHEND LAENGERE VERZOEGERUNGSZEITEN ZULAESST.	
0020	0015	05	05			DELAY	LODI,R1	05	VERZOE.-ZEIT EINSTELLEN	
0021	0017	06	FF			DELAY2	LODI,R2	FF	VERZOE.-ZEIT EINSTELLEN	
0022	0019	07	FF			DELAY3	LODI,R3	FF	VERZOE.-ZEIT EINSTELLEN	
0023	001B	FB	7E			DELAY4	BDRR,R3	DELAY4	R3 BIS 0 ZURUECKZAEHLEN	
0024	001D	FA	7A				BDRR,R2	DELAY3	R2 BIS 0 ZURUECKZAEHLEN	

```

0025 001F F9 76      BDRR,R1  DELAY2  R1 BIS 0 ZURUECKZAEHLEN
0026 0021 17          RETC,UN   ZURUECK INS HAUPTPROGRAMM SPRINGEN
0027 0022 D4 09      END  WRTE,R0  PORT  LETZTE LED LOESCHEN
0028 0024 40          HALT
0029 0025            ** DIE DENKZEIT KANN DURCH VERAENDERUNG VON R1,R2
0030 0025            ** ODER R3 VERLAENGERT ODER VERKUERZT WERDEN.

```

Verkehrsampel 2

```

LINE  ADDR B1 B2 B3 B4 LABEL  OPCODE  OPERAND  COMMENTS  PAGE 0001
0001  0000            ** PROGRAMM: VERKEHRSAMPEL 2
0002  0000            ** HARDWARE: GRUNDVERSION - 1 MHZ
0003  0000            **
0004  0000            ** FUNKTION: DAS PROGRAMM SIMULIERT EINE VERKEHRSAMPEL. DIE
0005  0000            ** OBEREN 3 PORT-LED STELLEN DIE HAUPTSTRASSEN-
0006  0000            ** AMPEL DAR, DIE UNTEREN 3 DIE QUERSTRASSENAMPEL
0007  0000            ORG      0000
0008  0000 04 24      START  LODI,R0  24      HAUPTSTR.GRUEN,QUERSTR.ROT
0009  0002 D4 09          WRTE,R0  PORT  INH.VON R0 ZUM PORT SENDEN
0010  0004 05 0A          LODI,R1  0A      ZEIT FUER GRUENPHASE HAUPTSTR.
0011  0006 3B 1A          BSTR,UN  DELAY  ZUR ZEITSCHLEIFE SPRINGEN
0012  0008 04 42          LODI,R0  42      GELBPHASE 1
0013  000A D4 09          WRTE,R0  PORT  INH.VON R0 ZUM PORT SENDEN
0014  000C 05 03          LODI,R1  03      ZEIT FUER GELBPHASE
0015  000E 3B 12          BSTR,UN  DELAY  ZUR ZEITSCHLEIFE SPRINGEN
0016  0010 04 B1          LODI,R0  B1      HAUPTSTR.ROT,QUERSTR.GRUEN
0017  0012 D4 09          WRTE,R0  PORT  INH.VON R0 ZUM PORT SENDEN
0018  0014 05 06          LODI,R1  06      ZEIT FUER GRUENPHASE QUERSTR.
0019  0016 3B 0A          BSTR,UN  DELAY  ZUR ZEITSCHLEIFE SPRINGEN
0020  0018 04 42          LODI,R0  42      GELBPHASE 2
0021  001A D4 09          WRTE,R0  PORT  INH.VON R0 ZUM PORT SENDEN
0022  001C 05 03          LODI,R1  03      ZEIT FUER GELBPHASE
0023  001E 3B 02          BSTR,UN  DELAY  ZUR ZEITSCHLEIFE SPRINGEN
0024  0020 1B 5E          BCTR,UN  START  ZURUECK ZUM ANFANG
0025  0022            ** VERZOEGERUNGSSCHLEIFE: BEI DIESER SCHLEIFE WIRD DAS REGISTER
0026  0022            ** R1 VOR ANSPRINGEN DER SCHLEIFE GESETZT, SO DASS VOM HAUPT-
0027  0022            ** PROGRAMM AUS DIE LAENGE DER VERZOEGERUNGSSCHLEIFE BESTIMMT
0028  0022            ** WERDEN KANN.
0029  0022 06 FF          DELAY  LODI,R2  FF      VERZOEGER.ZEIT EINSTELLEN
0030  0024 07 FF          DELAY2  LODI,R3  FF      VERZOEGER.ZEIT EINSTELLEN
0031  0026 FB 7E          DELAY3  BDRR,R3  DELAY3  R3 AUF 0 ZURUECKZAEHLEN
0032  0028 FA 7A          BDRR,R2  DELAY2  R2 AUF 0 ZURUECKZAEHLEN
0033  002A F9 76          BDRR,R1  DELAY  R1 AUF 0 ZURUECKZAEHLEN
0034  002C 17          RETC,UN
0035  002D            ** DAS PROGRAMM WIRKT REALISTISCHER, WENN AN DEN PORT-
0036  002D            ** STECKER ROTE,GELBE UND GRUENE LED ANGESCHLOSSEN WER-
0037  002D            ** DEN. DIE WERTE FUER DIE BEIDEN GRUENPHASEN UND DIE
0038  002D            ** GELBPHASE KOENNEN UNTERSCHIEDLICH EINGESTELLT WER-
0039  002D            ** DEN.

```

VU-Meter

FILE 'PROG18' AS ASSEMBLED BY SYSTEM ON 07-03-84

PAGE 0002

```

LINE  ADDR B1 B2 B3 B4 LABEL  OPCODE  OPERAND  COMMENTS
0001  0000            ** PROGRAMM: VU-METER
0002  0000            ** HARDWARE: GRUNDVERSION + A/D-WANDLER
0003  0000            ** - 1MHZ
0004  0000            **
0005  0000            ** FUNKTION: AN DEN EINGANG DES A/D-WANDLERS WIRD DAS
0006  0000            ** GLEICHGERICHTETE LAUTSPRECHERSIGNAL EINES KOFFER-
0007  0000            ** RADIOS ODER KASSETTENREKORDERES ANGESCHLOSSEN. DIE
0008  0000            ** LAUTSTAERKE WIRD DABEI AUF DEM PORT IN EINEN BALKEN
0009  0000            ** WECHSELNDE HOEHE UMGESETZT.
0010  0000            ORG      0000
0011  0000 55 0A          START  REDE,R1  0A      WERT VON A/D-WANDLER HOLEN
0012  0002 E5 7F          COMI,R1  7F      WERT AUF 7F PRUEFEN
0013  0004 99 04          BCFR,GT  BAR1    WENN NICHT GROESSER,WEITER SUCHEN
0014  0006 05 FF          LODI,R1  FF      VOLLE BALKENLAENGE
0015  0008 1B 3A          BCTR,UN  BAR9    ANZEIGEN
0016  000A E5 3F          BAR1  COMI,R1  3F      WERT AUF 3F PRUEFEN
0017  000C 99 04          BCFR,GT  BAR2    WENN NICHT GROESSER,WEITER SUCHEN
0018  000E 05 FE          LODI,R1  FE      BALKEN VOLLE LAENGE-1
0019  0010 1B 32          BCTR,UN  BAR9    ANZEIGEN
0020  0012 E5 1F          BAR2  COMI,R1  1F      BESCHREIBUNG
0021  0014 99 04          BCFR,GT  BAR3    WIE
0022  0016 05 FC          LODI,R1  FC      OBERN
0023  0018 1B 2A          BCTR,UN  BAR9
0024  001A E5 0F          BAR3  COMI,R1  0F
0025  001C 99 04          BCFR,GT  BAR4
0026  001E 05 F8          LODI,R1  F8
0027  0020 1B 22          BCTR,UN  BAR9
0028  0022 E5 07          BAR4  COMI,R1  07
0029  0024 99 04          BCFR,GT  BAR5
0030  0026 05 F0          LODI,R1  F0
0031  0028 1B 1A          BCTR,UN  BAR9
0032  002A E5 03          BAR5  COMI,R1  03
0033  002C 99 04          BCFR,GT  BAR6
0034  002E 05 E0          LODI,R1  E0
0035  0030 1B 12          BCTR,UN  BAR9
0036  0032 E5 01          BAR6  COMI,R1  01
0037  0034 99 04          BCFR,GT  BAR7
0038  0036 05 C0          LODI,R1  C0
0039  0038 1B 0A          BCTR,UN  BAR9
0040  003A E5 00          BAR7  COMI,R1  00
0041  003C 99 04          BCFR,GT  BAR8
0042  003E 05 80          LODI,R1  80
0043  0040 1B 02          BCTR,UN  BAR9
0044  0042 05 00          BAR8  LODI,R1  00
0045  0044 D5 09          BAR9  WRTE,R1  09      WERT ZUM PORT SENDEN
0046  0046 1F 00 00          BCTA,UN  START

```


Laufschrift

```

LINE  ADDR B1 B2 B3 B4 LABEL  OPCODE  OPERAND  COMMENTS  PAGE 0001
0001  0000          ** PROGRAMM: LAUFSCHRIFT
0002  0000          ** HARDWARE: GRUNDVERSION + HEX-TASTATUR/ANZEIGE
0003  0000          **
0004  0000          **
0005  0000          ** FUNKTION: DAS PROGRAMM LAESST EINEN AB SPEICHERSTELLE
0006  0000          ** 0A00 EINGEGEBENEN TEXT ALS LAUFSCHRIFT UEBER
0007  0000          ** DIE ANZEIGE LAUFEN. MIT DEM VORGEgebenEN TEXT
0008  0000          ** STELLT SICH DER COMPUTER SELBER VOR
0009  08FE          ORG      08FE
0010  08FE 00 00    TPTR   RES      2
0011  0900 BB 95    START  ZBSR     *CDIS   ANZEIGE LOESCHEN
0012  0902 04 0A          LODI,R0  0A     TEXT-
0013  0904 CC 08 FE          STRA,R0  TPTR   ANFANGS-
0014  0907 04 00    START2 LODI,R0  00     ADRESSE
0015  0909 CC 08 FF          STRA,R0  TPTR+1 SETZEN
0016  090C 05 00    SHIFT  LODI,R1  00     DISPLAY
0017  090E 0D 28 0D    SHIFT2 LODA,R1  DISBUF,+ EINE
0018  0911 CD 68 0C          STRA,R1  DISBUF-1,I STELLE
0019  0914 E5 05          COMI,R1  05     NACH LINKS
0020  0916 98 76          BCFR,EQ  SHIFT2  VERSCHIEBEN
0021  0918 0F 88 FE          LODA,R3  *TPTR  NAECHSTES ZEICHEN
0022  091B E7 FF          COMI,R3  FF     TEXT-ENDE?
0023  091D 1C 09 07          BCTA,EQ  START2  WENN JA, NACH TEXTANFANG
0024  0920 BB 85          ZBSR     *CONV   IN 7 SEG.M.WANDELN
0025  0922 CF 08 12          STRA,R3  DISBUF+5 ZEICHEN NACH ANZEIGESP.
0026  0925 0D 08 FF          LODA,R1  TPTR+1  TEXTZAHLER
0027  0928 85 01          ADDI,R1  01     UM EINS
0028  092A CD 08 FF          STRA,R1  TPTR+1  ERHOEHEN
0029  092D 06 FF          LODI,R2  FF     VERZOEG.-ZEIT EINSTELLEN
0030  092F 05 07    SHOW2  LODI,R1  07     VERZOEG.-ZEIT EINSTELLEN
0031  0931 BB 8B    SHOW   ZBSR     *DIS    7 SEGMENTANZEIGE AKTIVIEREN
0032  0933 F9 7C          BDRR,R1  SHOW   INNERE VERZOEG.SCHLEIFE
0033  0935 FA 78          BDRR,R2  SHOW2  AEUSSERE VERZOEG.SCHLEIFE
0034  0937 1F 09 0C          BCTA,UN  SHIFT  NAECHSTES ZEICHEN
0035  093A          ** GEBEN SIE AB SPEICHERSTELLE 0A00 EIGENEN
0036  093A          ** TEXT EIN. DIE SPEICHERSTELLE NACH DEM
0037  093A          ** LETZTEN TEXTZEICHEN MUSS 'FF' SEIN! DIE
0038  093A          ** LAUFSCHRIFT WIRD SCHNELLER ODER LANGSAMER
0039  093A          ** WENN MAN SPEICHERSTELLE 092E ODER 0930 VER-
0040  093A          ** AENDERT
0041  093A          ** JETZT FOLGT DER TEXTBEREICH          **
0042  0A00          ORG      0A00
0043  0A00 12 0C 11 2C    DATA   12,0C,11,2C,0B,12,17,2C,0D,0E,1B,2C
0044  0A04 08 12 17 2C
0045  0A08 0D 0E 1B 2C
0046  0A0C 02 06 05 00    DATA   02,06,05,00,2C,16,12,17,12,16,0A,15,2B
0047  0A10 2C 16 12 17
0048  0A14 12 16 0A 15
0049  0A18 2B
0050  0A19 0C 18 16 19    DATA   0C,18,16,19,1E,1D,0E,1B,2C,2C,2C,2C,FF

```

```

0051  0A1D 1E 1D 0E 1B
0052  0A21 2C 2C 2C 2C
0053  0A25 2C FF

```

Würfelspiel

```

LINE  ADDR B1 B2 B3 B4 LABEL  OPCODE  OPERAND  COMMENTS  PAGE 0001
0001  0000          ** PROGRAMM: WUERFELSPIEL
0002  0000          ** HARDWARE: GRUNDVERSION + HEX-TASTATUR/ANZEIGE
0003  0000          **
0004  0000          **          SENSE ANSCHLUSS(PIN 6 DER 7-POLBUCHSE AUF DER CPU)
0005  0000          **          MIT 50 HZ AUSGANG DER NETZPLATINE VERBINDEN
0006  0000          ** FUNKTION: DAS PROGRAMM ERZEUGT DREI ZUFALLSZAHLEN ZWISCHEN 1 UND 6
0007  0000          ** UND ZEIGT SIE AUF DER 7 SEG.M.ANZEIGE AN. MAN KANN DAMIT
0008  0000          ** ALSO EIN WUERFELSPIEL MIT 3 WUERFELN SIMULIEREN
0009  0900          ORG      0900
0010  0900 BB 95    START  ZBSR     *CDIS   ANZEIGE LOESCHEN
0011  0902 04 00          LODI,R0  0       EINE 0 ERZEUGEN
0012  0904 CC 08 1B          STRA,R0  WBUF+6  DATA1 AUF 0 SETZEN
0013  0907 3F 09 5E          BSTA,UN  RANDOM  ZUFALLSZAHL ERMITTELN
0014  090A 3F 09 30          BSTA,UN  SUMME   SUMMENFELD ERZEUGEN
0015  090D BB 85          ZBSR     *CONV   WUERFELWERT AUF 7 SEG.M.UMSETZ.
0016  090F CF 08 0D          STRA,R3  DISBUF  ZUR ANZEIGE SENDEN
0017  0912 BB 8D          ZBSR     *KIN   AUF EINGABE WARTEN
0018  0914 3F 09 5E          BSTA,UN  RANDOM  2.ZUFALLSZAHL ERMITTELN
0019  0917 3F 09 30          BSTA,UN  SUMME   SUMMENFELD ERZEUGEN
0020  091A BB 85          ZBSR     *CONV   WUEFELWERT IN 7 SEG.M.UMSETZEN
0021  091C CF 08 0E          STRA,R3  DISBUF+1 ZUR ANZEIGE SENDEN
0022  091F BB 8D          ZBSR     *KIN   AUF EINGABE WARTEN
0023  0921 3B 3B          BSTR,UN  RANDOM  3.WUERFELWERT ERMITTELN
0024  0923 3F 09 30          BSTA,UN  SUMME   SUMMENFELD ERZEUGEN
0025  0926 BB 85          ZBSR     *CONV   3.WUERFELWERT AUF 7.SEG.M.UMSETZEN
0026  0928 CF 08 0F          STRA,R3  DISBUF+2 ZUR ANZEIGE SENDEN
0027  092B BB 8D          ZBSR     *KIN   AUF EINGABE WARTEN
0028  092D 1F 09 00          BCTA,UN  START   ZUM ANFANG ZURUECK
0029  0930 CF 08 1C    SUMME  STRA,R3  WBUF+7  R3 SICHERN
0030  0933 BF 08 1B          ADDA,R3  WBUF+6  ZUM SUMMENSPEICHER ADDIEREN
0031  0936 CF 08 1B          STRA,R3  WBUF+6  SUMME SPEICHERN
0032  0939 20          EDRZ,R0          R0 AUF 0
0033  093A CC 08 2B          STRA,R0  RSLT   RSLT AUF 0
0034  093D CF 08 2C          STRA,R3  RSLT+1 FUER DEZ.UMWANDLUNG BEREITSTELLEN
0035  0940 BB 83          ZBSR     *HEXDEZ DEZIMALWERT ERZEUGEN
0036  0942 0F 08 30          LODA,R3  BCDD+2  DEZ.WERT LADEN
0037  0945 CF 08 01          STRA,R3  DATA2  ZUR SEPAR.VORBEREITEN
0038  0948 BB 87          ZBSR     *SEP   IN 2 ZIFFERN WANDELN
0039  094A 0F 08 01          LODA,R3  DATA2  1.DEZ.STELLE HOLEN
0040  094D BB 85          ZBSR     *CONV   IN 7 SEG.M. WANDELN
0041  094F CF 08 11          STRA,R3  DISBUF+4 ZUR ANZEIGE SENDEN
0042  0952 0F 08 02          LODA,R3  DATA3  2.DEZ.STELLE HOLEN
0043  0955 BB 85          ZBSR     *CONV   IN 7 SEG.M. WANDELN
0044  0957 CF 08 12          STRA,R3  DISBUF+5 ZUR ANZEIGE SENDEN

```

0045	095A 0F 08 1C	LODA,R3	WBUF+7	R3 WIEDERGWINNEN
0046	095D 17	RETC,UN		ZURUECK INS HAUPTPROGRAMM
0047	095E 12	RANDOM	SPSU	INHALT VON PSU NACH R0 LADEN
0048	095F 44 80	ANDI,R0	80	NUR HOECHSTES BIT BETRACHTEN
0049	0961 CC 08 15	STRA,R0	WBUF	SENSE ZUSTAND SPEICHERN
0050	0964 07 00	NEW	LODI,R3 00	AUGEN-ZAEHLER SETZEN
0051	0966 87 01	LOOP	ADDI,R3 01	AUGEN-ZAEHLER UM 1 ERHOEHEN
0052	0968 E7 07	COMI,R3	07	WUERFEL WERT GROESSER 6?
0053	096A 18 78	BCTR,EQ	NEW	VON VORN ZAEHLEN
0054	096C 12	SPSU		INHALT VON PSU NACH R0 LADEN
0055	096D 44 80	ANDI,R0	80	NUR HOECHSTES BIT BETRACHTEN
0056	096F EC 08 15	COMA,R0	WBUF	SENSE ZUSTAND VERAENDERT?
0057	0972 18 72	BCTR,EQ	LOOP	WENN NEIN, WEITERZAEHLEN
0058	0974 17	RETC,UN		ZURUECK INS HAUPTPROGRAMM
0059	0975	**		DIE RANDOM ROUTINE LAESST SICH IMMER DANN VERWENDEN, WENN
0060	0975	**		ZUFALLSZAHLEN ERZEUGT WERDEN SOLLN. DURCH AENDERUNG DES
0061	0975	**		WERTES IM COMPARE BEFEHL DER RANDOM ROUTINE LASSEN SICH
0062	0975	**		AUCH ANDERE ZAHLEN ERZEUGEN

Reaktionstest

LINE	ADDR	B1	B2	B3	B4	LABEL	OPCODE	OPERAND	COMMENTS	PAGE
0001	0900					ORG	0900			0001
0002	0900					**	PROGRAMM: REAKTIONSTEST			
0003	0900					**	HARDWARE: GRUNDVERSION + HEX-TASTATUR/ANZEIGE			
0004	0900					**	- 1MHZ			
0005	0900					**	DER SENSE-EINGANG (PIN 6 DER RECORDER-BUCHSE) IST MIT DEM			
0006	0900					**	50 HZ AUSGANG DER NETZTEIL-PLATINE ZU VERBINDEN			
0007	0900					**				
0008	0900					**	FUNKTION: DAS PROGRAMM TESTET DAS REAKTIONSVERMOEGEN.WENN EINE			
0009	0900					**	ZAHL ZWISCHEN 0 UND F AUFLEUCHTET, MUSS SCHNELLSTENS			
0010	0900					**	DIE ENTSPRECHENDE TASTE GEDRUECKT WERDEN. DAS PROGRAMM			
0011	0900					**	ERMITTELT DIE ERFOLGREICHEN VERSUCHE UND DIE			
0012	0900					**	FEHLER UND ZEIGT DAS ERGEBNIS AN			
0013	0900					ORG	0900			
0014	0900 05 1C	START	LODI,R1	1C		VAR.VERZOEGL.SCHLEIFE SETZEN				
0015	0902 CD 08 2D		STRA,R1	082D		VERZ,WERT SPEICHERN				
0016	0905 04 00		LODI,R0	00		EINE 0 ERZEUGEN				
0017	0907 CC 08 2E		STRA,R0	082E		SPEICHERSTELLE LOESCHEN				
0018	090A CC 08 2F		STRA,R0	082F		SPEICHERSTELLE LOESCHEN				
0019	090D 77 02		PPSL	COM		LOGICAL COMPARE EINSTELLEN				
0020	090F 12	RANDOM	SPSU			INHALT VON PSU NACH R0 LADEN				
0021	0910 44 80		ANDI,R0	80		NUR HOECHSTES BIT BETRACHTEN				
0022	0912 CC 08 16		STRA,R0	WBUF+1		SENSE ZUSTAND SPEICHERN				
0023	0915 07 FF	NEW	LODI,R3	FF		AUGEN ZAEHLER SETZEN				
0024	0917 87 01	LOOP	ADDI,R3	01		AUGEN-ZAEHLER UM 1 ERHOEHEN				
0025	0919 E7 10		COMI,R3	10		ZUFALLSWERT >10?				
0026	091B 18 78		BCTR,EQ	NEW		VON VORN ZAEHLEN				
0027	091D 12		SPSU			INHALT VON PSU NACH R0 LADEN				
0028	091E 44 80		ANDI,R0	80		NUR HOECHSTES BIT BETRACHTEN				
0029	0920 EC 08 16		COMA,R0	WBUF+1		SENSE ZUSTAND VERAENDERT?				

0030	0923 18 72		BCTR,EQ	LOOP		WENN NEIN, WEITERZAEHLEN
0031	0925 BB 95	PLAY	ZBSR	*CDIS		ANZEIGE LOESCHEN
0032	0927 CF 08 2C		STRA,R3	082C		INH VON R3 SPEICHERN
0033	092A BB 85		ZBSR	*CONV		WERT IN 7 SEGM.DARST.WANDELN
0034	092C 0F 08 12		STRA,R3	DISBUF+5		ZUR ANZEIGE SENDEN
0035	092F 06 FF		LODI,R2	FF		VERZ.SCHLEIFE INITIALISIEREN
0036	0931 0D 08 2D	GET	LODA,R1	082D		VAR.SCHLEIFENWERT LADEN
0037	0934 BB 88	GET2	ZBSR	*DIS		ANZEIGE AKTIVIEREN
0038	0936 57 17		REDE,R3	17		KEYBOARD LESEN
0039	0938 F7 80		TMI ,R3	80		STROBE=1?
0040	093A 98 11		BCFR,EQ	GET3		WENN KEINE TASTE GEDRUECKT,WEITER
0041	093C CF 08 15		STRA,R3	WBUF		TAST.WERT ZUR ENTPRELLUNG SPEICHERN
0042	093F 57 17		REDE,R3	17		KEYBOARD ERNEUT LESEN
0043	0941 EF 08 15		COMA,R3	WBUF		TASTENWERT NOCH GLEICH?
0044	0944 98 6E		BCFR,EQ	GET2		WENN GEPRELLT,ERNEUT LESEN
0045	0946 47 0F		ANDI,R3	0F		STROBE ABTRENKEN
0046	0948 EF 08 2C		COMA,R3	082C		TASTATURWERT=ZUFALLSWERT?
0047	094B 18 0E		BCTR,EQ	RIGHT		WENN JA, ERGEBNIS ERMITT.
0048	094D F9 65	GET3	BDRR,R1	GET2		WENN NEIN, WEITER WARTEN
0049	094F FA 60		BDRR,R2	GET		VERZOEGERUNG
0050	0951 0F 08 2E	FALSCH	LODA,R3	082E		FEHLERSPEICHER LADEN
0051	0954 87 01		ADDI,R3	01		UM 1 ERHOEHEN
0052	0956 CF 08 2E		STRA,R3	082E		FEHLER SPEICHERN
0053	0959 1B 10		BCTR,UN	SHOW		ZUR ANZEIGE VERZWEIGEN
0054	095B 0F 08 2F	RIGHT	LODA,R3	082F		RICHTIG-SPEICHER LADEN
0055	095E 87 01		ADDI,R3	01		UM 1 ERHOEHEN
0056	0960 CF 08 2F		STRA,R3	082F		RICHTIG-ERGENN.SPEICHERN
0057	0963 0D 08 2D		LODA,R1	082D		VAR.VERZOEGL.WERT LADEN
0058	0966 A5 01		SUBI,R1	01		VERZOEGL.VERKUERZEN
0059	0968 CD 08 2D		STRA,R1	082D		VAR.VERZOEGL.WERT SPEICHERN
0060	096B BB 95	SHOW	ZBSR	*CDIS		ANZEIGE LOESCHEN
0061	096D 0F 08 2E		LODA,R3	082E		FEHLERSPEICHER HOLEN
0062	0970 E7 0F		COMI,R3	0F		16 FEHLER?
0063	0972 1C 09 B1		BCTA,EQ	END2		WENN JA, VERLOREN ANZEIGEN
0064	0975 BB 85		ZBSR	*CONV		IN 7 SEGM.WANDELN
0065	0977 CF 08 0F		STRA,R3	DISBUF+2		ZUR ANZEIGE SENDEN
0066	097A 0F 08 2F		LODA,R3	082F		RICHTIG-SPEICHER LADEN
0067	097D E7 0F		COMI,R3	0F		16 RICHTIGE ANTW.?
0068	097F 18 12		BCTR,EQ	END1		WENN JA, GEWONN.ANZEIGEN
0069	0981 BB 85		ZBSR	*CONV		IN 7 SEGM.WANDELN
0070	0983 CF 08 0D		STRA,R3	DISBUF		ZUR ANZEIGE SENDEN
0071	0986 05 FF		LODI,R1	FF		1.VERZOEGL.EINSTELLEN
0072	0988 06 10	SHOW2	LODI,R2	10		2.VERZOEGL.EINSTELLEN
0073	098A BB 88	SHOW3	ZBSR	*DIS		ANZEIGE AKTIVIEREN
0074	098C FA 7C		BDRR,R2	SHOW3		VERZOEGERUNGSSCHLEIFE
0075	098E F9 78		BDRR,R1	SHOW2		DURCHLAUFEN
0076	0990 1F 09 0F		BCTA,UN	RANDOM		ZURUECK ZUM START
0077	0993 07 7D	END1	LODI,R3	7D		WORT
0078	0995 CF 08 0D		STRA,R3	DISBUF		'GEWONNEN'
0079	0998 07 79		LODI,R3	79		
0080	099A CF 08 0E		STRA,R3	DISBUF+1		UND
0081	099D 07 3E		LODI,R3	3E		ZUR
0082	099F CF 08 0F		STRA,R3	DISBUF+2		ANZEIGE
0083	09A2 07 5C		LODI,R3	5C		SENDEN
0084	09A4 CF 08 10		STRA,R3	DISBUF+3		

```

0085 09A7 07 54      LODI,R3  54
0086 09A9 CF 08 11  STRA,R3  DISBUF+4
0087 09AC CF 08 12  STRA,R3  DISBUF+5
0088 09AF 1B 1E      BCTR,UN  END3
0089 09B1 07 3C      END2  LODI,R3  3C      WORT
0090 09B3 CF 08 0D  STRA,R3  DISBUF 'VERLOREN'
0091 09B6 07 79      LODI,R3  79      GENERIEREN
0092 09BB CF 08 0E  STRA,R3  DISBUF+1  UND
0093 09BB 07 50      LODI,R3  50      ZUR
0094 09BD CF 08 0F  STRA,R3  DISBUF+2  ANZEIGE
0095 09C0 07 38      LODI,R3  38      SENDEN
0096 09C2 CF 08 10  STRA,R3  DISBUF+3
0097 09C5 07 5C      LODI,R3  5C
0098 09C7 CF 08 11  STRA,R3  DISBUF+4
0099 09CA 07 50      LODI,R3  50
0100 09CC CF 08 12  STRA,R3  DISBUF+5
0101 09CF BB 8E      END3  ZBSR  *DIS  ANZEIGE AKTIVIEREN
0102 09D1 1B 7C      BCTR,UN  END3
0103 09D3      ** DER SCHWIERIGKEITSGRAD KANN DURCH DEN WERT IN 0901
0104 09D3      ** EINGESTELLT WERDEN.

```

Stoppuhr

```

LINE ADDR B1 B2 B3 B4 LABEL OPCODE OPERAND COMMENTS PAGE 0001
0001 0000      ** PROGRAMM: STOPUHR
0002 0000      ** HARDWARE: GRUNDVERSION + HEX-TASTATUR/ANZEIGE
0003 0000      **
0004 0000      ** DER SENSE-EINGANG (PIN 6 DER RECORDER-BUCHSE) IST MIT DEM 50 HZ
0005 0000      ** AUSGANG DER NETZTEIL-PLATINE ZU VERBINDEN
0006 0000      **
0007 0000      ** FUNKTION: DAS PROGRAMM ARBEITET ALS STOPUHR MIT EINER AUFLUESUNG
0008 0000      ** VON 1/100 SEC. DIE HIGH/LOW UND LOW/HIGH UEBERGANGE DER NETZ-
0009 0000      ** FREQUENZ WERDEN DABEI ALS ZEITTAKT VERWENDET. BEI 50 HZ ERGEBEN SICH
0010 0000      ** ALSO JE 50 HIGH-LOW UND LOW-HIGH UEBERGAENGE= 100 TAKT-
0011 0000      ** IMPULSE.
0012 0900      ORG      0900
0013 0900 77 02  RESET  PPSL  COM  PROGRAMM-STATUS AUF LOGICAL COMPARE
0014 0902 3F 09 22  BSTA,UN  RESET3  ANZEIGE AUF 0
0015 0905 3F 09 15  BSTA,UN  RESET2  UHR AUF 0
0016 0908 BB 8D      START  ZBSR  *KIN  TASTENWERT HOLEN
0017 090A 0F 08 00  LODA,R3  DATA1  TASTENWERT NACH R3
0018 090D E7 00      COMI,R3  00      TASTE 0 GEDRUECKT?
0019 090F 1C 09 39  BCTA,EQ  RUN  WENN JA, STOPUHR ANLAUFEN LASSEN
0020 0912 1F 09 08  BCTA,UN  START  WENN NICHT, TASTATUR ABFRAGEN
0021 0915 06 03  RESET2  LODI,R2  03      INDEX-REGISTER SETZEN
0022 0917 04 00      LODI,R0  00      REGISTER 0 AUF 0
0023 0919 CE 48 FD  RESET4  STRA,R2  UHR,-  WBUF LOESCHEN
0024 091C E6 00      COMI,R2  00      ALLE WORKBUFFER GELOESCHT?
0025 091E 9C 09 19  BCFA,EQ  RESET4  WENN NICHT, WEITER LOESCHEN
0026 0921 17      RETC,UN
0027 0922 07 3F  RESET3  LODI,R3  3F      7-SEGMENT 0 LADEN

```

```

0028 0924 CF 08 0D  STRA,R3  DISBUF  ANZEIGE AUF 0
0029 0927 CF 08 0F  STRA,R3  DISBUF+2
0030 092A CF 08 11  STRA,R3  DISBUF+4
0031 092D CF 08 12  STRA,R3  DISBUF+5
0032 0930 87 80      ADDI,R3  80      DEZIMALPUNKT SETZEN
0033 0932 CF 08 0E  STRA,R3  DISBUF+1  ANZEIGE AUF 0. SETZEN
0034 0935 CF 08 10  STRA,R3  DISBUF+3
0035 0938 17      RETC,UN
0036 0939 12      RUN  SPSU
0037 093A 44 80      ANDI,R0  80      SENSE EINGANG ABFRAGEN
0038 093C CC 08 1C  STRA,R0  WBUF+7  50 HZ ZUSTAND SPEICHERN
0039 093F 56 17      RUN2  REDE,R2  17      TASTATUR ABFRAGEN
0040 0941 46 7F      ANDI,R2  7F      STROBE AUSBLENDEN
0041 0943 E6 01      COMI,R2  01      TASTE 1 GEDRUECKT?
0042 0945 1C 09 F9  BCTA,EQ  STOP  WENN JA, ZEIT STOPPEN
0043 0948 BB 8B      ZBSR  *DIS  WERT ANZEIGEN
0044 094A 12      SPSU
0045 094B 44 80      ANDI,R0  80      SENSE BIT ABFRAGEN
0046 094D EC 08 1C  COMA,R0  WBUF+7  SENSE BIT SELEKTIEREN
0047 0950 9C 09 56  BCFA,EQ  COUNT  WENN JA, ZAEHLEN
0048 0953 1F 09 3F  BCTA,UN  RUN2  WEITER WARTEN
0049 0956 CC 08 1C  COUNT  STRA,R0  WBUF+7  GEANDERTEN SENSE PEGEL SPEICHERN
0050 0959 0F 08 FF  LODA,R3  UHR+2  1/100 SEC ZAEHLER HOLEN
0051 095C 87 01      ADDI,R3  01      UM 1 ERHOEHEN
0052 095E E7 64      COMI,R3  64      100/100 SEC ERREICHT?
0053 0960 1C 09 7C  BCTA,EQ  SEC  WENN JA, SEKUNDEN WEITERZAEHLEN
0054 0963 CF 08 FF  STRA,R3  UHR+2  SONST WIEDER IN 1/100 SEC ZAEHLER SPEICHERN
0055 0966 3F 09 EB  BSTA,UN  SEPA  DEZ-WANDLUNG VORBEREITEN
0056 0969 0F 08 02  LODA,R3  DATA3  EINER LADEN
0057 096C BB 85      ZBSR  *CONV  IN 7-SEGMENT WANDELN
0058 096E CF 08 12  STRA,R3  DISBUF+5  IN ANZEIGESPEICHER UEBERTRAGEN
0059 0971 0F 08 01  LODA,R3  DATA2  ZEHNER LADEN
0060 0974 BB 85      ZBSR  *CONV  IN 7-SEGMENT WANDELN
0061 0976 CF 08 11  STRA,R3  DISBUF+4  IN ANZEIGESPEICHER UEBERTRAGEN
0062 0979 1F 09 3F  BCTA,UN  RUN2  TAKT ABFRAGEN
0063 097C 07 00      LODI,R3  00      0 LADEN
0064 097E CF 08 FF  STRA,R3  UHR+2  1/100 SEC ZAEHLER AUF 0
0065 0981 07 3F      LODI,R3  3F      7 SEGMENT 0 LADEN
0066 0983 CF 08 12  STRA,R3  DISBUF+5  ANZEIGE AUF 0
0067 0986 CF 08 11  STRA,R3  DISBUF+4
0068 0989 0F 08 FE  LODA,R3  UHR+1  SEKUNDENZAehler HOLEN
0069 098C 87 01      ADDI,R3  01      UM 1 ERHOEHEN
0070 098E E7 3C      COMI,R3  3C      60 SEKUNDEN?
0071 0990 1C 09 AE  BCTA,EQ  MIN  WENN JA, MINUTENZAehler ERHOEHEN
0072 0993 CF 08 FE  STRA,R3  UHR+1  SONST SEKUNDENZAehler WIEDER SPEICHERN
0073 0996 3F 09 EB  BSTA,UN  SEPA  DEZ.WANDLUNG VORBEREITEN
0074 0999 0F 08 02  LODA,R3  DATA3  EINER HOLEN
0075 099C BB 85      ZBSR  *CONV  IN 7 SEGMENT WANDELN
0076 099E 87 80      ADDI,R3  80      DEZIMALPUNKT SETZEN
0077 09A0 CF 08 10  STRA,R3  DISBUF+3  NACH ANZEIGESPEICHER
0078 09A3 0F 08 01  LODA,R3  DATA2  ZEHNER HOLEN
0079 09A6 BB 85      ZBSR  *CONV  IN 7 SEGMENT WANDELN
0080 09AB CF 08 0F  STRA,R3  DISBUF+2  NACH ANZEIGESPEICHER
0081 09AB 1F 09 3F  BCTA,UN  RUN2  TAKT HOLEN
0082 09AE 07 00      MIN  LODI,R3  00      0 LADEN

```


0083	09B0	CF	08	FE	STRA,R3	UHR+1	SEKUNDENZAehler AUF 0
0084	09B3	07	BF		LODI,R3	BF	0. LADEN
0085	09B5	CF	08	10	STRA,R3	DISBUF+3	MINUTENZAehler AUF 0.
0086	09B8	07	3F		LODI,R3	3F	7 SEGMA.0 LADEN
0087	09BA	CF	08	0F	STRA,R3	DISBUF+2	10 MIN.ZAEHLER AUF 0
0088	09BD	0F	08	FD	LODA,R3	UHR	MINUTENZAehler HOLEN
0089	09C0	87	01		ADDI,R3	01	UM 1 ERHOEHEN
0090	09C2	E7	64		COMI,R3	64	100 MINUTEN?
0091	09C4	9C	09	DD	BCFA,EQ	MIN2	WENN NEIN, WEITER
0092	09C7	3F	09	15	BSTA,UN	RESET2	UHRENZAehler LOESCHEN
0093	09CA	3F	09	22	BSTA,UN	RESET3	ANZEIGE AUF 0
0094	09CD	1F	09	3F	BCTA,UN	RUN2	UHR VON VORN ANLAUFEN LASSEN
0095	09D0	CF	08	FB	MIN2 STRA,R3	UHR	MINUTENZAehler SPEICHERN
0096	09D3	3F	09	1B	BSTA,UN	SEPA	DEZ.WANDLUNG VORBEREITEN
0097	09D6	0F	08	02	LODA,R3	DATA3	EINER HOLEN
0098	09D9	BB	85		ZBSR	*CONV	IN 7 SEGMENT WANDELN
0099	09DB	87	80		ADDI,R3	80	DEZIMALPUNKT SETZEN
0100	09DD	CF	08	0E	STRA,R3	DISBUF+1	NACH ANZEIGESPEICHER
0101	09E0	0F	08	01	LODA,R3	DATA2	ZEHNER HOLEN
0102	09E3	BB	85		ZBSR	*CONV	IN 7 SEGMENT WANDELN
0103	09E5	CF	08	0D	STRA,R3	DISBUF	NACH ANZEIGESPEICHER
0104	09EB	1F	09	3F	BCTA,UN	RUN2	TAKT HOLEN
0105	09EB	CF	08	2C	SEPA STRA,R3	RSLT+1	HEX-ZAHL IN RSLT+1 SPEICHERN
0106	09EE	BB	83		ZBSR	*HEXDEZ	IN DEZIMALZAHL WANDELN
0107	09F0	0F	08	30	LODA,R3	BCDD+2	DEZIMALZAHL HOLEN
0108	09F3	CF	08	01	STRA,R3	DATA2	NACH DATA2 UEBERTRAGEN
0109	09F6	BB	87		ZBSR	*SEP	NIBBLES IN 2 BYTES TRENNEN
0110	09F8	17			RETC,UN		ZURUECK ZUM HAUPTPROGRAMM
0111	09F9	BB	8D		STOP ZBSR	*KIN	AUF TASTENWERT WARTEN
0112	09FB	0F	08	00	LODA,R3	DATA1	TASTENWERT HOLEN
0113	09FE	E7	02		COMI,R3	02	TASTE 2 GEDRUECKT?
0114	0A00	1C	09	00	BCTA,EQ	RESET	RESET AUSFUEHREN
0115	0A03	E7	00		COMI,R3	00	TASTE 0 GEDRUECKT
0116	0A05	1C	09	3F	BCTA,EQ	RUN2	ERNEUT STARTEN
0117	0A08	1F	09	F9	BCTA,UN	STOP	SONST IN STOP SCHLEIFE BLEIBEN

Die Stoppuhr wird mit einem Druck auf die Taste „0“ gestartet und mit einem Druck auf die Taste „1“ gestoppt. Mehrere Stoppzeiten werden addiert. Die Uhr wird mit RST und RUN auf 00.00.00 zurückgestellt.

Digitaluhr

LINE	ADDR	B1	B2	B3	B4	LABEL	OPCODE	OPERAND	COMMENTS	PAGE
0001	0000					** PROGRAMM: DIGITALUHR				0001
0002	0000					** HARDWARE: GRUNDVERSION + HEX-TASTATUR/ANZEIGE				
0003	0000					**			- 1MHZ	
0004	0000					** DER SENSE EINGANG (PIN 6 DER RECORDER-BUCHSE) IST MIT				
0005	0000					** DEM 50 HZ AUSGANG DER NETZTEIL-PLATINE ZU VERBINDEN				
0006	0000					**				

0007	0000					** FUNKTION: DAS PROGRAMM ARBEITET ALS DIGITALUHR MIT				
0008	0000					** STUNDEN, MINUTEN UND SEKUNDEN ANZEIGE. ALS TAKT DIENST DER				
0009	0000					** 50 HZ AUSGANG DER NETZTEIL-PLATINE.				
0010	0BFC					ORG	0BFC			
0011	0BFC	00	00	00	00	UHR	RES	04		
0012	0900	3F	0A	45	STELL	BSTA,UN	RESET2	UHRENZAehler LOESCHEN		
0013	0903	3F	0A	52		BSTA,UN	RESET3	ANZEIGE AUF 00.00.00		
0014	0906	07	00		STELL1	LODI,R3	00	0 LADEN		
0015	0908	CF	08	FE		STRA,R3	UHR+2	SEKUNDEN AUF 0		
0016	090B	CF	08	FF		STRA,R3	UHR+3	1/100 SEK AUF 0		
0017	090E	3F	0A	69		BSTA,UN	DELAY	WARTEN		
0018	0911	57	17			REDE,R3	17	KEYBOARD LESEN		
0019	0913	F7	80			THI ,R3	80	STROBE?		
0020	0915	9C	09	06		BCFA,EQ	STELL1	WENN NEIN,WEITER LESEN		
0021	0918	CF	08	15		STRA,R3	WBUF	WENN JA,ZEICHEN SPEICHERN		
0022	091B	57	17			REDE,R3	17	KEYBOARD LESEN		
0023	091D	EF	08	15		COMA,R3	WBUF	NOCH GLEICH?		
0024	0920	98	64			BCFR,EQ	STELL1	WENN NEIN, WEITER LESEN		
0025	0922	47	7F			ANDI,R3	7F	STROBE AUSBLENDEN		
0026	0924	E7	00			COMI,R3	00	TASTE 0 GEDRUECKT?		
0027	0926	1C	09	35		BCTA,UN	STU	WENN JA, STUNDEN STELLEN		
0028	0929	E7	01			COMI,R3	01	TASTE 1 GEDRUECKT?		
0029	092B	1C	09	4C		BCTA,EQ	MI	WENN JA, MINUTEN STELLEN		
0030	092E	E7	02			COMI,R3	02	TASTE 2:GEDRUECKT?		
0031	0930	1C	09	64		BCTA,UN	RUN	WENN JA, UHR ANLAUFEN LASSEN		
0032	0933	1B	51		STELL2	BCTR,EQ	STELL1	WENN KEINE TASTE GEDRUECKT, WEITERSUCHEN		
0033	0935	0F	08	FC	STU	LODA,R3	UHR	STUNDENZAehler LADEN		
0034	0938	87	01			ADDI,R3	01	UM 1 ERHOEHEN		
0035	093A	E7	18			COMI,R3	18	24 UHR?		
0036	093C	1B	06			BCTR,EQ	STU1	WENN JA, AUF 00 UHR SETZEN		
0037	093E	3F	0A	1E		BSTA,UN	STU2	UHRZEIT ANZEIGEN		
0038	0941	1F	09	06		BCTA,UN	STELL1	WEITER STELLEN		
0039	0944	07	00		STU1	LODI,R3	00	0 LADEN		
0040	0946	3F	0A	1E		BSTA,UN	STU2	UHRZEIT ANZEIGEN		
0041	0949	1F	09	06		BCTA,UN	STELL1	WEITER STELLEN		
0042	094C	0F	08	FD	MI	LODA,R3	UHR+1	MINUTENZAehler LADEN		
0043	094F	87	01			ADDI,R3	01	UM 1 ERHOEHEN		
0044	0951	E7	3C			COMI,R3	3C	60 MINUTEN?		
0045	0953	1C	09	5C		BCTA,EQ	MI1	WENN JA, AUF 00 MINUTEN SETZEN		
0046	0956	3F	09	DD		BSTA,UN	MI2	ANZEIGEN		
0047	0959	1F	09	06		BCTA,UN	STELL1	WEITER STELLEN		
0048	095C	07	00		MI1	LODI,R3	00	0 LADEN		
0049	095E	3F	09	DD		BSTA,UN	MI2	ANZEIGEN		
0050	0961	1F	09	06		BCTA,UN	STELL1	ZUR STELL FUNKTION		
0051	0964	12			RUN	SPSU		SENSE EINGANG ABFRAGEN		
0052	0965	44	80			ANDI,R0	80	SENSE BIT SELEKTIEREN		
0053	0967	CC	08	1C		STRA,R0	WBUF+7	50 HZ ZUSTAND SPEICHERN		
0054	096A	BB	8B		RUN2	ZBSR	*DIS	WERT ANZEIGEN		
0055	096C	57	17			REDE,R3	17	KEYBOARD LESEN		
0056	096E	47	7F			ANDI,R3	7F	STROBE AUSBLENDEN		
0057	0970	E7	03			COMI,R3	03	TASTE 3?		
0058	0972	1C	09	06		BCTA,EQ	STELL1	WENN JA, ZUR STELL-FUNKTION		
0059	0975	12				SPSU		SENSE BIT ABFRAGEN		
0060	0976	44	80			ANDI,R0	80	SENSE BIT SELEKTIEREN		
0061	0978	EC	08	1C		COMA,R0	WBUF+7	PEGELWECHSEN?		

0062	097B	9C	09	B1		BCFA,EQ	COUNT	WENN JA, ZAEHLN
0063	097E	1F	09	6A		BCTA,UN	RUN2	WEITER WARTEN
0064	0981	CC	08	1C	COUNT	STRA,R0	WBUF+7	GEAENDERTEN SENSE PEGEL SPEICHERN
0065	0984	0F	08	FF		LODA,R3	UHR+3	1/100 SEC ZAEHLER HOLEN
0066	0987	87	01			ADDI,R3	01	UM 1 ERHOEHN
0067	0989	E7	64			COMI,R3	64	100/100 SEC ERREICHT?
0068	098B	1C	09	94		BCTA,EQ	SEC	WENN JA, SEKUNDEN WEITERZAEHLN
0069	098E	CF	08	FF		STRA,R3	UHR+3	1/100 SEC ZAEHLER SPEICHERN
0070	0991	1F	09	6A		BCTA,UN	RUN2	TAKT ABFRAGEN
0071	0994	07	00		SEC	LODI,R3	00	0 LADEN
0072	0996	CF	08	FF		STRA,R3	UHR+3	1/100 SEC ZAEHLER AUF 0
0073	0999	0F	08	FE		LODA,R3	UHR+2	SEKUNDENZAehler HOLEN
0074	099C	87	01			ADDI,R3	01	UM 1 ERHOEHN
0075	099E	E7	3C			COMI,R3	3C	60 SEKUNDEN?
0076	09A0	1C	09	BE		BCTA,EQ	MIN	WENN JA, MINUTENZAehler ERHOEHN
0077	09A3	CF	08	FE		STRA,R3	UHR+2	SONST SEKUNDENZAehler WIEDER SPEICHERN
0078	09A6	3F	0A	37		BSTA,UN	SEPA	DEZIMALWANDLUNG
0079	09A9	0F	08	02		LODA,R3	DATA3	EINER HOLEN
0080	09AC	BB	85			ZBSR	*CONV	IN 7 SEGMENT WANDELN
0081	09AE	87	80			ADDI,R3	80	DEZIMALPUNKT SETZEN
0082	09B0	CF	08	12		STRA,R3	DISBUF+5	NACH ANZEIGESPEICHER
0083	09B3	0F	08	01		LODA,R3	DATA2	ZEHNER HOLEN
0084	09B6	BB	85			ZBSR	*CONV	IN 7 SEGMENT WANDELN
0085	09B8	CF	08	11		STRA,R3	DISBUF+4	NACH ANZEIGESPEICHER
0086	09BB	1F	09	6A		BCTA,UN	RUN2	TAKT HOLEN
0087	09BE	07	00		MIN	LODI,R3	00	0 LADEN
0088	09C0	CF	08	FE		STRA,R3	UHR+2	SEKUNDENZAehler AUF 0
0089	09C3	07	BF			LODI,R3	BF	0. LADEN
0090	09C5	CF	08	12		STRA,R3	DISBUF+5	MINUTENZAehler AUF 0.
0091	09C8	07	3F			LODI,R3	3F	7 SEGH.0 LADEN
0092	09CA	CF	08	11		STRA,R3	DISBUF+4	10 MIN.ZAEHLER AUF 0
0093	09CD	0F	08	FD		LODA,R3	UHR+1	MINUTENZAehler HOLEN
0094	09D0	87	01			ADDI,R3	01	UM 1 ERHOEHN
0095	09D2	E7	3C			COMI,R3	3C	60 MINUTEN?
0096	09D4	1C	09	F6		BCTA,EQ	STD	WENN JA,WEITER
0097	09D7	3F	09	DD		BSTA,UN	MI2	MINUTEN SPEICHERN
0098	09DA	1F	09	6A		BCTA,UN	RUN2	TAKT HOLEN
0099	09DD	CF	08	FD	MI2	STRA,R3	UHR+1	MINUTENZAehler SPEICHERN
0100	09E0	3F	0A	37		BSTA,UN	SEPA	DEZIMALWANDLUNG
0101	09E3	0F	08	02		LODA,R3	DATA3	EINER HOLEN
0102	09E6	BB	85			ZBSR	*CONV	IN 7 SEGMENT WANDELN
0103	09E8	87	80			ADDI,R3	80	DEZIMALPUNKT SETZEN
0104	09EA	CF	08	10		STRA,R3	DISBUF+3	NACH ANZEIGESPEICHER
0105	09ED	0F	08	01		LODA,R3	DATA2	ZEHNER HOLEN
0106	09F0	BB	85			ZBSR	*CONV	IN 7 SEGMENT WANDELN
0107	09F2	CF	08	0F		STRA,R3	DISBUF+2	NACH ANZEIGESPEICHER
0108	09F5	17				RETC,UN		RUECKSPRUNG
0109	09F6	07	00		STD	LODI,R3	00	0 LADEN
0110	09F8	CF	08	FD		STRA,R3	UHR+1	MINUTENZAehler AUF 0
0111	09FB	07	BF			LODI,R3	BF	0.LADEN
0112	09FD	CF	08	10		STRA,R3	DISBUF+3	MINUTENZAehler AUF 0
0113	0A00	07	3F			LODI,R3	3F	0 LADEN
0114	0A02	CF	08	0F		STRA,R3	DISBUF+2	10 MINUTEN ZAEHLER AUF 0
0115	0A05	0F	08	FC		LODA,R3	UHR	STUNDENZAehler HOLEN
0116	0A08	87	01			ADDI,R3	01	UM 1 ERHOEHN

0117	0A0A	E7	18			COMI,R3	18	24 STUNDEN?
0118	0A0C	9C	0A	18		BCFA,EQ	STD2	WENN NEIN, WEITER
0119	0A0F	3F	0A	45		BSTA,UN	RESET2	ALLE ZAEHLER AUF 0
0120	0A12	3F	0A	52		BSTA,UN	RESET3	ALLE ANZEIGEN AUF 0
0121	0A15	1F	09	6A		BCTA,UN	RUN2	NEU BEGINNEN BEI 00.00.00
0122	0A18	3F	0A	1E	STD2	BSTA,UN	STU2	STUNDENWERT SPEICHERN
0123	0A1B	1F	09	6A		BCTA,UN	RUN2	WEITER ZAEHLN
0124	0A1E	CF	08	FC	STU2	STRA,R3	UHR	STUNDENWERT SPEICHERN
0125	0A21	3F	0A	37		BSTA,UN	SEPA	DEZIMALWANDLUNG
0126	0A24	0F	08	02		LODA,R3	DATA3	STUNDEN LADEN
0127	0A27	BB	85			ZBSR	*CONV	IN 7-SEGMENT WANDELN
0128	0A29	87	80			ADDI,R3	80	PUNKT SETZEN
0129	0A2B	CF	08	0E		STRA,R3	DISBUF+1	IN ANZEIGE-SPEICHER UEBERTRAGEN
0130	0A2E	0F	08	01		LODA,R3	DATA2	10 STUNDEN LADEN
0131	0A31	BB	85			ZBSR	*CONV	IN 7-SEGMENT WANDELN
0132	0A33	CF	08	0D		STRA,R3	DISBUF	IN ANZEIGE-SPEICHER UEBERTRAGEN
0133	0A36	17				RETC,UN		RUECKSPRUNG
0134	0A37	CF	08	2C	SEPA	STRA,R3	RSLT+1	WERT IN RSLT+1 SPEICHERN
0135	0A3A	BB	83			ZBSR	*HEXDEZ	IN DEZIMALZAHL WANDELN
0136	0A3C	0F	08	30		LODA,R3	BCDD+2	DEZIMALWERT HOLEN
0137	0A3F	CF	08	01		STRA,R3	DATA2	IN DATA2 SPEICHERN
0138	0A42	BB	87			ZBSR	*SEP	HALBBYTES TRENNEIN
0139	0A44	17				RETC,UN		ZURUECK INS HAUPTPROGRAMM
0140	0A45	06	04			RESET2	LODI,R2	04 INDEX-REGISTER SETZEN
0141	0A47	04	00				LODI,R0	00 REGISTER 0 AUF 0
0142	0A49	CE	48	FC	RESET4	STRA,R2	UHR,-	WBUF LOESCHEN
0143	0A4C	E6	00			COMI,R2	00	ALLE WORKBUFFER GELOESCHT?
0144	0A4E	9C	0A	49		BCFA,EQ	RESET4	WENN NICHT, WEITER LOESCHEN
0145	0A51	17				RETC,UN		SONST RUECKSPRUNG
0146	0A52	07	3F		RESET3	LODI,R3	3F	7-SEGH.0 LADEN
0147	0A54	CF	08	0D		STRA,R3	DISBUF	ANZEIGE AUF 0
0148	0A57	CF	08	0F		STRA,R3	DISBUF+2	
0149	0A5A	CF	08	11		STRA,R3	DISBUF+4	
0150	0A5D	CF	08	12		STRA,R3	DISBUF+5	
0151	0A60	87	80			ADDI,R3	80	DEZIMALPUNKT SETZEN
0152	0A62	CF	08	0E		STRA,R3	DISBUF+1	ANZEIGE AUF 0. SETZEN
0153	0A65	CF	08	10		STRA,R3	DISBUF+3	
0154	0A68	17				RETC,UN		RUECKSPRUNG
0155	0A69	06	02		DELAY	LODI,R2	02	VERZOEGERUNG EINSTELLEN
0156	0A6B	05	FF		DELAY1	LODI,R1	FF	VERZOEGERUNG EINSTELLEN
0157	0A6D	BB	8B		DELAY2	ZBSR	*DIS	ANZEIGEN
0158	0A6F	F9	7C			BDRR,R1	DELAY2	R1 BIS 0 ZAEHLN
0159	0A71	FA	78			BDRR,R2	DELAY1	R0 BIS 0 ZAEHLN
0160	0A73	17				RETC,UN		RUECKSPRUNG INS HAUPTPROGRAMM
0161	0A74							** SOLL DAS PROGRAMM Z.B. ALS MODELLUHR MIT SCHNELLEREM TAKT BETRIEBEN
0162	0A74							** WERDEN, SO IST SPEICHERSTELLE 098A VON 64 ZU ERNIEDRIGEN.

Stellen der Uhr: Stunden mit der Taste „0“
 Minuten mit der Taste „1“.
 Starten: Taste „2“ bei voller Minute (Sekundenanzeige 00).

HEX-DEZ-HEX-Wandlung

LINE	ADDR	B1	B2	B3	B4	LABEL	OPCODE	OPERAND	COMMENTS	PAGE
0001	0000								** PROGRAMM: HEX-DEZ-HEX WANDLUNG	0002
0002	0000								** HARDWARE: GRUNDVERSION + HEX-TASTATUR/ANZEIGE	
0003	0000								**	
0004	0000								**	
0005	0000								** FUNKTION: DAS PROGRAMM WANDELT EINE EINGEGEBENE 4-STELLIGE	
0006	0000								** BINAERZAHL (2 BYTE LANG) IN EINE 5-STELLIGE DEZIMALZAHL MIT	
0007	0000								** VORZEICHEN UND UMGEGEHRT EINE 5-STELLIGE DEZIMALZAHL MIT	
0008	0000								** VORZEICHEN IN EINE 4 STELLIGE BINAERZAHL (2 BYTE LANG).	
0009	0000								** NEGATIVE BINAERZAHLEN WERDEN DABEI ALS ZWEIERKOMPLEMENT	
0010	0000								** DARGESTELLT.	
0011	0000						REGS	EQU	09FF	
0012	0A00							ORG	0A00	
0013	0A00	BB	BD			WANDL	ZBSR	*KIN	HOLE TASTENWERT	
0014	0A02	BB	95				ZBSR	*CDIS	LOESCHE ANZEIGE	
0015	0A04	75	08				CPSL	WC	UEBERTRAG ABSCHALTEN	
0016	0A06	0F	08	00			LODA,R3	DATA1	LADE TASTENWERT NACH R3	
0017	0A09	E7	0D				COMI,R3	OD	TASTE 'D' GEDRUECKT?	
0018	0A0B	1C	0A	30			BCTA,EQ	DH	WENN JA,VERZWEIGE ZUR DEZ-HEX WANDL.	
0019	0A0E	E7	0B				COMI,R3	OB	TASTE 'B' GEDRUECKT?	
0020	0A10	18	02				BCTR,EQ	HD	WENN JA,VERZWEIGE ZUR HEX-DEZ WANDL.	
0021	0A12	1B	6C				BCTR,UN	WANDL	WENN SONST.TASTE, WEITER ABFRAGEN	
0022	0A14	06	76			HD	LODI,R2	76	'H' LADEN	
0023	0A16	CE	08	0D			STRA,R2	DISBUF	IN DEN ANZEIGESPEICHER	
0024	0A19	06	04				LODI,R2	4	4 STELLIGE EINGABE VORBEREITEN	
0025	0A1B	CE	08	2D			STRA,R2	STAT	4 STELLEN FUER LODAT FESTLEGEN	
0026	0A1E	BB	99				ZBSR	*LODAT	4 STELLEN EINGEBEN	
0027	0A20	0E	08	1B			LODA,R2	WBUF+6	4-STELLIGE BINAERZAHL NACH	
0028	0A23	CE	08	2B			STRA,R2	RSLT	RSLT UND	
0029	0A26	0E	08	1C			LODA,R2	WBUF+7	RSLT+1 UEBER-	
0030	0A29	CE	08	2C			STRA,R2	RSLT+1	TRAGEN	
0031	0A2C	BB	A5				ZBSR	*HEXD	IN DEZIMALZAHL WANDELN UND ANZEIGEN	
0032	0A2E	1B	50				BCTR,UN	WANDL	ZURUECK ZUR NAECHSTEN EINGABE	
0033	0A30	BB	BD			DH	ZBSR	*KIN	TASTATUR ABFRAGEN	
0034	0A32	0F	08	00			LODA,R3	DATA1	TASTATURWERT NACH R3	
0035	0A35	E7	40				COMI,R3	40	'+' EINGEGEBEN?	
0036	0A37	18	06				BCTR,EQ	DHEX1	WENN JA, IN POSITIV-ROUTINE SPRING.	
0037	0A39	E7	20				COMI,R3	20	'-' EINGEGEBEN?	
0038	0A3B	18	0C				BCTR,EQ	DHEX2	WENN JA, IN NEGATIV-ROUT-SPRING.	
0039	0A3D	1B	71				BCTR,UN	DH	SONST WEITER TASTATUR ABFRAGEN	
0040	0A3F	CF	09	FF		DHEX1	STRA,R3	REGS	VORZEICHEN SICHERN	
0041	0A42	07	73				LODI,R3	73	'P' = POSITIV SPEICHERN	
0042	0A44	CF	08	0D			STRA,R3	DISBUF	NACH ANZEIGESPEICHER	
0043	0A47	1B	08				BCTR,UN	DHEX3	WEITER ZUR EINGABEROUTINE	
0044	0A49	CF	09	FF		DHEX2	STRA,R3	REGS	VORZEICHEN SICHERN	
0045	0A4C	07	40				LODI,R3	40	'-' NACH R3	
0046	0A4E	CF	08	0D			STRA,R3	DISBUF	ZUM ANZEIGESPEICHER	
0047	0A51	06	05			DHEX3	LODI,R2	05	5-STELLIGE ANZEIGE VORBEREITEN	
0048	0A53	CE	08	2D			STRA,R2	STAT	5 STELLEN FUER LODAT VORGEHEN	
0049	0A56	BB	99				ZBSR	*LODAT	5-STELLIGE EINGABE	
0050	0A58	07	20				LODI,R3	20	'+' SPEICHERN	

0051	0A5A	EF	09	FF		COMA,R3	REGS	IST DEZIMALZAHL POSITIV?	
0052	0A5D	98	08			BCFR,EQ	DHEX5	WENN JA, IN BINAERZAHL WANDELN	
0053	0A5F	0F	08	1A		LODA,R3	WBUF+5	HOECHSTES BYTE HOLEN	
0054	0A62	67	90			IORI,R3	90	NEGATIVES VORZEICHEN SETZEN	
0055	0A64	CF	08	1A		STRA,R3	WBUF+5	DEZIMALZAHL WIEDER SPEICHERN	
0056	0A67	06	03			DHEX5	LODI,R2	3 LADEN	
0057	0A69	0E	48	1A		DHEX6	LODA,R2	WBUF+5,-	DEZIMALZAHL VON WBUF NACH BCDD
0058	0A6C	CE	68	2E		STRA,R2	BCDD,I	UEBERTRAGEN	
0059	0A6F	5A	78			BRNR,R2	DHEX6	WEITERE BYTES UEBERTRAGEN	
0060	0A71	BB	A7			ZBSR	*DHEX	IN HEX-ZAHL WANDELN	
0061	0A73	1F	0A	00		BCTA,UN	WANDL	ZURUECK ZUM ANFANG	
0062	0A76							** ACHTUNG!! DIESES PROGRAMM WURDE ABSICHTLICH AB ADRESSE	
0063	0A76							** 0A00 GESCHRIEBEN. ES KANN ZUSAMMEN MIT DEM PROGRAMM	
0064	0A76							** HEX-RECHNEN GELADEN WERDEN. VOR PROGRAMM-START MUSS DER	
0065	0A76							** PROGRAMM-COUNTER PC AUF 0A00 GESETZT WERDEN. DAS PROGRAMM	
0066	0A76							** WIRD DANN MIT DER 'GOTO' TASTE GESTARTET. DAS GLEICHZEITIG	
0067	0A76							** GELADENE PROGRAMM HEX-RECHNEN WIRD WIE GEWOHNT MIT 'RUN'	
0068	0A76							** GESTARTET.	
0069	0A76							** WIRD DAS PROGRAMM OHNE HEX-RECHNEN BENUTZT, SO EMPFIEHLT	
0070	0A76							** ES SICH, BEI DER ADRESSE 0900 DEN BEFEHL 1F 0A 00	
0071	0A76							** (BCTA,UN 0A00 = UNBEDINGTER SPRUNG NACH 0A00) EINZUGEBEN	
0072	0A76							** DAS PROGRAMM KANN DANN, WIE GEWOHNT MIT DER RUN TASTE	
0073	0A76							** GESTARTET WERDEN.	

Nach dem Laden stellen Sie den Programmzähler auf die Adresse 0A00. Danach starten Sie das Programm mit GOTO:

1. Dez-Hex-Wandlung der Zahl 123₁₀:

Taste	Anzeige	Bemerkungen
GOTO	HALLO.	Der Prozessor wartet auf eine Eingabe; er springt nicht in die Routine CDIS, daher keine Änderung der Anzeige;
D	leer	Entscheidung für Dez-Hex-Wandlung;
+(RUN)	P	es soll eine positive Zahl eingegeben werden;
oder		
-(GOTO)	-	es soll eine negative Zahl eingegeben werden.
0	P 0.	Die (ganzen) Zahlen müssen 5stellig, also u. U. mit vorangehenden Nullen eingegeben werden. Die Punkte erscheinen als „Merker“ wie bei der Daten- oder Adreßeingabe.
0	P 0.0	
1	P 0.0.1	
2	P 0.0.1.2	
3	leer, dann	Sofort nach dem Loslassen der letzten Taste erscheint das Ergebnis
	H 007B	bei Eingabe von +123, oder
	H FF85	bei Eingabe von -123.
		Das H steht als Index für HEX-Zahl.

Zur Umrechnung der nächsten Dez-Zahl „D - RUN/GOTO - 5stellige Zahl“ eingeben.

2. Hex-Dez-Wandlung der Zahl 007B₁₆:

GOTO HALLO. s.o.
 B H Entscheidung für die Hex-Dez-Wandlung;
 0 H.0. keine Vorzeicheneingabe; Zahlen von 0000 bis 7FFF
 0 H.0.0 werden als positive Zahlen, und Zahlen von 8000 bis
 7 H.0.0.7 FFFF werden als negative Hex-Zahlen im Zweierkom-
 B leer, dann plement interpretiert. Sofort nach dem Loslassen der
 P 00123 letzten Taste erscheint das Ergebnis.

HEX-Rechnen

FILE 'PROG16' AS ASSEMBLED BY SYSTEM ON 08-03-84

PAGE 0001

LINE	ADDR	B1	B2	B3	B4	LABEL	OPCODE	OPERAND	COMMENTS
0001	0000								** PROGRAMM: HEX-RECHNEN
0002	0000								** HARDWARE: GRUNDVERSION + HEX-TASTATUR/ANZEIGE
0003	0000								**
0004	0000								** - 1MHZ
0005	0000								**
0005	0000								** FUNKTION: DAS PROGRAMM FUEHRT ADDITION,SUBTRAKTION,
0006	0000								** MULTIPLIKATION UND DIVISION MIT HEX-ZAHLEN AUS UND
0007	0000								** ZEIGT DAS ERGEBNIS AUF DEM DISPLAY AN.
0008	0000					REGS	EQU	08FF	
0009	0900						ORG	0900	
0010	0900	3F	09	54		START	BSTA,UN	GHEX	ERSTE ZAHL HOLEN
0011	0903	CF	08	27			STRA,R3	OPR1	HOHES BYTE SPEICHERN
0012	0906	CE	08	28			STRA,R2	OPR1+1	NIEDR.BYTE SPEICHERN
0013	0909	BB	8D			SIGN	ZBSR	*KIN	KEYBOARD LESEN
0014	090B	0F	08	00			L0DA,R3	DATA1	TASTENWERT HOLEN
0015	090E	CF	08	FF			STRA,R3	REGS	TASTENWERT SICHERN
0016	0911	E7	0F				COMI,R3	0F	TASTENWERT 0F?
0017	0913	9D	09	09			BCFA,GT	SIGN	WENN NICHT GROESSER,WEITER SUCHEN
0018	0916	E7	50				COMI,R3	50	TASTENWERT 50?
0019	0918	9A	6F				BCFR,LT	SIGN	WENN NICHT KLEINER,WEITER SUCHEN
0020	091A	3F	09	54			BSTA,UN	GHEX	ZWEITE ZAHL HOLEN
0021	091D	CF	08	29			STRA,R3	OPR2	HOHES BYTE SPEICHERN
0022	0920	CE	08	2A			STRA,R2	OPR2+1	NIEDR.BYTE SPEICHERN
0023	0923	0F	08	FF			L0DA,R3	REGS	OPERATOR HOLEN
0024	0926	E7	20				COMI,R3	20	OPERATOR -?
0025	0928	18	13				BCTR,EQ	MAT1	ZUR SUBTRAKTION
0026	092A	E7	40				COMI,R3	40	OPERATOR +?
0027	092C	18	13				BCTR,EQ	MAT2	ZUR ADDITION
0028	092E	E7	30				COMI,R3	30	OPERATOR *?
0029	0930	18	1A				BCTR,EQ	MAT3	ZUR MULTIPLIKATION
0030	0932	E7	10				COMI,R3	10	OPERATOR /?
0031	0934	18	1A				BCTR,EQ	MAT4	ZUR DIVISION
0032	0936	07	4F				LODI,R3	4F	'3' LADEN
0033	0938	CF	08	12			STRA,R3	DISBUF+5	ZUM ANZEIGESPEICHER
0034	093B	9B	AD				ZBRR	*ERROR	ZUR ERROR ROUTINE

0035	093D	77	02			MAT1	PSSL	COM	SUBTRAKTION ANFORDERN
0036	093F	1B	02				BCTR,UN	MAT5	RECHNEN
0037	0941	75	02			MAT2	CPSL	COM	ADDITION ANFORDERN
0038	0943	BB	9B			MAT5	ZBSR	*ADDSUB	RECHNEN
0039	0945	BB	A9			MAT6	ZBSR	*HEX	ERGEBNIS ANZEIGEN
0040	0947	BB	8D				ZBSR	*KIN	AUF TASTENDRUCK WARTEN
0041	0949	1F	09	00			BCTA,UN	START	VON VORN
0042	094C	BB	9D			MAT3	ZBSR	*MULT	MULTIPLIZIEREN
0043	094E	1B	75				BCTR,UN	MAT6	ZUR ANZEIGE
0044	0950	BB	9F			MAT4	ZBSR	*DIV	DIVIDIEREN
0045	0952	1B	71				BCTR,UN	MAT6	ZUR ANZEIGE
0046	0954	BB	95			GHEX	ZBSR	*CDIS	ANZEIGE LOESCHEN
0047	0956	75	08				CPSL	WC	UEBERTRAG ABSCHALTEN
0048	0958	07	76				LODI,R3	76	H=HEX LADEN
0049	095A	CF	08	0D			STRA,R3	DISBUF	ZUR ANZEIGE
0050	095D	07	04				LODI,R3	4	4 LADEN
0051	095F	CF	08	2D			STRA,R3	STAT	INDEX FUER ZIFFERNZAHL
0052	0962	BB	99				ZBSR	*LODAT	ZAHL VON TASTATUR HOLEN
0053	0964	0F	08	1B			L0DA,R3	WBUF+6	HOHES BYTE LADEN
0054	0967	0E	08	1C			L0DA,R2	WBUF+7	NIEDR.BYTE LADEN
0055	096A	17					RETC,UN		ZURUECK INS HAUPTPROGRAMM
0056	096B								** NACH DEM START MIT 'RUN' ERSCHEINT H AUF DER ANZEIGE.
0057	096B								** ES WIRD DANN DIE ERSTE 4STELLIGE HEX ZAHL EINGEGEBEN.
0058	096B								** DANN EINGABE DES OPERATOREN:
0059	096B								** + = TASTE RUN
0060	096B								** - = TASTE GOTO
0061	096B								** * = TASTE PC
0062	096B								** / = TASTE NEXT
0063	096B								** DANN EINGABE DER ZWEITEN 4STELLIGEN HEX ZAHL. UNMITTEL-
0064	096B								** BAR DANACH ERSCHEINT DAS ERGEBNIS AUF DER ANZEIGE.
0065	096B								** NACH EINEM WEITEREN BELIEBIGEN TASTENDRUCK BEGINNT DAS
0066	096B								** PROGRAMM NEU.
0067	096B								** HOECHSTE VERARBEITBARE ZAHLEN:
0068	096B								** ADDITION/SUBTRAKTION: H7FFF(POSITIV) HFFFF(NEGATIV)
0069	096B								** MULTIPLIKATION : H007F(POSITIV) H007F(NEGATIV)
0070	096B								** DIE ERSTEN BEIDEN STELLEN WERDEN IGNORIERT
0071	096B								** DIVISION : DIVIDEND H7FFF(POSITIV)
0072	096B								** HFFFF(NEGATIV)
0073	096B								** DIVISOR H007F (NEG.DIVISOR WIRD POS.)

DEZ-Rechnen

FILE 'PROG17' AS ASSEMBLED BY SYSTEM ON 07-03-84

PAGE 0001

LINE	ADDR	B1	B2	B3	B4	LABEL	OPCODE	OPERAND	COMMENTS
0001	0000								** PROGRAMM: DEZ-RECHNEN
0002	0000								** HARDWARE: GRUNDVERSION + HEX-TASTATUR/ANZEIGE
0003	0000								**
0004	0000								** - 1MHZ
0005	0000								**
0005	0000								** FUNKTION: DAS PROGRAMM FUEHRT ADDITION,SUBTRAKTION,
0006	0000								** MULTIPLIKATION UND DIVISION MIT DEZIMALZAHLEN AUS UND

```

0007 0000      ** ZEIGT DAS ERGEBNIS AUF DEM DISPLAY AN.
0008 0000      REGS  EQU    08FE
0009 0900      ORG    0900
0010 0900 3F 09 54  START  BSTA,UN  GDEZ  ERSTE ZAHL HOLEN
0011 0903 CF 08 27      STRA,R3  OPR1  HOHES BYTE SPEICHERN
0012 0906 CE 08 28      STRA,R2  OPR1+1  NIEDR.BYTE SPEICHERN
0013 0909 BB 8D      SIGN  ZBSR    *KIN  KEYBOARD LESEN
0014 090B OF 08 00      LODA,R3  DATA1  TASTENWERT HOLEN
0015 090E CF 08 FE      STRA,R3  REGS   TASTENWERT SICHERN
0016 0911 E7 0F      COMI,R3  OF      TASTENWERT OF?
0017 0913 9D 09 09      BCFA,GT  SIGN   WENN NICHT GROESSER,WEITER SUCHEN
0018 0916 E7 50      COMI,R3  50     TASTENWERT 50?
0019 0918 9A 6F      BCFR,LT  SIGN   WENN NICHT KLEINER,WEITER SUCHEN
0020 091A 3F 09 54      BSTA,UN  GDEZ  ZWEITE ZAHL HOLEN
0021 091D CF 08 29      STRA,R3  OPR2  HOHES BYTE SPEICHERN
0022 0920 CE 08 2A      STRA,R2  OPR2+1  NIEDR.BYTE SPEICHERN
0023 0923 OF 08 FE      LODA,R3  REGS   OPERATOR HOLEN
0024 0926 E7 20      COMI,R3  20     OPERATOR -?
0025 0928 18 13      BCTR,EQ  MAT1   ZUR SUBTRAKTION
0026 092A E7 40      COMI,R3  40     OPERATOR +?
0027 092C 18 13      BCTR,EQ  MAT2   ZUR ADDITION
0028 092E E7 30      COMI,R3  30     OPERATOR *?
0029 0930 18 1A      BCTR,EQ  MAT3   ZUR MULTIPLIKATION
0030 0932 E7 10      COMI,R3  10     OPERATOR /?
0031 0934 18 1A      BCTR,EQ  MAT4   ZUR DIVISION
0032 0936 07 4F      LODI,R3  4F     '3' LADEN
0033 0938 CF 08 12      STRA,R3  DISBUF+5  ZUM ANZEIGESPEICHER
0034 093B 9B AD      ZBSR    *ERROR  ZUR ERROR ROUTINE
0035 093D 77 02      MAT1  PPSL    COM   SUBTRAKTION ANFORDERN
0036 093F 1B 02      BCTR,UN  MAT5   RECHNEN
0037 0941 75 02      MAT2  CPSL    COM   ADDITION ANFORDERN
0038 0943 BB 9B      MAT5  ZBSR    *ADDSUB  RECHNEN
0039 0945 BB A5      MAT6  ZBSR    *HEXD   ERGEBNIS IN DEZ WANDELN UND ANZEIGEN
0040 0947 BB 8D      ZBSR    *KIN  AUF TASTENDRUCK WARTEN
0041 0949 1F 09 00      BCTA,UN  START  VON VORN
0042 094C BB 9D      MAT3  ZBSR    *MULT  MULTIPLIZIEREN
0043 094E 1B 75      BCTR,UN  MAT6   ZUR ANZEIGE
0044 0950 BB 9F      MAT4  ZBSR    *DIV   DIVIDIEREN
0045 0952 1B 71      BCTR,UN  MAT6   ZUR ANZEIGE
0046 0954 BB 95      GDEZ  ZBSR    *CDIS  ANZEIGE LOESCHEN
0047 0956 BB 8D      ZBSR    *KIN  TASTATURWERT HOLEN
0048 0958 75 08      CPSL    WC    UEBERTRAG ABSCHALTEN
0049 095A OF 08 00      LODA,R3  DATA1  TASTATURWERT HOLEN
0050 095D E7 40      COMI,R3  40     PLUS?
0051 095F 1B 06      BCTR,EQ  DHEX1  WENN JA, ZUR POS.ROUTINE
0052 0961 E7 20      COMI,R3  20     MINUS?
0053 0963 1B 0C      BCTR,EQ  DHEX2  ZUR NEGATIV-ROUTINE
0054 0965 1B 6D      BCTR,UN  GDEZ   SONST WEITER SUCHEN
0055 0967 CF 08 FF      DHEX1  STRA,R3  REGS+1  VORZEICHEN SICHERN
0056 096A 07 73      LODI,R3  73     'P' LADEN
0057 096C CF 08 0D      STRA,R3  DISBUF  ZUR ANZEIGE
0058 096F 1B 08      BCTR,UN  DHEX3  ZUR HEX-WANDLUNG
0059 0971 CF 08 FF      DHEX2  STRA,R3  REGS+1  VORZEICHEN SICHERN
0060 0974 07 40      LODI,R3  40     '-' LADEN
0061 0976 CF 08 0D      STRA,R3  DISBUF  ZUR ANZEIGE

```

```

0062 0979 06 05      DHEX3  LODI,R2  5     5 LADEN
0063 097B CE 08 2D      STRA,R2  STAT   INS INDEXREGISTER
0064 097E BB 99      ZBSR    *LODAT  DEZIMALZAHL EINGEBEN
0065 0980 07 20      LODI,R3  20     20 LADEN
0066 0982 EF 08 FF      COMA,R3  REGS+1  VORZEICHEN -?
0067 0985 9B 08      BCFR,EQ  DHEX5  WENN NICHT,WEITER
0068 0987 OF 08 1A      LODA,R3  WBUF+5  HOECHSTES BYTE LADEN
0069 098A 67 90      IORI,R3  90     NEG.VORZEICHEN SETZEN
0070 098C CF 08 1A      STRA,R3  WBUF+5  HOECHSTES BYTE SPEICHERN
0071 098F 06 03      DHEX5  LODI,R2  3     INDEXREGISTER LADEN
0072 0991 0E 4B 1A      DHEX6  LODA,R2  WBUF+5,-  ARBEITSSPEICHER ZUM
0073 0994 CE 68 2E      STRA,R2  BCDD,I  DEZIMALSPEICHER UMLADEN
0074 0997 5A 7B      BRNR,R2  DHEX6  ZAHL ZUENDE, SONST WEITER
0075 0999 BB A1      ZBSR    *DEZHEX  DEZ.ZAHL IN HEX WANDELN
0076 099B OF 08 2B      LODA,R3  RSLT   HOHES HEX BYTE LADEN
0077 099E 0E 08 2C      LODA,R2  RSLT+1  NIEDR.HEX BYTE LADEN
0078 09A1 17      RETC,UN  ZURUECK INS HAUPTPROGRAMM
0079 09A2      ** NACH DEM START MIT 'RUN' EINGABE VON + (TASTE RUN) ODER
0080 09A2      ** - (TASTE GOTO), DANN 5 STELLIGE DEZIMALZAHL.
0081 09A2      ** DANN EINGABE DES OPERATOREN:
0082 09A2      ** + = TASTE RUN
0083 09A2      ** - = TASTE GOTO
0084 09A2      ** * = TASTE PC
0085 09A2      ** / = TASTE NEXT
0086 09A2      ** DANN EINGABE DER ZWEITEN DEZIMALZAHL WIE OBEN. UNMITTEL-
0087 09A2      ** BAR DANACH ERSCHEINT DAS ERGEBNIS AUF DER ANZEIGE. NACH
0088 09A2      ** EINEM BELIEBIGEN TASTENDRUCK BEGINNT DAS PROGRAMM NEU.
0089 09A2      ** HOECHSTE VERARBEITBARE ZAHLEN:
0090 09A2      ** ADDITION/SUBTRAKTION: +32767 -32768
0091 09A2      ** MULTIPLIKATION      : +-127
0092 09A2      ** DIVISION            : DIVIDEND +32767 -32768
0093 09A2      **                      : DIVISOR 127

```

Lottozahlen

FILE 'PROG21' AS ASSEMBLED BY SYSTEM ON 08-03-84

PAGE 0002

```

LINE ADDR B1 B2 B3 B4 LABEL OPCODE OPERAND COMMENTS
0001 0000      ** PROGRAMM: LOTTOZAHLEN
0002 0000      ** HARDWARE: GRUNDVERSION + HEX-TASTATUR/ANZEIGE
0003 0000      ** - 1MHZ
0004 0000      ** DER SENSE-EINGANG (PIN 6 DER RECORDER-BUCHSE) IST MIT DEM
0005 0000      ** 50 HZ AUSGANG DER NETZTEIL-PLATINE ZU VERBINDEN
0006 0000      **
0007 0000      ** FUNKTION: DAS PROGRAMM ERMITTELT NACHEINANDER 6 ZUFALLS-
0008 0000      ** ZAHLEN ZWISCHEN 1 UND 49 UND VERMEIDET DABEI DOBBELTE ZAHLEN
0009 0000      LOTTO  EQU    08F0
0010 0900      ORG    0900
0011 0900 BB 95      START  ZBSR    *CDIS  ANZEIGE LOESCHEN
0012 0902 06 0F      LODI,R2  OF     INDEX LADEN
0013 0904 20      EORZ,R0

```

```

0014 0905 CE 48 F0 CLR STRA,R2 LOTTO,- ZAHLENSPEICHER LOESCHEN
0015 0908 5A 7B BRNR,R2 CLR LOESCHEN BIS ALLE LEER
0016 090A CC 08 2B STRA,R0 RSLT RSLT LOESCHEN
0017 090D 06 00 LODI,R2 0 R2 AUF 0
0018 090F CE 08 F7 ZIEH STRA,R2 LOTTO+7 INDEX SICHERN
0019 0912 3F 09 50 BSTA,UN RANDOM ZUFALLSZAH ZIEHEN
0020 0915 0E 08 F7 LODA,R2 LOTTO+7 INDEX HOLEN
0021 0918 CE 28 F0 STRA,R2 LOTTO,+ ZUFALLSZAH SPEICHERN
0022 091B CE 08 F7 STRA,R2 LOTTO+7 INDEX SICHERN
0023 091E CC 08 2C STRA,R0 RSLT+1 ZUFALLSZAH ZUR DEZ.WANDL EINGEBEN
0024 0921 BB 83 ZBSR *HEXDEZ IN DEZ.ZAHL WANDELN
0025 0923 0F 08 30 LODA,R3 BCDD+2 DEZIMALZAH HOLEN
0026 0926 CF 08 01 STRA,R3 DATA2 ZUR SEPARIERUNG VORBEREITEN
0027 0929 BB 87 ZBSR *SEP IN 2 ZAHLEN AUFTHEILEN
0028 092B 0F 08 02 LODA,R3 DATA3 LETZTE ZAH HOLEN
0029 092E BB 85 ZBSR *CONV IN 7-SEGM.WANDELN
0030 0930 CF 08 12 STRA,R3 DISBUF+5 ZUR ANZEIGE
0031 0933 0F 08 01 LODA,R3 DATA2 ERSTE ZAH HOLEN
0032 0936 BB 85 ZBSR *CONV IN 7-SEGM.WANDELN
0033 0938 CF 08 11 STRA,R3 DISBUF+4 ZUR ANZEIGE
0034 093B 0F 08 F7 LODA,R3 LOTTO+7
0035 093E BB 85 ZBSR *CONV IN 7-SEGM.WANDELN
0036 0940 CF 08 0D STRA,R3 DISBUF ZUR ANZEIGE
0037 0943 BB 8D ZBSR *KIN AUF TASTENDRUCK WARTEN
0038 0945 0E 08 F7 LODA,R2 LOTTO+7 INDEX HOLEN
0039 0948 E6 07 COMI,R2 7 6 ZAHLEN GEZOGEN?
0040 094A 1C 09 00 BCTA,EQ START WENN JA, ZUM ANFANG
0041 094D 1F 09 0F BCTA,UN ZIEH WENN NEIN, NAECHSTE ZAH ZIEHEN
0042 0950 12 RANDOM SPSU INHALT VON PSU NACH RO LADEN
0043 0951 44 80 ANDI,R0 80 NUR HOECHSTES BIT BETRACHTEN
0044 0953 CC 08 15 STRA,R0 WBUF SENSE ZUSTAND SPEICHERN
0045 0956 07 00 NEW LODI,R3 0 ZAEHLER AUF 0 SETZEN
0046 0958 87 01 LOOP ADDI,R3 1 UM 1 ERHOEHEN
0047 095A E7 32 COMI,R3 32 WERT GROESSER ALS 49?
0048 095C 18 7B BCTR,EQ NEW WENN JA, VON VORN BEGINNEN
0049 095E 12 SPSU INHALT VON PSU NACH RO LADEN
0050 095F 44 80 ANDI,R0 80 NUR HOECHSTES BIT BETRACHTEN
0051 0961 EC 08 15 COMA,R0 WBUF SENSE ZUSTAND VERAENDERT?
0052 0964 18 72 BCTR,EQ LOOP WENN NEIN, WEITER ZAEHLEN
0053 0966 03 LODZ,R3 ZUFALLSZAH NACH RO
0054 0967 06 00 VERGL LODI,R2 0 INDEX AUF 0
0055 0969 EE 28 F0 VERGL1 COMA,R2 LOTTO,+ WAR ZAH SCHON DA?
0056 096C 18 62 BCTR,EQ RANDOM WENN JA, WEITER ZIEHEN
0057 096E E6 07 COMI,R2 7 ALLE ZAHLEN ABGESUCHT?
0058 0970 98 77 BCFR,EQ VERGL1 WENN NICHT, WEITER VERGLEICHEN
0059 0972 17 RETC,UN SONST ZURUECK INS HAUPTPROGRAMM
0060 0973 ** DAS PROGRAMM LAESST SICH ANDEREN ZAHLENSYSTEMEN ANPASSEN,
0061 0973 ** WENN DIE ENZZIFFER IM COMPARE-BEFEHL DER RANDOM ROUTINE
0062 0973 ** GEAENDERT WIRD.

```

Morse-Übungsprogramm

```

LINE ADDR B1 B2 B3 B4 LABEL OPCODE OPERAND COMMENTS PAGE 0002
0001 0000 ** PROGRAMM: MORSEN
0002 0000 ** HARDWARE: GRUNDVERSION + HEX-TASTATUR/ANZEIGE
0003 0000 ** - 1MHZ
0004 0000 ** SENSE ANSCHLUSS (PIN 6 DER 7-POL-BUCHSE AUF CPU)
0005 0000 ** MIT 50 HZ AUSGANG DER NETZTEILPLATINE VERBINDEN
0006 0000 ** 150 OHM LAUTSPRECHER AN EINEN PORT ANSCHLUSS
0007 0000 **
0008 0000 ** FUNKTION: DAS PROGRAMM ERZEUGT MORSEZEICHEN IN FUENFER-
0009 0000 ** GRUPPEN UND GIBT SIE ALS TOENE UEBER EINEN AN DEN PORT
0010 0000 ** 09 ANGESCHLOSSENEN LAUTSPRECHER WIEDER. DAS PROGRAMM
0011 0000 ** BEGINNT MIT DER ZEICHENGRUPPE 'KA' ALS ANFANGSZEICHEN.
0012 0000 ** IM DATA-TEIL SIND DIE MORSEZEICHEN ABGELEGT. DAS ERSTE
0013 0000 ** BYTE GIBT DIE LAENGE (ZAHL DER ELEMENTE) DES MORSEZEICHENS
0014 0000 ** AN, DAS ZWEITE DIE ELEMENTEFOLGE. DABEI ENTSPRICHT EIN
0015 0000 ** STRICH EINER EINS, EIN PUNKT EINER NULL.
0016 0900 ORG 0900
0017 0900 PAU EQU 08FE
0018 0900 ZAE EQU 08FF
0019 0900 05 05 RUN LODI,R1 05 KN ALS ANFANGS-
0020 0902 07 A8 LODI,R3 A8 ZEICHEN LADEN
0021 0904 04 00 LODI,R0 00 FUENFERGRUPPEN ZAEHLER SETZEN
0022 0906 CC 08 FF STRA,R0 ZAE ABSPEICHERN
0023 0909 3F 09 36 BSTA,UN ROTA ZEICHEN SENDEN
0024 090C 3F 09 4C RUN1 BSTA,UN GRPAU GROSSE PAUSE
0025 090F 3F 09 94 BSTA,UN RANDOM ZUFALLSZEICHEN ERZEUGEN
0026 0912 0E 6A 00 LODA,R2 SIGN,I 1.BYTE DES ZEICHENS HOLEN
0027 0915 C1 STRZ,R1 IN R1 SPEICHERN
0028 0916 0E 6A 01 LODA,R2 SIGN+1,I 2.BYTE DES ZEICHENS HOLEN
0029 0919 C3 STRZ,R3 IN R3 SPEICHERN
0030 091A 3B 1A BSTR,UN ROTA ZEICHEN SENDEN
0031 091C 0E 08 FF LODA,R2 ZAE FUENFER-ZAEHLER LADEN
0032 091F 86 01 ADDI,R2 01 UM 1 ERHOEHEN
0033 0921 CE 08 FF STRA,R2 ZAE FUENFER-ZAEHLER WIEDER SPEICHERN
0034 0924 E6 05 COMI,R2 05 FUENF ZEICHEN GESENDET?
0035 0926 9C 09 0C BCFA,EQ RUN1 WENN NEIN, WEITER
0036 0929 06 00 LODI,R2 00 SONST FUENFERZAEHLER WIEDER AUF ANFANG
0037 092B CE 08 FF STRA,R2 ZAE UND ABSPEICHERN
0038 092E 3F 09 4C BSTA,UN GRPAU GROSSE PAUSE
0039 0931 3F 09 4C BSTA,UN GRPAU GROSSE PAUSE
0040 0934 1B 56 RUN1 BCTR,UN RUN1 NAECHSTES ZEICHEN HOLEN
0041 0936 77 08 ROTA PPSL WC UEBERTRAG EINSCHALTEN
0042 0938 D3 RRL ,R3 BIT INS UEBERTRAGSBIT SCHIEBEN
0043 0939 B5 01 TPSP 01 1 ODER 0?
0044 093B 9C 09 43 BCFA,EQ PUN WENN 0,ZUR PUNKT ROUTINE
0045 093E 3F 09 72 BSTA,UN STRICH SONST ZUR STRICH-ROUTINE
0046 0941 1B 03 BCTR,UN PUN1 NAECHSTES BIT HOLEN
0047 0943 3F 09 77 PUN BSTA,UN PUNKT PUNKT SENDEN
0048 0946 FD 09 36 PUN1 BDRA,R1 ROTA NAECHSTES BIT HOLEN
0049 0949 75 08 CPSL WC UEBERTRAG ABSCHALTEN
0050 094B 17 ROTA1 RETC,UN ZURUECK INS HAUPTPROGRAMM

```


0051	094C	77	10	GRPAU	PPSL	RS	REGISTERBANK WECHSELN
0052	094E	05	00		LODI,R1	00	R1 AUF 0
0053	0950	D5	09		WRTE,R1	09	PORT AUSSCHALTEN
0054	0952	20			EDRZ,R0		R0 AUF 0
0055	0953	06	20	GRPAU2	LODI,R2	20	R2 AUF 20
0056	0955	0F	08	FE	GRPAU1	LODA,R3	PAU
0057	0958	FB	7E		BDRR,R3	\$	R3 AUF 0 ZAEHLEN
0058	095A	FA	79		BDRR,R2	GRPAU1	R2 AUF 0 ZAEHLEN
0059	095C	F8	75		BDRR,R0	GRPAU2	R0 AUF 0 ZAEHLEN
0060	095E	75	10		CPSL	RS	REGISTERBANK WECHSELN
0061	0960	17			RETC,UN		ZURUECK INS HAUPTPROGRAMM
0062	0961	77	10	KLPAU	PPSL	RS	REGISTERBANK WECHSELN
0063	0963	05	00		LODI,R1	00	R1 AUF 0
0064	0965	D5	09		WRTE,R1	09	PORT ABSCHALTEN
0065	0967	06	00		LODI,R2	00	R2 AUF 0
0066	0969	07	30	KLPAU1	LODI,R3	30	R3 AUF 30
0067	096B	FB	7E		BDRR,R3	\$	R3 AUF 0 ZAEHLEN
0068	096D	FA	7A		BDRR,R2	KLPAU1	R2 AUF 0 ZAEHLEN
0069	096F	75	10		CPSL	RS	REGISTERBANK WECHSELN
0070	0971	17			RETC,UN		ZURUECK INS HAUPTPROGRAMM
0071	0972	04	F0	STRICH	LODI,R0	F0	R0 AUF F0
0072	0974	1F	09	79	BCTA,UN	TON	ZUR TONERZEUGUNG
0073	0977	04	50	PUNKT	LODI,R0	50	R0 AUF 50
0074	0979	77	10	TON	PPSL	RS	REGISTERBANK WECHSELN
0075	097B	06	FF	TON2	LODI,R2	FF	R2 AUF FF
0076	097D	D6	09		WRTE,R2	09	PORT EINSCHALTEN
0077	097F	3B	0E		BSTR,UN	DELAY	VERZOEGERN
0078	0981	06	00	TON3	LODI,R2	00	R2 AUF 0
0079	0983	D6	09		WRTE,R2	09	PORT AUSSCHALTEN
0080	0985	3B	08		BSTR,UN	DELAY	VERZOEGERN
0081	0987	FB	72		BDRR,R0	TON2	R0 AUF 0 ZAEHLEN
0082	0989	3F	09	61	BSTA,UN	KLPAU	KLEINE PAUSE
0083	098C	75	10		CPSL	RS	REGISTERBANK WECHSELN
0084	098E	17			RETC,UN		ZURUECK INS HAUPTPROGRAMM
0085	098F	07	38	DELAY	LODI,R3	38	R3 AUF 38 (TONHOEHE SPEICHERN)
0086	0991	FB	7E		BDRR,R3	\$	R3 AUF 0 ZAEHLEN
0087	0993	17			RETC,UN		ZURUECK INS HAUPTPROGRAMM
0088	0994	75	08	RANDOM	CPSL	WC	UEBERTRAG ABSCHALTEN
0089	0996	12			SPSU		PSU LESEN
0090	0997	44	80		ANDI,R0	80	NUR SENSE BIT BETRACHTEN
0091	0999	CC	08	15	STRA,R0	WBUF	ZUSTAND SPEICHERN
0092	099C	06	00	NEW	LODI,R2	00	R2 AUF 0
0093	099E	86	02	LOOP	ADDI,R2	02	2 ADDIEREN
0094	09A0	E6	54		COMI,R2	54	HOECHSTER ZEICHENWERT?
0095	09A2	18	78		BCTR,EQ	NEW	WENN JA, VON VORN ZAEHLEN
0096	09A4	12			SPSU		PSU LESEN
0097	09A5	44	80		ANDI,R0	80	NUR SENSE BIT BETRACHTEN
0098	09A7	EC	08	15	COMA,R0	WBUF	ZUSTAND GEAENDERT
0099	09AA	18	72		BCTR,EQ	LOOP	WENN NICHT, WEITERZAEHLEN
0100	09AC	17			RETC,UN		ZURUECK INS HAUPTPROGRAMM
0101	0A00				ORG	0A00	
0102	0A00	02	40	04	80	SIGN	DATA 02,40,04,80
0103	0A04	04	A0	03	80		DATA 04,A0,03,80,01,00
0104	0A08	01	00				

0105	0A0A	04	20	03	C0		DATA	04,20,03,C0,04,00
0106	0A0E	04	00					
0107	0A10	02	00	04	70		DATA	02,00,04,70,03,A0
0108	0A14	03	A0					
0109	0A16	04	04	02	C0		DATA	04,04,02,C0,02,80
0110	0A1A	02	80					
0111	0A1C	03	E0	04	60		DATA	03,E0,04,60,04,D0
0112	0A20	04	D0					
0113	0A22	03	40	03	00		DATA	03,40,03,00,01,80
0114	0A26	01	80					
0115	0A28	03	20	04	10		DATA	03,20,04,10,03,60
0116	0A2C	03	60					
0117	0A2E	04	90	04	B0		DATA	04,90,04,B0,04,C0
0118	0A32	04	C0					
0119	0A34	05	F8	05	78		DATA	05,F8,05,78,05,38
0120	0A38	05	38					
0121	0A3A	05	18	05	08		DATA	05,18,05,08,05,00
0122	0A3E	05	00					
0123	0A40	05	80	05	C0		DATA	05,80,05,C0,05,E0
0124	0A44	05	E0					
0125	0A46	05	F0				DATA	05,F0
0126	0A48	05	88	06	A8		DATA	05,88,06,A8,06,CC
0127	0A4C	06	CC					
0128	0A4E	06	84	05	90		DATA	06,84,05,90,06,E0
0129	0A52	06	E0					
0130	0A54	06	30	06	54		DATA	06,30,06,54
0131	0A58	05	A8	05	50		DATA	05,A8,05,50
0132	0A5C						** DURCH EINGABE EINES WERTES IN DIE SPEICHERSTELLE OBFE KANN	
0133	0A5C						** DIE MORSE-GESCHWINDIGKEIT (PAUSENLAENGE) UND DAMIT DER	
0134	0A5C						** SCHWIERIGKEITSGRAD EINGESTELLT WERDEN. SOLL AUCH DIE	
0135	0A5C						** ZEICHENGESCHWINDIGKEIT VERAENDERT WERDEN, SO SIND DIE	
0136	0A5C						** ZAEHL-REGISTER IN DEN ROUTINEN KLPAU, TON UND DELAY	
0137	0A5C						** ZU VERAENDERN. SOLL NICHT DER GESAMTE ZEICHENVORRAT	
0138	0A5C						** VERWENDET WERDEN, SO IST IN DER RANDOM ROUTINE DIE	
0139	0A5C						** ZAHL 54 DURCH EINE KLEINERE GERADE ZAHL ZU ERSETZEN.	

Voltmeter 2 (3stellig dezimal)

LINE	ADDR	B1	B2	B3	B4	LABEL	OPCODE	OPERAND	COMMENTS	PAGE
0001	0000								** PROGRAMM: VOLTMETER 2	0001
0002	0000								** HARDWARE: GRUNDVERSION + HEX-TASTATUR/ANZEIGE	
0003	0000								** + A/D-WANDLER	
0004	0000								** - 1 MHZ	
0005	0000								**	
0006	0000								** FUNKTION: PROGRAMM FRAGT DEN JEWEILIGEN WERT	
0007	0000								** DES A/D-WANDLERS AB, UEBERSETZT IHN	
0008	0000								** VON HEXADEZIMAL AUF DEZIMAL UND	
0009	0000								** ZEIGT IHN AUF DER 7-SEGMENT-ANZEIGE AN.	
0010	0000								** DAS PROGRAMM VERWandelt DEN COMPUTER OHNE	
0011	0000								** WEITEREN ZUSATZ IN EIN DIGITAL-VOLTMETER MIT	
0012	0000								** EINEM MESSBEREICH ZWISCHEN 0.255 UND 2.55 VOLT,	

0013	0000		** JE NACH SCHALTERSTELLUNG AUF DER A/D-WANDLER-PLAT.
0014	0900	ORG	0900
0015	0900	BB 95	START ZBSR *CDIS 7 SEGM.ANZ.LOESCHEN
0016	0902	57 0A	REDE,R3 A/D WERT VON A/D-WANDL.HOLEN
0017	0904	CF 08 2C	STRA,R3 RSLT+1 FUER HEX/DEZ WANDL.VORBER.
0018	0907	20	EDRZ,R0 R0 AUF 0
0019	0908	CC 08 2B	STRA,R0 RSLT RSLT AUF 0 SETZEN
0020	090B	BB A5	ZBSR *HEXD IN DEZIMALZAHL WANDELN U.ANZEIG.
0021	090D	0F 08 10	LODA,R3 DISBUF+3 ANZEIGE-STELLE LADEN
0022	0910	B7 80	ADDI,R3 80 DEZIMALPUNKT SETZEN
0023	0912	CF 08 10	STRA,R3 DISBUF+3 ANZEIGESTELLE SPEICHERN
0024	0915	06 20	LODI,R2 20 HELLIGKEIT DER ANZEIGE
0025	0917	BB 8B	SHOW ZBSR *DIS 7 SEGMENTANZEIGE AKTIVIEREN
0026	0919	FA 7C	BDRR,R2 SHOW MEHRFACH ANZEIGEN
0027	091B	1F 09 00	BCTA,UN START NAECHST.WERT VON A/D-W.HOLEN
0028	091E		** DER DEZIMALPUNKT KANN AN EINE ANDERE STELLE GESETZT WERDEN,
0029	091E		** WENN DIE SPEICHERPOSITION IN BEFEHL 090D UND 0912 VERAENDERT
0030	091E		** WIRD.

Anhang 1

Anhang 2

Detallierte Beschreibung der Befehle des Mikroprozessors 2650

Die folgende Beschreibung der Befehle wurde dem VALVO-Handbuch „Mikroprozessor 2650 A“, Hamburg 1980, entnommen.

VERWENDUNG DES CONDITION-CODES IM PROGRAMM-STATUS-WORT

Die beiden Condition-Code-Bits sind ein Teil des unteren Programm-Status-Wortes. Der Condition-Code wird bei Operationen, bei denen Daten in ein Register eingeschrieben werden, sowie bei Vergleichs- und Testoperationen festgelegt. Nach Ausführung eines solchen Befehls enthält der Condition-Code zusätzliche Informationen über jenes Datenbyte, mit dem zuletzt gearbeitet wurde, bzw. das Ergebnis bei Vergleichsoperationen und Testbefehlen. Wenn z.B. ein Register mit einem Datenbyte geladen wird, so enthält der Condition-Code die Information, ob das Byte positiv, negativ oder null ist. Das Vorzeichenbit eines Datenbytes ist das Bit 7, das "most significant" Bit (Zweier-Komplementdarstellung). Bei der Durchführung von bedingten Sprungbefehlen wird auf den Condition-Code zurückgegriffen.

Beispiel:

```
LODI,r H'F1' (Load Immediate, das Register r mit F116)  
BCTA,v Ende (Branch on Condition True, Absolute)
```

F1₁₆ ist eine negative Zahl; deshalb wird bei der Abarbeitung des Ladebefehls LODI,r das Bitmuster 1 0 in die beiden Condition-Code-Felder geschrieben. Stimmt das Feld v des Sprungbefehls BCTA,v mit dem Condition-Code überein, so wird ein Sprung zur symbolischen Adresse "Ende" ausgeführt. Herrscht keine Übereinstimmung, wird das Programm mit dem nächstfolgenden Befehl fortgesetzt (kein Sprung).

A D R E S S I E R U N G S A R T E N

Der Mikroprozessor 2650 besitzt folgende acht Adressierungsarten, die sich in der Art der Adreßberechnung und in der Ausführung unterscheiden:

- * Registeradressierung,
- * immediate Adressierung,
- * relative Adressierung,
- * relative, indirekte Adressierung,
- * absolute Adressierung,
- * absolute, indirekte Adressierung,
- * absolute Adressierung mit Indizierung,
- * absolute, indirekte Adressierung mit Indizierung.

Das Ergebnis jeder Adressierung wird **e f f e k t i v e** **A d r e s s e** genannt.

Registeradressierung

Alle Register/Register-Befehle sind ein Byte lang.

OP CODE							r
7	6	5	4	3	2	1	0

Bit 2 bis Bit 7 beinhalten den Operationscode. Das Registerfeld besteht aus den Bits 1 und 0. Da die sieben Rechenregister in zwei Bänke unterteilt sind (die Auswahl der Registerbank erfolgt durch das Register-Bank-Select-Bit im Programm-Status-Wort), kann mit dem Registerfeld jedes Rechenregister angesprochen werden. Der zweite Operand (wenn vorhanden) und das Ergebnis stehen immer im Register 0.

Beispiel 1: LODZ,r (Load Register 0)

Dieser Befehl lädt den Inhalt des im Registerfeld angegebenen Registers r in das Register 0, wobei der Inhalt von Register r unverändert bleibt.

Befehlscode:

0	0	0	0	0	0	r
---	---	---	---	---	---	---

Beispiel 2: RRL,r (Rotate Register r Left)

Bei dieser Operation wird der Inhalt des Registers r um eine Stelle nach links verschoben.

Befehlscode:

1	1	0	1	0	0	r
---	---	---	---	---	---	---

Immediate-Adressierung

Alle Befehle mit Immediate-Adressierung benötigen 2 Bytes. Das 1. Byte enthält den Operationscode und eine Registerbezeichnung, während das 2. Byte Daten enthält, welche im nächsten Befehl verwendet werden.

OP CODE							r	OPERAND (Daten)							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Byte 1								Byte 2							

Beispiel 1: ADDI,r v (Add Immediate)

Dieser Befehl bewirkt die Addition des im Datenfeld (2. Byte) stehenden Datenbytes v (eine Zahl im Zweier-Komplement) zum Inhalt des Registers r.

Befehlscode:

1	0	0	0	0	1	r										v
---	---	---	---	---	---	---	--	--	--	--	--	--	--	--	--	---

Beispiel 2: ANDI, r v (AND Immediate)

Hierbei werden der Inhalt des Registers r und der Inhalt des Datenfeldes v (Bitmuster) logisch UND-verknüpft. Diese UND-Verknüpfung wird bitweise durchgeführt. Das Resultat ersetzt das jeweilige Bit des Registers r.

Befehlscode:

0	1	0	0	0	1	r										v
---	---	---	---	---	---	---	--	--	--	--	--	--	--	--	--	---

Relative Adressierung

Befehle mit relativer Adressierung benötigen 2 Bytes. Ein Operand des auszuführenden Befehls ist ein Registerinhalt und der andere ist Inhalt eines Speicherplatzes. Das 1. Byte enthält den Befehlscode und eine Registerbezeichnung, während das 2. Byte die relative Adresse des zweiten Operanden enthält. Die effektive Adresse des zweiten Operanden wird durch Addition der relativen Adresse zum Inhalt des Befehlszählers (IAR) ermittelt (befehlszählerrelative Adressierung). Die relative Adresse wird als Zweier-Komplementzahl interpretiert. Bei der 7-bit-Adresse kann ein Bereich von -64 bis +63, bezogen auf den augenblicklichen Inhalt des Befehlszählers (IAR), adressiert werden. Enthält Bit 7 des 2. Bytes (I) eine 1, wird die relative, indirekte Adressierung durchgeführt. Siehe indirekte Adressierung.

OPCODE							r	I relat. Adresse							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Byte 1								Byte 2							

Beispiel: LODR,r a (Load relative)

Dieser Befehl transferiert Daten aus einer Speicherzelle in das bezeichnete Register r. Die Daten werden von der effektiven Speicheradresse ausgelesen, die durch Addition der relativen Adresse a und der Adresse des auf diesen Befehl folgenden Bytes ermittelt wird.

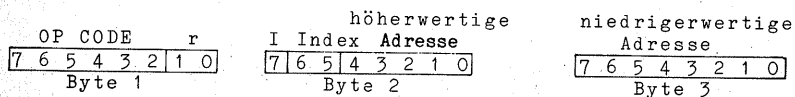
ADR		Inhalt	hex
500	LODR,1	0000 1001	09
501	04	0000 0100	04
		:	
		:	
		:	
506	XX	XXXX XXXX	XX

Wenn der Befehl "Load Relative" mit der relativen Adresse 04¹⁶ wie in diesem Beispiel auf den Adressen 500 und 501 steht, wird der Speicherinhalt XX der Adresse 506 in das Register r geschrieben.

Absolute Adressierung

Absolute Adressierung für Nicht-Sprung-Befehle:

Alle Befehle mit absoluter Adressierung sind drei Bytes lang. Ein Operand steht im Arbeitsregister, der andere ist der Inhalt jener Speicherstellen, die durch das 2. und 3. Byte angegeben werden.



Die Bits 0...4 vom 2. Byte und das gesamte 3. Byte geben die Adresse an. Es können somit alle Adressen innerhalb einer "Page" angesprochen werden.

Die beiden Index-Control-Bits (Bit 6 und Bit 5 des 2. Byte) bestimmen, wie die effektive Adresse berechnet wird und welches Register das Arbeitsregister darstellt.

Die Index-Control-Bits haben folgende Bedeutung:

Bit 6.	Bit 5	
0	0	keine Indexregisteradressierung
0	1	Indizierte Adressierung mit automatischer Inkrementierung des Indexregisters
1	0	Indizierte Adressierung mit automatischer Dekrementierung des Indexregisters
1	1	Indizierte Adressierung

a) Index-Control-Bits: 00

In diesem Fall beinhaltet das Registerfeld des 1. Bytes das Arbeitsregister. Die effektive Adresse ist in den Bits 0...4 des 2. Bytes und im 3. Byte enthalten.

b) Index-Control-Bits: 01

Die Berechnung der effektiven Adresse erfolgt wie in Punkt d), nur, daß vor Addition des Inhalts des Indexregisters zur angegebenen Adresse der Inhalt des Indexregisters um eins erhöht wird.

c) Index-Control-Bits: 10

Die Berechnung der effektiven Adresse erfolgt wie in d), nur, daß vor Addition des Inhalts des Indexregisters zur angegebenen Adresse, der Inhalt des Indexregisters um eins vermindert wird.

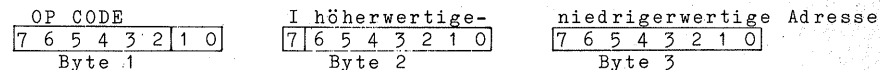
d) Index-Control-Bits: 11

Das Arbeitsregister ist in diesem Fall immer das Register 0. Im Registerfeld des 1. Bytes wird das Indexregister angegeben. Durch Addition des Inhalts des Indexregisters zur angegebenen Adresse ergibt sich dann die effektive Adresse.

Indirekte Adressierung ist bei dieser Adressierungsart möglich. Sie wird durch eine 1 im Bit 7 des 2. Bytes angezeigt.

Absolute Adressierung für Sprungbefehle:

Das Format des Befehls:



Bei diesen Befehlen werden die beiden Bits 6 und 5 des 2. Bytes als Adreßbits interpretiert. Die angegebene Adresse umfaßt somit 15 Bits und es kann somit der gesamte Speicherbereich (32 Kbyte) adressiert werden.

Indirekte Adressierung

Die indirekte Adressierung ist bei der relativen wie auch bei der absoluten Adressierung möglich. Sie wird durch eine 1 im Bit 7 des 2. Bytes angegeben. Hier wird die effektive Adresse nicht durch den Befehl selbst angegeben, sondern sie ist in jenen beiden Speicherzellen zu finden, die durch die Adresse im Befehl und die nächstfolgende Adresse angegeben werden (Adresse von der Adresse).

Bei der Durchführung werden deshalb zwei zusätzliche Prozessorzyklen (sechs Taktschritte) benötigt. Mittels indirekter Adressierung kann man den gesamten Speicherraum (32 Kbyte) adressieren, da die effektive Adresse zwei Bytes (16 Bits) umfaßt.

Beispiel:

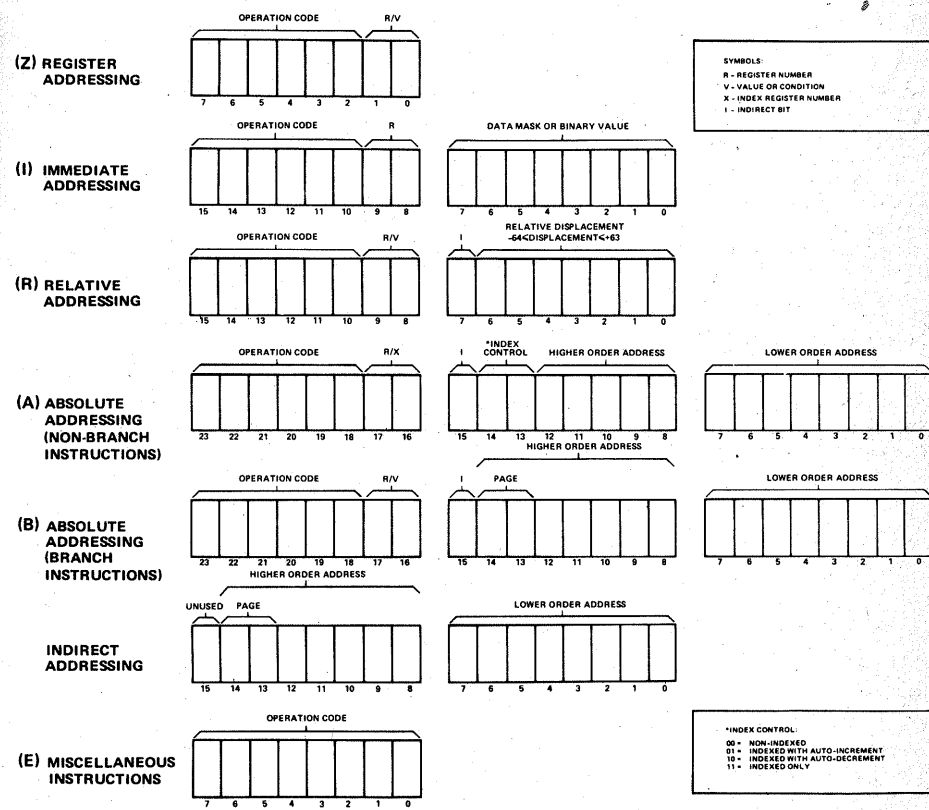
Adresse		Hex	Dual
0010		0E	0000 1110
0011	LODA,2 *H'51'	80	1000 0000
0012		51	0101 0001
.			
.			
0051		12	0001 0010
0052		34	0011 0100
.			
.			
1234		99	1001 1001

Durch diesen Befehl wird das Register 2 mit H'99' geladen.

Indirekte Indizierte Adressierung

Sie ist nur bei absoluter Adressierung möglich. Es wird hier zuerst die indirekte Adresse und anschließend die indizierte Adresse berechnet.

BEFEHLSFORMAT



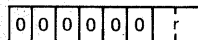
DETAILED PROCESSOR INSTRUCTIONS

LOAD REGISTER ZERO

(Register Addressing)

Mnemonic LODZ

Binary Coding



7 6 5 4 3 2 1 0

Execution Time 2 cycles (6 clock periods)

Description

Dieser 1-byte-Befehl transferiert den Inhalt des spezifizierten Registers r, in das Register 0. Der ursprüngliche Inhalt des spezifizierten Register 0 wird überschrieben. Der Inhalt des Registers r bleibt unverändert.

Processor Registers Affected

CC

Condition Code Setting

Register Zero	CC1	CC0
Positive	0	1
Zero	0	0
Negative	1	0

LOAD IMMEDIATE

(Immediate Addressing)

Mnemonic LODI,r v

Binary Coding



7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0

Execution Time 2 cycles (6 clock periods)

Description

Dieser 2-byte-Befehl transferiert den Inhalt v des zweiten Bytes dieser Instruktion in das spezifizierte Register r. Der ursprüngliche Inhalt von r wird überschrieben.

Processor Registers Affected

CC

Condition Code Setting

Register r	CC1	CC0
Positive	0	1
Zero	0	0
Negative	1	0

LOAD RELATIVE

(Relative Addressing)

Mnemonic LODR,r (*)a

Binary Coding



7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0

Execution Time 3 cycles (9 clock periods)

Description

Dieser 2-byte-Befehl transferiert ein Daten-Byte vom Speicher in das spezifizierte Register r. Die effektive Adresse des Daten-Bytes wird durch die Addition des Feldes a mit der Adresse des auf diesen Befehl folgenden Befehls gefunden. Der ursprüngliche Inhalt des Registers r wird überschrieben. Indirekte Adressierung ist möglich.

Processor Registers Affected

CC

Condition Code Setting

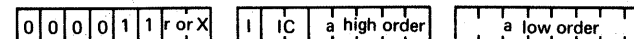
Register r	CC1	CC0
Positive	0	1
Zero	0	0
Negative	1	0

LOAD ABSOLUTE

(Absolute Addressing)

Mnemonic LODA,r (*)a(X)

Binary Coding



7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0

Execution Time 4 cycles (12 clock periods)

Description

Dieser 3-byte-Befehl transferiert ein Daten-Byte vom Speicher in das spezifizierte Register r. Das Daten-Byte wird vom Speicherplatz der absoluten Adresse genommen. Wurde indizierte Adressierung spezifiziert, geben Bit 0 und 1 von Byte 0 das Indexregister an. In diesem Fall wird das Rechenregister immer das Register 0 sein. Der ursprüngliche Inhalt des Registers r wird überschrieben. Indirekte Adressierung und/oder indizierte Adressierung ist möglich.

Processor Registers Affected

CC

Condition Code Setting

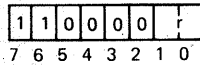
Register r	CC1	CC0
Positive	0	1
Zero	0	0
Negative	1	0

STORE REGISTER ZERO

(Register Addressing)

Mnemonic STRZ r

Binary Code



Execution Time 2 cycles (6 clock periods)

Description

Dieser 1-byte-Befehl transferiert den Inhalt des Registers 0 in das spezifizierte Register r. Der ursprüngliche Inhalt von Register r wird überschrieben. Der Inhalt des Registers 0 bleibt unverändert.

Merke: Als Register r darf nie Register 0 angegeben werden. Dieser "1100 0000"-Operationscode ist für den Befehl NOP reserviert.

Processor Registers Affected CC

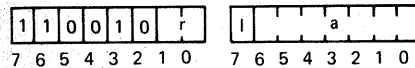
Condition Code Setting	Register r.	CC1	CC0
Positive	0	0	1
Zero	0	0	0
Negative	1	1	0

STORE RELATIVE

(Relative Addressing)

Mnemonic STRR,r (*a)

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Dieser 2-byte-Befehl transferiert ein Daten-Byte vom spezifizierten Register r zu dem, durch die effektive Adresse sich ergebenden Speicherplatz. Der Inhalt des Registers r bleibt unverändert. Der Inhalt des Speicherplatzes wird überschrieben. Indirekte Adressierung ist möglich.

Processor Registers Affected None

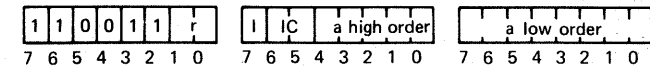
Condition Code Setting N/A

STORE ABSOLUTE

(Absolute Addressing)

Mnemonic STRA,r (*a),X)

Binary Code



Execution Time 4 cycles (12 clock periods)

Description

Dieser 3-byte-Befehl transferiert ein Daten-Byte vom angegebenen Register r in einen Speicherplatz, welcher durch die effektive Adresse angegeben wird. Der Inhalt des Registers r wird nicht verändert. Der Inhalt des Speicherplatzes wird überschrieben.

Indirekte Adressierung und/oder indizierte Adressierung ist möglich.

Wird indizierte Adressierung angegeben, geben Bit 0 und 1 von Byte 0 das Indexregister an. Das Rechenregister ist in diesem Fall immer das Register 0.

Processor Registers Affected None

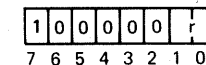
Condition Code Setting N/A

ADD TO REGISTER ZERO

(Register Addressing)

Mnemonic ADDZ r

Binary Code



Execution Time 2 cycles (6 clock periods)

Description

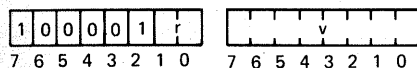
Durch diesen 1-byte-Befehl wird der Inhalt des spezifizierten Registers r zum Inhalt des Registers 0 addiert. Die 8-bit-Summe dieser Addition überschreibt den ursprünglichen Inhalt des Registers 0. Der Inhalt des Registers r bleibt unverändert. Merke: Addition mit Carry ist möglich. Siehe Carry-Bit.

Processor Registers Affected C, CC, IDC, OVF

Condition Code Setting	Register Zero	CC1	CC0
Positive	0	0	1
Zero	0	0	0
Negative	1	1	0

ADD IMMEDIATE

(Immediate Addressing)

Mnemonic ADDI,r v**Binary Coding****Execution Time** 2 cycles (6 clock periods)**Description**

Durch diesen 2-byte-Befehl wird das zweite Byte dieser Instruktion, v, zum Inhalt des Registers r addiert. Die 8-bit-Summe überschreibt den Inhalt des Registers r.

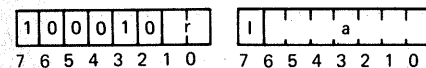
Merke: Addition mit Carry ist möglich. Siehe Carry-Bit.

Processor Registers Affected C, CC, IDC, OVF

Condition Code Setting	Register r	CC1	CC0
Positive	0	1	
Zero	0	0	
Negative	1	0	

ADD RELATIVE

(Relative Addressing)

Mnemonic ADDR,r (*a)**Binary Coding****Execution Time** 3 cycles (9 clock periods)**Description**

Durch diesen 2-byte-Befehl werden der Inhalt des Registers r und der Inhalt eines Speicherplatzes, welcher durch die effektive Adresse angegeben wird, addiert. Die 8-bit-Summe überschreibt den Inhalt des Registers r.

Indirekte Adressierung ist möglich.

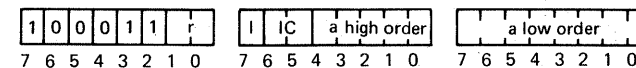
Merke: Addition mit Carry ist möglich. Siehe Carry-Bit.

Processor Registers Affected C, CC, IDC, OVF

Condition Code Setting	Register r	CC1	CC0
Positive	0	1	
Zero	0	0	
Negative	1	0	

ADD ABSOLUTE

(Absolute Addressing)

Mnemonic ADDA,r (*a,X)**Binary Coding****Execution Time** 4 cycles (12 clock periods)**Description**

Durch diesen 3-byte-Befehl werden der Inhalt des Registers r und der Inhalt des Speicherplatzes, welcher durch die effektive Adresse angegeben wird, addiert. Die 8-bit-Summe überschreibt den Inhalt des Registers r.

Indirekte Adressierung und/oder indizierte Adressierung ist möglich.

Wurde indizierte Adressierung angegeben, geben Bit 0 und 1 von Byte 0 das Indexregister an. Das Rechenregister dieser Operation wird dann immer das Register 0 sein.

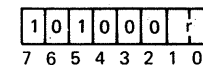
Merke: Addition mit Carry ist möglich. Siehe Carry-Bit.

Processor Registers Affected C, CC, IDC, OVF

Condition Code Setting	Register r	CC1	CC0
Positive	0	1	
Zero	0	0	
Negative	1	0	

SUBTRACT FROM REGISTER ZERO

(Register Addressing)

Mnemonic SUBZ r**Binary Coding****Execution Time** 2 cycles (6 clock periods)**Description**

Durch diesen 1-byte-Befehl wird der Inhalt des Registers r vom Inhalt des Registers 0 subtrahiert. Das Resultat dieser Subtraktion überschreibt den Inhalt des Registers 0.

Die Subtraktion erfolgt durch Addition des Zweier-Komplements des Registers r zum Inhalt des Registers 0. Der Inhalt des Registers r wird nicht verändert. Merke: Subtraktion mit Borrow ist möglich. Siehe Carry-Bit.

Processor Registers Affected C, CC, IDC, OVF

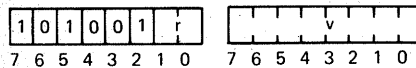
Condition Code Setting	Register Zero	CC1	CC0
Positive	0	1	
Zero	0	0	
Negative	1	0	

SUBTRACT IMMEDIATE

(Immediate Addressing)

Mnemonic SUBI,r v

Binary Code



Execution Time 2 cycles (6 clock periods)

Description

Durch diesen 2-byte-Befehl wird das zweite Byte dieser Instruktion, v, vom Inhalt des Registers r subtrahiert. Das Ergebnis dieser Subtraktion überschreibt den Inhalt des Registers r.

Die Subtraktion erfolgt durch Addition des Zweier-Komplements des zweiten Bytes zum Register r.

Merke: Subtraktion mit Borrow ist möglich. Siehe Carry-Bit.

Processor Registers Affected

C, CC, IDC, OVF

Condition Code Setting

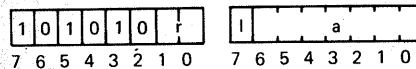
Register r	CC1	CC0
Positive	0	1
Zero	0	0
Negative	1	0

SUBTRACT RELATIVE

(Relative Addressing)

Mnemonic SUBR,r (*a)

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 2-byte-Befehl wird der Inhalt des durch die effektive Adresse bezeichneten Speicherplatzes vom Inhalt des Registers r subtrahiert. Das Ergebnis dieser Subtraktion überschreibt den Inhalt des Registers r.

Die Subtraktion erfolgt durch Addition des Zweierkomplements des adressierten Speicherplatzes zum Inhalt des Registers r.

Indirekte Adressierung ist möglich.

Merke: Subtraktion mit Borrow ist möglich. Siehe Carry-Bit.

Processor Registers Affected

C, CC, IDC, OVF

Condition Code Setting

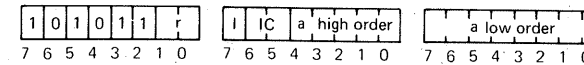
Register r	CC1	CC0
Positive	0	1
Zero	0	0
Negative	1	0

SUBTRACT ABSOLUTE

(Absolute Addressing)

Mnemonic SUBA,r (*a)(X)

Binary Code



Execution Time 4 cycles (12 clock periods)

Description

Durch diesen 3-byte-Befehl wird der Inhalt des Speicherplatzes, welcher durch die effektive Adresse angegeben wird, vom Inhalt des Registers r subtrahiert. Das Ergebnis dieser Subtraktion überschreibt den Inhalt des Registers r.

Die Subtraktion erfolgt durch Addition des Zweier-Komplements des angegebenen Speicherplatzes zum Register r.

Indirekte Adressierung und/oder indizierte Adressierung ist möglich.

Bei indizierter Adressierung geben Bit 0 und 1 von Byte 0 das Indexregister an.

Das Rechenregister wird in diesem Fall immer das Register 0 sein.

Merke: Subtraktion mit Borrow ist möglich. Siehe Carry-Bit.

Processor Registers Affected

C, CC, IDC, OVF

Condition Code Setting

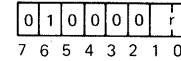
Register r	CC1	CC0
Positive	0	1
Zero	0	0
Negative	1	0

AND TO REGISTER ZERO

(Register Addressing)

Mnemonic ANDZ r

Binary Code



Execution Time 2 cycles (6 clock periods)

Description

Durch diesen 1-byte-Befehl werden der Inhalt des Registers r und der Inhalt des Registers 0 durch die logische AND-Funktion verknüpft. Das Ergebnis dieser Operation überschreibt den Inhalt des Registers 0. Der Inhalt des Registers r bleibt unverändert.

Der AND-Befehl behandelt jedes der acht Argument-Bits auf folgende Weise:

Bit (0-7)	Bit (0-7)	AND Result
0	0	0
0	1	0
1	1	1
1	0	0

Merke: Als Register r darf nie das Register 0 angegeben werden. Dieser Operationscode '0100 0000' ist für den Befehl HALT reserviert.

Processor Registers Affected

CC

Condition Code Setting

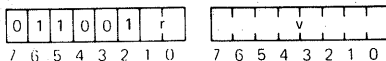
Register Zero	CC1	CC0
Positive	0	1
Zero	0	0
Negative	1	0

INCLUSIVE OR IMMEDIATE

(Immediate Addressing)

Mnemonic IORI,r v

Binary Code



Execution Time 2 cycles (6 clock periods)

Description

Durch diesen 2-byte-Befehl werden der Inhalt des Registers r und der Inhalt des zweiten Bytes dieser Instruktion, v, durch die logische OR-Funktion verknüpft. Das Ergebnis dieser Operation überschreibt den Inhalt des Registers r. Der OR-Befehl behandelt jedes der acht Argument-Bits auf folgende Weise:

Bit (0-7)	Bit (0-7)	Inclusive OR Result
0	0	0
0	1	1
1	1	1
1	0	1

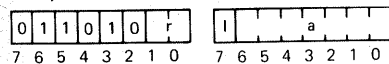
Processor Registers Affected	CC	
Condition Code Setting	Register r	
	CC1	CC0
Positive	0	1
Zero	0	0
Negative	1	0

INCLUSIVE OR RELATIVE

(Relative Addressing)

Mnemonic IOIR,r (*a)

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 2-byte-Befehl werden der Inhalt des Registers r und der Inhalt des durch die effektive Adresse angegebenen Speicherplatzes durch die logische OR-Funktion verknüpft. Das Ergebnis dieser Operation überschreibt den Inhalt des Registers r. Indirekte Adressierung ist möglich.

Der OR-Befehl behandelt jedes der acht Argument-Bits auf folgende Weise:

Bit (0-7)	Bit (0-7)	Inclusive OR Result
0	0	0
0	1	1
1	1	1
1	0	1

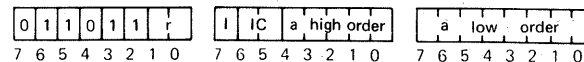
Processor Registers Affected	CC	
Condition Code Setting	Register r	
	CC1	CC0
Positive	0	1
Zero	0	0
Negative	1	0

INCLUSIVE OR ABSOLUTE

(Absolute Addressing)

Mnemonic IORA,r (*a),X

Binary Code



Execution Time 4 cycles (12 clock periods)

Description

Durch diesen 3-byte-Befehl werden der Inhalt des Registers r und der Inhalt des durch die effektive Adresse angegebenen Speicherplatzes durch die logische OR-Funktion verknüpft. Das Ergebnis dieser Operation überschreibt den Inhalt des Registers r.

Der OR-Befehl behandelt jedes der acht Argument-Bits auf folgende Weise:

Bit (0-7)	Bit (0-7)	Inclusive OR Result
0	0	0
0	1	1
1	1	1
1	0	1

Indirekte Adressierung und/oder indizierte Adressierung ist möglich. Bei indizierter Adressierung geben Bit 0 und 1 von Byte 0, das Indexregister an. Das Rechenregister wird in diesem Fall immer das Register 0 sein.

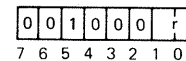
Processor Registers Affected	CC	
Condition Code Setting	Register Zero	
	CC1	CC0
Positive	0	1
Zero	0	0
Negative	1	0

EXCLUSIVE OR TO REGISTER ZERO

(Register Addressing)

Mnemonic EORZ r

Binary Code



Execution Time 2 cycles (6 clock periods)

Description

Durch diesen 1-byte-Befehl werden der Inhalt des spezifizierten Registers r und der Inhalt des Register 0 durch die logische EOR-Funktion verknüpft. Das Ergebnis dieser Operation überschreibt den Inhalt des Registers 0. Der Inhalt des Registers r bleibt unverändert.

Der EOR-Befehl behandelt jedes der acht Argument-Bits auf folgende Weise:

Bit (0-7)	Bit (0-7)	Exclusive OR Result
0	0	0
0	1	1
1	1	0
1	0	1

Processor Registers Affected	CC	
Condition Code Setting	Register Zero	
	CC1	CC0
Positive	0	1
Zero	0	0
Negative	1	0

EXCLUSIVE OR IMMEDIATE

(Immediate Addressing)

Mnemonic EORI,r v

Binary Code



Execution Time 2 cycles (6 clock periods)

Description

Durch diesen 2-byte-Befehl werden der Inhalt des Registers r und der Inhalt des zweiten Bytes dieser Instruktion, v, durch die logische EOR-Funktion verknüpft. Das Ergebnis dieser Operation überschreibt den Inhalt des Registers r. Der EOR-Befehl behandelt jedes der acht Argument-Bits auf folgende Weise:

Bit (0-7)	Bit (0-7)	Exclusive OR Result
0	0	0
0	1	1
1	1	0
1	0	1

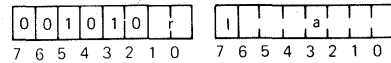
Processor Registers Affected	CC		
Condition Code Setting	Register r	CC1	CC0
	Positive	0	1
	Zero	0	0
	Negative	1	0

EXCLUSIVE OR RELATIVE

(Relative Addressing)

Mnemonic EORR,r (*a)

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch dieser 2-byte-Befehl werden der Inhalt des Registers r und der Inhalt der Speicherzelle, welche durch die effektive Adresse angegeben wird, durch die logische EOR-Funktion verknüpft. Das Ergebnis dieser Operation überschreibt den Inhalt des Registers r.

Indirekte Adressierung ist möglich.

Der EOR-Befehl behandelt jedes der acht Argument-Bits auf folgende Weise:

Bit (0-7)	Bit (0-7)	Exclusive OR Result
0	0	0
0	1	1
1	1	0
1	0	1

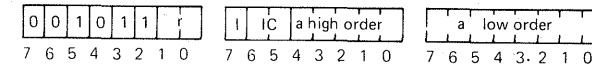
Processor Registers Affected	CC		
Condition Code Setting	Register r	CC1	CC0
	Positive	0	1
	Zero	0	0
	Negative	1	0

EXCLUSIVE OR ABSOLUTE

(Absolute Addressing)

Mnemonic EORA,r (*a,X)

Binary Code



Execution Time 4 cycles (12 clock periods)

Description

Durch diesen 3-byte-Befehl werden der Inhalt des Registers r und der Inhalt der Speicherzelle, welche durch die effektive Adresse angegeben wird, durch die logische EOR-Funktion verknüpft. Das Ergebnis dieser Operation überschreibt den Inhalt des Registers r. Der EOR-Befehl behandelt jedes der acht Argument-Bits auf folgende Weise:

Bit (0-7)	Bit (0-7)	Exclusive OR Result
0	0	0
0	1	1
1	1	0
1	0	1

Indirekte Adressierung und/oder indizierte Adressierung ist möglich. Bei indizierter Adressierung geben Bit 0 und 1 von Byte 0 das Indexregister an. Das Rechenregister wird in diesem Fall immer das Register 0 sein.

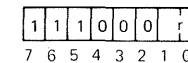
Processor Registers Affected	CC		
Condition Code Setting	Register r	CC1	CC0
	Positive	0	1
	Zero	0	0
	Negative	1	0

COMPARE TO REGISTER ZERO

(Register Addressing)

Mnemonic COMZ r

Binary Code



Execution Time 2 cycles (6 clock periods)

Description

Durch diesen 1-byte-Befehl wird der Inhalt des Registers r mit dem Inhalt des Registers 0 verglichen. Der Vergleich kann entweder im "arithmetic"- oder im "logical"-Mode, abhängig vom COM-Bit des Programm-Status-Wortes erfolgen. Ist COM=1 (logical mode) werden die Daten als positive Acht-Bit-Zahl interpretiert. Ist COM=0 (arithmetic mode) werden die Daten als Acht-Bit-Zweier-Komplement-Zahl interpretiert. Durch diesen Befehl wird nur der Condition-Code im Programm-Status-Wort auf folgende Weise gesetzt:

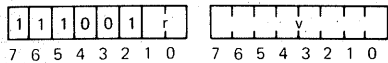
Processor Registers Affected	CC		
Condition Code Setting	Register zero greater than Register r	0	1
	Register zero equal to Register r	0	0
	Register zero less than Register r	1	0

COMPARE IMMEDIATE

(Immediate Addressing)

Mnemonic COMI,r v

Binary Code



Execution Time 2 cycles (6 clock periods)

Description

Durch diesen 2-byte-Befehl wird der Inhalt des spezifizierten Registers r mit dem Inhalt des zweiten Bytes dieser Instruktion, v, verglichen. Der Vergleich kann entweder im "arithmetic"- oder im "logical"-Mode, abhängig vom COM-Bit des Programm-Status-Wortes, erfolgen.

Ist COM=1 (logical mode) werden die Daten als positive Acht-Bit-Zahl interpretiert. Ist COM=0 (arithmetic mode) werden die Daten als Acht-Bit-Zweierkomplement-Zahl interpretiert.

Durch diesen Befehl wird nur der Condition-Code im Programm-Status-Wort auf folgende Weise geändert:

Processor Registers Affected CC

Condition Code Setting	CC1	CC0
Register r greater than v	0	1
Register r equal to v	0	0
Register r less than v	1	0

COMPARE RELATIVE

(Relative Addressing)

Mnemonic COMR,r (*a)

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 2-byte-Befehl wird der Inhalt des spezifizierten Registers r mit dem Inhalt der Speicherzelle, welche durch die effektive Adresse angegeben wird, verglichen. Der Vergleich kann im "arithmetic"- oder im "logical"- Mode, abhängig vom COM-Bit des Programm-Status-Wortes, erfolgen.

Ist COM=1 (logical mode) werden die Daten als positive Acht-Bit-Zahl interpretiert. Ist COM=0 (arithmetic mode) werden die Daten als Acht-Bit-Zweierkomplement-Zahl interpretiert.

Durch diesen Befehl wird nur der Condition-Code im Programm-Status-Wort auf folgende Weise geändert:

Processor Registers Affected CC

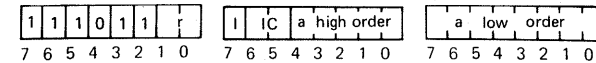
Condition Code Setting	CC1	CC0
Register r greater than memory byte	0	1
Register r equal to memory byte	0	0
Register r less than memory byte	1	0

COMPARE ABSOLUTE

(Absolute Addressing)

Mnemonic COMA,r (*a)(X)

Binary Code



Execution Time 4 cycles (12 clock periods)

Description

Durch diesen 3-byte-Befehl wird der Inhalt des Registers r mit dem Inhalt der Speicherzelle, welche durch die effektive Adresse angegeben wird, verglichen. Ist COM=1 (logical mode) werden die Daten als positive Acht-Bit-Zahl interpretiert. Ist COM=0 (arithmetic mode) werden die Daten als Acht-Bit-Zweierkomplement-Zahl interpretiert. Durch diesen Befehl wird nur der Condition-Code im Programm-Status-Wort auf folgende Weise geändert:

Processor Registers Affected CC

Condition Code Setting	CC1	CC0
Register r greater than memory byte	0	1
Register r equal to memory byte	0	0
Register r less than memory byte	1	0

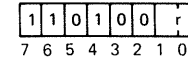
Indirekte und/oder indizierte Adressierung ist möglich. Bei indizierter Adressierung geben Bit 0 und 1 von Byte 0 das Indexregister an. Das Rechenregister wird in diesem Fall immer das Register 0 sein.

ROTATE REGISTER LEFT

(Register Addressing)

Mnemonic RRL,r

Binary Code



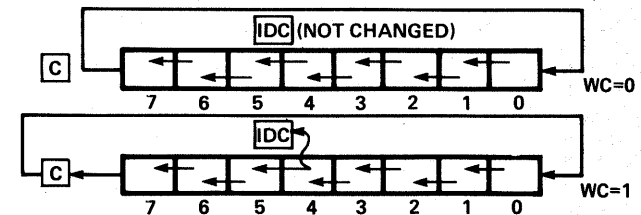
Execution Time 2 cycles (6 clock periods)

Description

Durch diesen 1-byte-Befehl wird der Inhalt des spezifizierten Registers r um eine Stelle nach links rotiert. Ist im Programm-Status-Wort WC=0 (without carry), dann wandert das Bit 7 des Registers r in das Bit 0.

Ist WC=1 (with carry), dann wandert das Bit 7 in das Carry-Bit und das ursprüngliche Carry-Bit ersetzt das Bit 0.

Das Bit 4 des Registers r wandert bei WC=1 in das IDC-Bit.



Merke: Immer wenn durch einen Rotierbefehl das Bit 7 sein Vorzeichen verändert, wird das OVF-Bit im Programm-Status-Wort gesetzt.

Processor Registers Affected C, CC, IDC, OVF

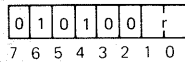
Condition Code Setting	CC1	CC0
Register r Positive	0	1
Register r Zero	0	0
Register r Negative	1	0

ROTATE REGISTER RIGHT

(Register Addressing)

Mnemonic RRR,r

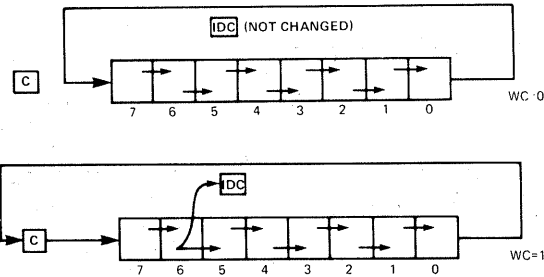
Binary Code



Execution Time 2 cycles (6 clock periods)

Description

Durch diesen 1-byte-Befehl wird der Inhalt des spezifizierten Registers r um eine Stelle nach rechts rotiert. Ist im Programm-Status-Wort WC=0 (without carry), dann wandert das Bit 0 des Registers r in das Bit 7. Ist WC=1 (with carry), dann wandert das Bit 0 in das Carry-Bit und das ursprüngliche Carry-Bit ersetzt das Bit 7. Das Bit 6 des Registers wandert bei WC=1 in das IDC-Bit.



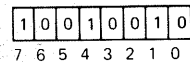
Merke: Immer, wenn durch einen Rotierbefehl das Bit 7 sein Vorzeichen verändert, wird das OVF-Bit im Programm-Status-Wort gesetzt.

Processor Registers Affected	C, CC, IDC, OVF	
Condition Code Setting	Register r	CC1 CC0
	Positive	0 1
	Zero	0 0
	Negative	1 0

LOAD PROGRAM STATUS, UPPER

Mnemonic LPSU

Binary Code



Execution Time 2 cycles (6 clock periods)

Description

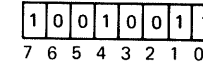
Durch diesen 1-byte-Befehl wird der jeweilige Inhalt des oberen Programm-Status-Wortes durch den Inhalt des Registers 0 ersetzt. Die Bits 3 und 4 des oberen Programm-Status-Wortes bleiben immer 0. Siehe PSU.

Processor Registers Affected	F, II, SP
Condition Code Setting	N/A

LOAD PROGRAM STATUS, LOWER

Mnemonic LPSL

Binary Code



Execution Time 2 cycles (6 clock periods)

Description

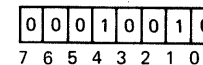
Durch diesen 1-byte-Befehl wird der jeweilige Inhalt des unteren Programm-Status-Wortes durch den Inhalt des Registers 0 ersetzt. Siehe PSL.

Processor Registers Affected	CC, IDC, RS, WC, OVF, COM, C
Condition Code Setting	The CC will take on the value in bits #7 and #6 of register zero.

STORE PROGRAM STATUS, UPPER

Mnemonic SPSU

Binary Code



Execution Time 2 cycles (6 clock periods)

Description

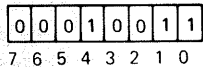
Durch diesen 1-byte-Befehl wird der jeweilige Inhalt des oberen Programm-Status-Wortes in das Register 0 übertragen. Bit 3 und 4 des oberen Programm-Status-Wortes werden immer 0 sein. Siehe PSU.

Processor Registers Affected	CC		
Condition Code Setting	Register Zero	CC1	CC0
	Positive	0	1
	Zero	0	0
	Negative	1	0

STORE PROGRAM STATUS, LOWER

Mnemonic SPSL

Binary Code



Execution Time 2 cycles (6 clock periods)

Description

Durch diesen 1-byte-Befehl wird der jeweilige Inhalt des unteren Programm-Status-Wortes in das Register 0 übertragen. Siehe PSL.

Processor Registers Affected

CC

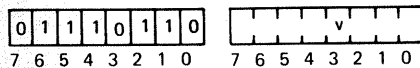
Condition Code Setting

Register Zero	CC1	CC0
Positive	0	1
Zero	0	0
Negative	1	0

PRESET PROGRAM STATUS UPPER, SELECTIVE (Immediate Addressing)

Mnemonic PPSU v

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 2-byte-Befehl können selektiv beliebige Bits des oberen Programm-Status-Wortes auf 1 gesetzt werden. Bei der Durchführung dieses Befehls werden alle Bits des v-Bytes auf das Vorkommen einer 1 geprüft. Enthält ein Bit eine 1, so wird das zugehörige Bit des oberen Programm-Status-Wortes ebenfalls auf 1 gesetzt. Alle jene Bits die im v-Byte eine 0 enthalten, werden im Programm-Status-Wort nicht verändert.

Processor Registers Affected

F, II, SP

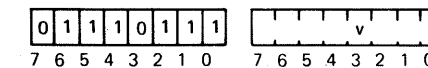
Condition Code Setting

N/A

PRESET PROGRAM STATUS LOWER, SELECTIVE (Immediate Addressing)

Mnemonic PPSL v

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 2-byte-Befehl können selektiv beliebige Bits des unteren Programm-Status-Wortes auf 1 gesetzt werden. Bei der Durchführung dieses Befehls werden alle Bits des v-Bytes auf das Vorkommen einer 1 geprüft. Enthält ein Bit eine 1, so wird das zugehörige Bit des unteren Programm-Status-Wortes ebenfalls auf 1 gesetzt. Alle jene Bits, die im v-Byte eine 0 enthalten, werden im Programm-Status-Wort nicht verändert.

Processor Registers Affected

CC, IDC, RS, WC, OVF, COM, C

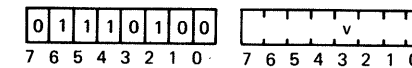
Condition Code Setting

The CC bits may be set by the execution of this instruction.

CLEAR PROGRAM STATUS UPPER, SELECTIVE (Immediate Addressing)

Mnemonic CPSU v

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 2-byte-Befehl können selektiv beliebige Bits des oberen Programm-Status-Wortes auf 0 gesetzt werden. Bei der Durchführung dieses Befehls werden alle Bits des v-Bytes auf das Vorkommen einer 1 geprüft. Enthält ein Bit eine 1, so wird das zugehörige Bit des oberen Programm-Status-Wortes auf 0 gesetzt. Alle jene Bits, die im v-Byte eine 0 enthalten, werden im Programm-Status-Wort nicht verändert.

Processor Registers Affected

F, II, SP

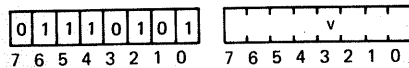
Condition Code Setting

N/A

CLEAR PROGRAM STATUS LOWER, SELECTIVE (Immediate Addressing)

Mnemonic CPSL v

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 2-byte-Befehl können selektiv beliebige Bits des unteren Programm-Status-Wortes auf 0 gesetzt werden. Bei der Durchführung dieses Befehls werden alle Bits des v-Bytes auf das Vorkommen einer 1 geprüft. Enthält ein Bit eine 1, so wird das zugehörige Bit des unteren Programm-Status-Wortes auf 0 gesetzt. Alle jene Bits, die im v-Byte eine 0 enthalten, werden im Programm-Status-Wort nicht verändert.

Processor Registers Affected CC, IDC, RS, WC, OVF, COM, C

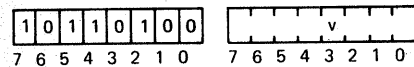
Condition Code Setting

Durch diesen Befehl kann der Condition-Code gelöscht werden.

TEST PROGRAM STATUS UPPER, SELECTIVE (Immediate Addressing)

Mnemonic TPSU v

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 2-byte-Befehl können selektiv beliebige Bits des oberen Programm-Status-Wortes auf das Vorkommen einer 1 getestet werden. Während der Durchführung dieses Befehls werden alle Bits des v-Bytes auf das Vorkommen einer 1 getestet. Enthält ein Bit eine 1, so wird auch das zugehörige Bit des oberen Programm-Status-Wortes auf das Vorkommen einer 1 untersucht. Der Condition-Code enthält das Ergebnis dieser Operation. Ist ein Bit im v-Byte gleich 0, so wird das zugehörige Bit im Status-Wort nicht untersucht.

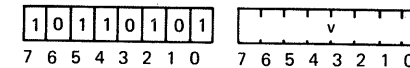
Processor Registers Affected CC

Condition Code Setting	CC1	CC0
All of the selected bits in PSU are 1s	0	0
Not all of the selected bits in PSU are 1s	1	0

TEST PROGRAM STATUS LOWER, SELECTIVE (Immediate Addressing)

Mnemonic TPSL v

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 2-byte-Befehl können beliebige Bits des unteren Programm-Status-Wortes auf das Vorkommen einer 1 getestet werden. Während der Durchführung dieses Befehls werden alle Bits des v-Bytes auf das Vorkommen einer 1 getestet. Enthält ein Bit eine 1, so wird auch das zugehörige Bit des unteren Status-Wortes auf das Vorkommen einer 1 untersucht. Der Condition-Code enthält das Ergebnis dieser Operation. Ist ein Bit im v-Byte gleich 0, so wird das zugehörige Bit im Status-Wort nicht untersucht.

Processor Registers Affected CC

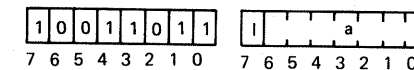
Condition Code Setting

	CC1	CC0
All of the selected bits in PSL are 1s	0	0
Not all of the selected bits in PSL are 1s	1	0

ZERO BRANCH RELATIVE (Relative Addressing)

Mnemonic ZBRR (*a)

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen unbedingten relativen 2-byte-Sprungbefehl wird die effektive Adresse anders als üblich berechnet. Der angegebene Wert a wird als relatives "Displacement" von Adresse 0 der Page 0 interpretiert. Das "Displacement" kann von -64 bis +63 angegeben werden. Die Adressberechnung erfolgt modulo 8192_{10} , also wird ein negatives "Displacement" auf das Ende von Page 0 hinweisen, z.B. ZBRR -8, ergibt einen unbedingten Sprung zur Adresse 8184_{10} , und ZBRR +52 ergibt einen Sprung zur Adresse 52_{10} . Durch diese Instruktion werden die Adreß-Bits 13 und 14, die Page-Adreß-Bits, gelöscht. Ebenso wird der Inhalt des Instruction-Address-Registers durch die effektive Adresse ersetzt. Dieser Befehl darf im gesamten adressierbaren Speicherbereich angewendet werden. Er wird immer zur Page 0 führen. Indirekte Adressierung ist möglich.

Processor Registers Affected None

Condition Code Setting N/A

BRANCH ON CONDITION TRUE, RELATIVE

(Relative Addressing)

Mnemonic BCTR,v (*)a

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen bedingten 2-byte-Sprungbefehl wird der Prozessor veranlaßt, die nächste auszuführende Instruktion von dem durch die effektive Adresse angegebenen Speicherplatz zu holen, aber nur dann, wenn die zwei Bits des v-Feldes mit dem momentanen Condition-Code im Programm-Status-Wort übereinstimmen.

Stimmt das v-Feld und der Condition-Code nicht überein, so wird als nächster Befehl der im Programm folgende ausgeführt (kein Sprung).

Indirekte Adressierung ist möglich.

Beinhaltet das v-Feld 3_{16} , so wird ein unbedingter Sprung durchgeführt.

Processor Registers Affected None

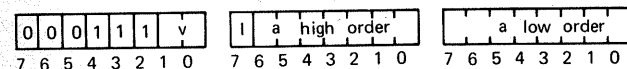
Condition Code Setting N/A

BRANCH ON CONDITION TRUE, ABSOLUTE

(Absolute Addressing)

Mnemonic BCTA,v (*)a

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen bedingten 3-byte-Sprungbefehl wird der Prozessor veranlaßt, die nächste auszuführende Instruktion von dem durch die effektive Adresse gegebenen Speicherplatz zu holen, aber nur dann, wenn die zwei Bits des v-Feldes mit dem momentanen Condition-Code im Programm-Status-Wort übereinstimmen.

Stimmt das v-Feld und der Condition-Code nicht überein, so wird der im Programm folgende Befehl ausgeführt (kein Sprung).

Indirekte Adressierung ist möglich.

Setzt man das v-Feld auf 3_{16} , so wird ein unbedingter Sprung durchgeführt.

Processor Registers Affected None

Condition Code Setting N/A

BRANCH ON CONDITION FALSE, RELATIVE

(Relative Addressing)

Mnemonic BCFR,v (*)a

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen bedingten 2-byte-Sprungbefehl wird der Prozessor veranlaßt, die nächste auszuführende Instruktion von dem durch die effektive Adresse gegebenen Speicherplatz zu holen, aber nur dann, wenn die zwei Bits des v-Feldes mit dem augenblicklichen Condition-Code im Programm-Status-Wort nicht übereinstimmen. Herrscht also keine Übereinstimmung, so wird der Inhalt des Instruction-Address-Registers (IAR) durch die berechnete effektive Adresse ersetzt.

Stimmt das v-Feld mit dem Condition-Code überein, so wird der im Programm folgende Befehl ausgeführt (kein Sprung).

Indirekte Adressierung ist möglich.

Das v-Feld darf nicht auf 3_{16} gesetzt werden. Dieser Operationscode ist für den Befehl ZBRR reserviert.

Processor Registers Affected None

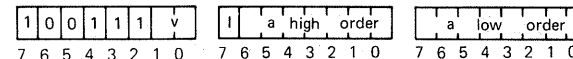
Condition Code Setting N/A

BRANCH ON CONDITION FALSE, ABSOLUTE

(Absolute Addressing)

Mnemonic BCFA,v (*)a

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen bedingten 3-byte-Sprungbefehl wird der Prozessor veranlaßt, die nächste auszuführende Instruktion von dem durch die effektive Adresse gegebenen Speicherplatz zu holen, aber nur dann, wenn die zwei Bits des v-Feldes mit dem augenblicklichen Condition-Code im Programm-Status-Wort nicht übereinstimmen. Herrscht also keine Übereinstimmung, so wird der Inhalt des Instruction-Address-Registers (IAR) durch die berechnete effektive Adresse ersetzt.

Stimmt das v-Feld mit dem Condition-Code überein, so wird der im Programm folgende Befehl ausgeführt (kein Sprung).

Indirekte Adressierung ist möglich.

Das v-Feld darf nicht auf 3_{16} gesetzt werden. Dieser Operationscode ist für den Befehl BXA reserviert.

Processor Registers Affected None

Condition Code Setting N/A

BRANCH ON INCREMENTING REGISTER, RELATIVE (Relative Addressing)

Mnemonic BIRR,r (*)a

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 2-byte-Sprungbefehl wird zunächst der Inhalt des spezifizierten Registers r um 1 erhöht. Ist der neue Wert des Registers r ungleich 0, so wird als nächster Befehl jene ausgeführt, der auf dem durch die effektive Adresse gegebenen Speicherplatz steht, d.h. die effektive Adresse ersetzt den Inhalt des Instruction-Address-Registers. Ist hingegen der neue Wert des Registers r gleich 0, so wird der im Programm folgende Befehl ausgeführt (kein Sprung).

Indirekte Adressierung ist möglich.

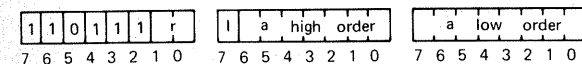
Processor Registers Affected None

Condition Code Setting N/A

BRANCH ON INCREMENTING REGISTER, ABSOLUTE (Absolute Addressing)

Mnemonic BIRA,r (*)a

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 3-byte-Sprungbefehl wird zunächst der Inhalt des spezifizierten Registers r um 1 erhöht. Ist der neue Wert des Registers r ungleich 0, so wird als nächster Befehl jener ausgeführt, der auf dem durch die effektive Adresse gegebenen Speicherplatz steht, d.h. die effektive Adresse ersetzt den Inhalt des Instruction-Address-Registers. Ist hingegen der neue Wert des Registers r gleich 0, so wird der im Programm folgende Befehl ausgeführt (kein Sprung).

Indirekte Adressierung ist möglich.

Processor Registers Affected None

Condition Code Setting N/A

BRANCH ON DECREMENTING REGISTER, RELATIVE (Relative Addressing)

Mnemonic BDRR,r (*)a

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 2-byte-Sprungbefehl wird zunächst der Inhalt des spezifizierten Registers r um 1 vermindert. Ist der neue Wert des Registers r ungleich 0, so wird als nächster Befehl jener ausgeführt, der auf dem durch die effektive Adresse gegebenen Speicherplatz steht, d.h. die effektive Adresse ersetzt den Inhalt des Instruction-Address-Registers. Ist hingegen der neue Wert des Registers r gleich 0, so wird der im Programm folgende Befehl ausgeführt (kein Sprung).

Indirekte Adressierung ist möglich.

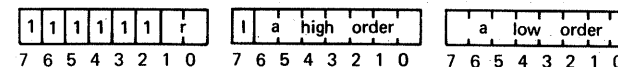
Processor Registers Affected None

Condition Code Setting N/A

BRANCH ON DECREMENTING REGISTER, ABSOLUTE (Absolute Addressing)

Mnemonic BDRA,r (*)a

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 3-byte-Sprungbefehl wird zunächst der Inhalt des spezifizierten Registers r um 1 vermindert. Ist der neue Wert des Registers r ungleich 0, so wird als nächster Befehl jener ausgeführt, der auf dem durch die effektive Adresse gegebenen Speicherplatz steht, d.h. die effektive Adresse ersetzt den Inhalt des Instruction-Address-Registers. Ist hingegen der neue Wert des Registers r gleich 0, so wird der im Programm folgende Befehl ausgeführt (kein Sprung).

Indirekte Adressierung ist möglich.

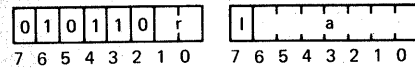
Processor Registers Affected None

Condition Code Setting N/A

BRANCH ON REGISTER NON-ZERO, RELATIVE (Relative Addressing)

Mnemonic BRNR,r (*)a

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 2-byte-Sprungbefehl wird der Inhalt des spezifizierten Registers r untersucht. Ist der Wert des Registers r ungleich 0, so wird als nächster Befehl jener ausgeführt, der auf dem durch die effektive Adresse gegebenen Speicherplatz zu finden ist, d.h. der Inhalt des Instruction-Address-Registers wird durch die effektive Adresse ersetzt. Ist der Wert des Registers r gleich 0, so wird der im Programm folgende Befehl ausgeführt (kein Sprung).
Indirekte Adressierung ist möglich.

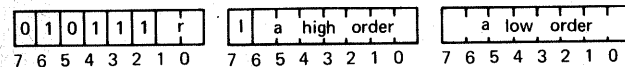
Processor Registers Affected None

Condition Code Setting N/A

BRANCH ON REGISTER NON-ZERO, ABSOLUTE (Absolute Addressing)

Mnemonic BRNA,r (*)a

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 3-byte-Sprungbefehl wird der Inhalt des spezifizierten Registers r untersucht. Ist der Wert des Registers r ungleich 0, so wird als nächster Befehl jener ausgeführt, der auf dem durch die effektive Adresse angegebenen Speicherplatz zu finden ist, d.h. der Inhalt des Instruction-Address-Registers wird durch die effektive Adresse ersetzt. Ist der Wert des Registers r gleich 0, so wird als nächstes der im Programm folgende Befehl ausgeführt (kein Sprung).
Indirekte Adressierung ist möglich.

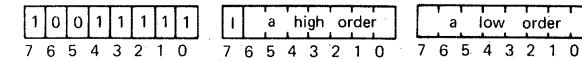
Processor Registers Affected None

Condition Code Setting N/A

BRANCH INDEXED, ABSOLUTE (Absolute Addressing)

Mnemonic BXA (*)a,X

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Dieser 3-byte-Befehl bewirkt einen unbedingten Sprung. Es wird die indizierte Adressierung verwendet und das Register 3 muß als Indexregister verwendet werden. Nach Ausführung dieses Befehls ist der Inhalt des Instruction-Address-Registers (IAR) durch die effektive Adresse ersetzt.
Wird zusätzlich indirekte Adressierung angegeben, so wird der Inhalt des Index-Registers (Register 3) zur indirekten Adresse addiert, um die effektive Adresse zu bekommen.

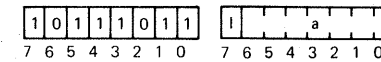
Processor Registers Affected None

Condition Code Setting N/A

ZERO BRANCH TO SUBROUTINE, RELATIVE (Relative Addressing)

Mnemonic ZBSR (*)a

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen unbedingten relativen 2-byte-Unterprogramm-Sprungbefehl wird die effektive Adresse anders als üblich berechnet. Der angegebene Wert a wird als relatives "Displacement" von Adresse 0, Page 0, interpretiert. Das "Displacement" kann von -64 bis +63 angegeben werden. Die Adreßberechnung erfolgt modulo 8192_{10} , also wird ein negatives "Displacement" auf das Ende von Page 0 hinweisen; z.B. ZBSR -8 ergibt einen Unterprogramm-Sprung zur Adresse 8184_{10} und ZBSR +52 ergibt einen Sprung zur Adresse 52_{10} .
Durch diese Instruktion werden die Adreßbits 13 und 14, die Page-Adreß-Bits, gelöscht. Ebenso wird der Inhalt des Instruction-Address-Registers durch die effektive Adresse ersetzt. Dieser Befehl darf im gesamten adressierbaren Speicherbereich angewendet werden. Er wird immer zu Page 0 führen.
Indirekte Adressierung ist möglich.
Während der Ausführung dieses Befehls wird der Stack-Pointer um 1 erhöht. Die Adresse des diesem Befehl folgenden Bytes (Inhalt von IAR) wird in den Return-Address-Stack (RAS) übertragen. Dann erst wird der Sprung ausgeführt.

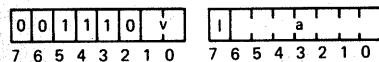
Processor Registers Affected SP

Condition Code Setting N/A

BRANCH TO SUBROUTINE ON CONDITION TRUE, RELATIVE (Relative Addressing)

Mnemonic BSTR,v (*a)

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Dies ist ein bedingter 2-byte-Unterprogramm-Sprungbefehl. Der Sprung zum Unterprogramm wird nur dann durchgeführt, wenn die zwei Bits des v-Feldes mit dem momentanen Condition-Code im Programm-Status-Wort übereinstimmen. In diesem Fall wird der Stack-Peinter um 1 erhöht und der momentane Inhalt des Instruction-Address-Registers, welcher auf das diesem Befehl folgende Byte hinweist, in den Return-Address-Stack geschrieben. Der vorherige Inhalt des Instruction-Address-Registers wird durch die effektive Adresse ersetzt (Sprung). Besteht keine Übereinstimmung zwischen dem v-Feld und dem Condition-Code, so wird der im Programm folgende Befehl ausgeführt. In diesem Fall bleibt der Stack-Peinter unverändert (kein Sprung).

Indirekte Adressierung ist möglich.

Setzt man das v-Feld auf 3₁₆, so wird ein unbedingter Unterprogramm-Sprung durchgeführt.

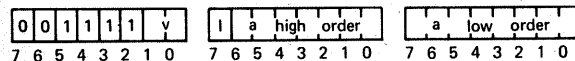
Processor Registers Affected SP

Condition Code Setting N/A

BRANCH TO SUBROUTINE ON CONDITION TRUE, ABSOLUTE (Absolute Addressing)

Mnemonic BSTA,v (*a)

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Dies ist ein bedingter 3-byte-Unterprogramm-Sprungbefehl. Der Sprung zum Unterprogramm wird durchgeführt, wenn die zwei Bits des v-Feldes mit dem momentanen Condition-Code im Programm-Status-Wort übereinstimmen. In diesem Fall wird der Stack-Peinter um 1 erhöht und der momentane Inhalt des Instruction-Address-Registers, welcher auf das diesem Befehl folgende Byte hinweist, in den Return-Address-Stack geschrieben. Der vorherige Inhalt des Instruction-Address-Registers wird durch die effektive Adresse ersetzt (Sprung). Besteht keine Übereinstimmung zwischen dem v-Feld und dem Condition-Code, so wird der im Programm folgende Befehl ausgeführt. In diesem Fall bleibt der Stack-Peinter unverändert (kein Sprung).

Indirekte Adressierung ist möglich.

Setzt man das v-Feld auf 3₁₆, so wird ein unbedingter Unterprogramm-Sprung durchgeführt.

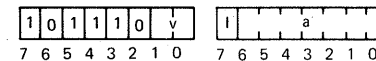
Processor Registers Affected SP

Condition Code Setting N/A

BRANCH TO SUBROUTINE ON CONDITION FALSE, RELATIVE (Relative Addressing)

Mnemonic BSFR,v (*a)

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Dies ist ein bedingter 2-byte-Unterprogramm-Sprungbefehl. Der Sprung zum Unterprogramm wird nur dann durchgeführt, wenn die zwei Bits des v-Feldes mit dem momentanen Condition-Code im Programm-Status-Wort nicht übereinstimmen. In diesem Fall wird der Stack-Peinter um 1 erhöht und der momentane Inhalt des Instruction-Address-Registers, welcher auf das diesem Befehl folgende Byte hinweist, in den Return-Address-Stack geschrieben. Der vorherige Inhalt des Instruction-Address-Registers wird durch die effektive Adresse ersetzt (Sprung). Besteht Übereinstimmung zwischen dem v-Feld und dem Condition-Code, so wird der im Programm folgende Befehl ausgeführt.

In diesem Fall bleibt der Stack-Peinter unverändert (kein Sprung).

Indirekte Adressierung ist möglich.

Das v-Feld darf nicht auf 3₁₆ gesetzt werden, da dieser Operationscode für den Befehl ZBSR reserviert ist.

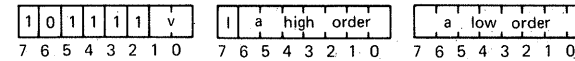
Processor Registers Affected SP

Condition Code Setting N/A

BRANCH TO SUBROUTINE ON CONDITION FALSE, ABSOLUTE (Absolute Addressing)

Mnemonic BSFA,v (*a)

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Dies ist ein bedingter 3-byte-Unterprogramm-Sprungbefehl. Der Sprung zum Unterprogramm wird nur dann durchgeführt, wenn die zwei Bits des v-Feldes mit dem momentanen Condition-Code im Programm-Status-Wort nicht übereinstimmen. In diesem Fall wird der Stack-Peinter um 1 erhöht und der momentane Inhalt des Instruction-Address-Registers, welcher auf das diesem Befehl folgende Byte hinweist, in den Return-Address-Stack geschrieben. Der vorherige Inhalt des Instruction-Address-Registers wird durch die effektive Adresse ersetzt (Sprung). Besteht Übereinstimmung zwischen dem v-Feld und dem Condition-Code, so wird der im Programm folgende Befehl ausgeführt.

In diesem Fall bleibt der Stack-Peinter unverändert (kein Sprung).

Indirekte Adressierung ist möglich.

Das v-Feld darf nicht auf 3₁₆ gesetzt werden, da dieser Operationscode für den Befehl BSXA reserviert ist.

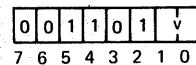
Processor Registers Affected SP

Condition Code Setting N/A

RETURN FROM SUBROUTINE AND ENABLE INTERRUPT, CONDITIONAL

Mnemonic RETE,v

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Dieser 1-byte-Befehl wird vorwiegend in Interruptroutinen verwendet, um einen bedingten Rücksprung zu jener Stelle im Programm zu bewirken, an der der Interrupt aufgetreten ist.

Wenn die zwei Bits des v-Feldes mit dem momentanen Condition-Code im Programm-Status-Wort übereinstimmen, bewirkt dieser Befehl folgendes: die Adresse in der Zeile des Return-Address-Stack, auf die der Stack-Pointer hinweist, wird in das Instruction-Address-Register übertragen (Sprung). Der Stack-Pointer wird nun um 1 vermindert.

Zusätzlich wird das Interrupt-Inhibit-Bit im PSU gelöscht. Dies erlaubt die Annahme weiterer Interrupts.

Besteht keine Übereinstimmung zwischen dem v-Feld und dem Condition-Code, erfolgt kein Rücksprung und als nächster Befehl wird der dem Return-Befehl folgende durchgeführt.

Setzt man das v-Feld auf 3₁₆, wird ein unbedingter Rücksprung durchgeführt.

Processor Registers Affected SP, II

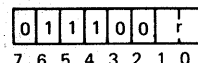
Condition Code Setting N/A

READ DATA

(Register Addressing)

Mnemonic REDD,r

Binary Code



Execution Time 2 cycles (6 clock periods)

Description

Durch diesen 1-byte-Input-Befehl wird ein Daten-Byte vom Daten-Bus in das spezifizierte Register r transferiert. Ein "High"-Signal am Daten-Bus wird als logische "1" interpretiert. Während der Ausführung dieses Befehls setzt der Prozessor das OPREQ-Signal "1". Gleichzeitig wird das M/ \overline{TO} -Signal auf \overline{TO} , das $\overline{R/W}$ -Signal auf \overline{R} , die D/ \overline{C} -Leitung auf D und die E/ \overline{NE} -Leitung auf \overline{NE} gesetzt.

Processor Registers Affected CC

Condition Code Setting

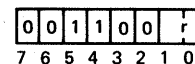
Register r	CC1	CC0
Positive	0	1
Zero	0	0
Negative	1	0

READ CONTROL

(Register Addressing)

Mnemonic REDC,r

Binary Code



Execution Time 2 cycles (6 clock periods)

Description

Durch diesen 1-byte-Input-Befehl wird ein Daten-Byte vom Daten-Bus in das spezifizierte Register r transferiert. Ein "High"-Signal am Daten-Bus wird als logische "1" interpretiert.

Während der Ausführung dieses Befehls setzt der Prozessor das OPREQ-Signal "1". Gleichzeitig wird das M/ \overline{TO} -Signal auf \overline{TO} , das $\overline{R/W}$ -Signal auf \overline{R} , die D/ \overline{C} -Leitung auf C und die E/ \overline{NE} -Leitung auf \overline{NE} gesetzt.

Processor Registers Affected CC

Condition Code Setting

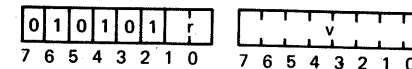
Register r	CC1	CC0
Positive	0	1
Zero	0	0
Negative	1	0

READ EXTENDED

(Immediate Addressing)

Mnemonic REDE,r v

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 2-byte-Input-Befehl wird ein Daten-Byte vom Daten-Bus in das Register r transferiert. Während der Ausführung dieses Befehls erscheint das zweite Byte (v-Byte) am Adreß-Bus. Ein "High"-Signal am Daten-Bus wird als logische "1" interpretiert. Während der Ausführung dieses Befehls wird das OPREQ-Signal "1" gesetzt. Gleichzeitig erscheint der Inhalt des zweiten Bytes auf dem Adreß-Bus. Während der Dauer des OPREQ-Signals wird das M/ \overline{TO} -Signal auf \overline{TO} , das $\overline{R/W}$ -Signal auf \overline{R} und das E/ \overline{NE} -Signal auf E gesetzt.

Processor Registers Affected CC

Condition Code Setting

Register r	CC1	CC0
Positive	0	1
Zero	0	0
Negative	1	0

WRITE DATA

(Register Addressing)

Mnemonic WRD,r**Binary Code**

1	1	1	1	0	0	r
---	---	---	---	---	---	---

7 6 5 4 3 2 1 0

Execution Time 2 cycles (6 clock periods)**Description**

Durch diesen 1-byte-Output-Befehl wird der Inhalt des Registers r auf den Daten-Bus gelegt. Eine logische "1" ergibt ein "High"-Signal. Während der Ausführung dieses Befehls wird das OPREQ-Signal "1" gesetzt. Gleichzeitig erscheinen die Daten am Daten-Bus. Das M/ \overline{T} O-Signal wird auf \overline{T} O, das \overline{R} /W-Signal auf W, das D/ \overline{C} -Signal auf D und das E/ \overline{NE} -Signal auf \overline{NE} gesetzt und ein "Write-Pulse" (WRP) wird ausgesendet.

Processor Registers Affected None**Condition Code Setting** N/A**WRITE CONTROL**

(Register Addressing)

Mnemonic WRTC,r**Binary Code**

1	0	1	1	0	0	r
---	---	---	---	---	---	---

7 6 5 4 3 2 1 0

Execution Time 2 cycles (6 clock periods)**Description**

Durch diesen 1-byte-Output-Befehl wird der Inhalt des Registers r auf den Daten-Bus gelegt. Eine logische "1" ergibt ein "High"-Signal. Während der Ausführung dieses Befehls wird das OPREQ-Signal "1" gesetzt. Gleichzeitig erscheinen die Daten am Daten-Bus. Das M/ \overline{T} O-Signal wird auf \overline{T} O, das \overline{R} /W-Signal auf W, das D/ \overline{C} -Signal auf \overline{C} und das E/ \overline{NE} -Signal auf \overline{NE} gesetzt und ein "Write-Pulse" (WRP) wird ausgesendet.

Processor Registers Affected None**Condition Code Setting** N/A**WRITE EXTENDED**

(Immediate Addressing)

Mnemonic WRTE,r v**Binary Code**

1	1	0	1	0	1	r	v	v	v	v	v	v
---	---	---	---	---	---	---	---	---	---	---	---	---

7 6 5 4 3 2 1 0 7 6 5 4 3 2 1 0

Execution Time 3 cycles (9 clock periods)**Description**

Durch diesen 2-byte-Output-Befehl wird der Inhalt des Registers r auf den Daten-Bus gelegt. Gleichzeitig steht das zweite Byte (v-Byte) dieser Instruktion auf dem Adreß-Bus zur Verfügung. Das v-Byte kann als Output-Adresse interpretiert werden. Eine logische "1" ergibt ein "High"-Signal. Während der Ausführung dieses Befehls wird das OPREQ-Signal "1" gesetzt. Gleichzeitig werden die Daten am Datenbus erscheinen. Das v-Byte wird auf den Adreß-Bus gelegt, die M/ \overline{T} O-Leitung wird auf \overline{T} O, das \overline{R} /W-Signal auf W, das E/ \overline{NE} -Signal auf E gesetzt und ein "Write-Pulse" (WRP) wird ausgesendet.

Processor Registers Affected None**Condition Code Setting** N/A**NO OPERATION****Mnemonic** NOP**Binary Code**

1	1	0	0	0	0	0	0
---	---	---	---	---	---	---	---

7 6 5 4 3 2 1 0

Execution Time 2 cycles (6 clock periods)**Description**

Dieser 1-byte-Befehl bewirkt weiter nichts. Es werden auch keine Register verändert. Die Ausführung dieses Befehls erfordert zwei Zyklen und kostet somit Zeit.

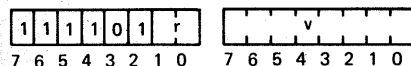
Processor Registers Affected None**Condition Code Setting** N/A

TEST UNDER MASK IMMEDIATE

(Immediate Addressing)

Mnemonic TMI, r v

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Durch diesen 2-byte-Befehl werden ausgewählte Bits des spezifizierten Registers r auf das Vorkommen einer 1 getestet. Bei der Ausführung dieses Befehls wird jedes Bit des v-Bytes auf das Vorkommen einer 1 geprüft. Enthält ein Bit des v-Bytes eine 1, so wird auch das korrespondierende Bit im Register r untersucht.
Der Condition-Code enthält das Ergebnis dieser Operation.
Enthält ein Bit im v-Byte eine 0, so wird das korrespondierende Bit im Register nicht untersucht.

Processor Registers Affected CC

Condition Code Setting

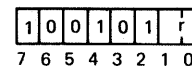
	CC1	CC0
All of the selected bits are 1s	0	0
Not all of the selected bits are 1s	1	0

DECIMAL ADJUST REGISTER

(Register Addressing)

Mnemonic DAR, r

Binary Code



Execution Time 3 cycles (9 clock periods)

Description

Dieser 1-byte-Befehl addiert unter bestimmten Bedingungen eine 10_{10} (A_{16}) (das Zweier-Komplement von -6 in einem 4-bit-Binärsystem) zu den vier höherwertigen und/oder den vier niedrigerwertigen Bits des spezifizierten Registers r.

Er dient zur Durchführung des 10-er-Übertrags bei BCD-Arithmetik in gepackter Form. Die nachfolgende Wahrheitstabelle zeigt die logische Operation. Die Art der Berechnung hängt vom Carry- und vom Interdigit-Carry-Bit im Programm-Status-Wort ab. Bei der Ausführung dieses Befehls bleiben das Carry und das Interdigit-Carry unverändert.

BCD-Arithmetik ist auf folgende Weise möglich:

- BCD-Addition:
1. Addiere 66_{16} zum ersten Summanden
 2. Addiere den ersten und zweiten Summanden
 3. DAR-Befehl mit der Summe

- BCD-Subtraktion:
1. Subtraktion des Minuenden vom Subtrahenden
 2. DAR-Befehl mit Differenz

Bei dieser Operation ist es wichtig, das Vorzeichen des Ergebnisses schon vor der Berechnung festzustellen, damit der Minuend und der Subtrahend richtig bestimmt werden können.

Carry	Interdigit Carry	Added to Register r
0	0	AA_{16}
0	1	$A0_{16}$
1	1	00_{16}
1	0	$0A_{16}$

Processor Registers Affected CC

Condition Code Setting

Der Condition-Code ist nach dieser Operation ohne Bedeutung.

HALT, ENTER WAIT STATE

Mnemonic HALT

Binary Code

0	1	0	0	0	0	0	0
---	---	---	---	---	---	---	---

7 6 5 4 3 2 1 0

Execution Time 2 cycles (6 clock periods)

Description

Durch diesen 1-byte-Befehl stellt der Prozessor die Abarbeitung des Programms ein. Das RUN/WAIT-Signal wird auf WAIT geschaltet.

Die einzige Möglichkeit um wieder auf RUN umzuschalten, ist das Auftreten eines Interrupts oder das Anlegen eines Signals auf die Reset-Leitung.

Processor Registers Affected None

Condition Code Setting N/A

Alphabetisches Inhaltsverzeichnis

HEX	OP	SEITE	HEX	OP	SEITE	HEX	OP	SEITE
8C	ADDA	319	FC	BDRA	339	38	BSTR	342
8D			FD			39		
8E			FE			3A		
8F			FF			3B		
84	ADDI	318	F8	BDRR	339	BF	BSXA	345
85			F9			9F	BXA	341
86			FA					
87			FB					
88	ADDR	318	DC	BIRA	338	EC	COMA	329
89			DD			ED		
8A			DE			EE		
8B			DF			EF		
80	ADDZ	317	D8	BIRR	338	E4	COMI	328
81			D9			E5		
82			DA			E6		
83			DB			E7		
4C	ANDA	323	5C	BRNA	340	E8	COMR	328
4D			5D			E9		
4E			5E			EA		
4F			5F			EB		
44	ANDI	322	58	BRNR	340	EO	COMZ	327
45			59			E1		
46			5A			E2		
47			5B			E3		
48	ANDR	322	BC	BSFA	343	75	CPSL	334
49			BD			74	CPSU	333
4A			BE					
4B			B8	BSFR	343			
41	ANDZ	321	B9			94	DAR	351
42			BA			95		
43			7C	BSNA	344	96		
			7D			97		
			7E					
9C	BCFA	337	7F					
9D			78	BSNR	344	2C	EORA	327
9E			79			2D		
			7A			2E		
98	BCFR	337	7B			2F		
99			3C	BSTA	342	24	EORI	326
9A			3D			25		
1C	BCTA	336	3E			26		
1D			3F			27		
1E						28	EORR	326
1F						29		
18	BCTR	336				2A		
19						2B		
1A								
1B								

Alphabetisches Inhaltsverzeichnis

HEX	OP	SEITE	HEX	OP	SEITE	HEX	OP	SEITE
20	EORZ	325	77	PPSL	333	A4	SUBI	320
21						A5		
22			76	PPSU	332	A6		
23						A7		
						A8	SUBR	320
						A9		
40	HALT	352	30	REDC	347	AA		
			31			AB		
			32			AO	SUBZ	319
			33			A1		
6C	IORA	325	70	REDD	346	A2		
6D			71			A3		
6E			72					
6F			73					
64	IORI	324	54	REDE	347	F4	TMI	350
65			55			F5		
66			56			F6		
67			58			F7		
68	IORR	36	14	RETC	345	B5	TPSL	335
69			15			B4	TPSU	334
6A			16					
6B			17			BO	WRTC	348
60	IORZ	323	34	RETE	346	B1		
61			35			B2		
62			36			B3		
63			37			FO	WRTD	348
			DO	RRL	329	F1		
			D1			F2		
OC	LODA	315	D2			F3		
OD			D3			D4	WRTE	349
OE			50	RRR	330	D5		
OF			51			D6		
04	LODI	314	52			D7		
05			53					
06								
07						9B	ZBRR	335
08	LODR	315	13	SPSL	332	BB	ZBSR	341
09			12	SPSU	331			
0A			CC	STRA	317			
0B			CD					
00	LODZ	314	CE					
01			CF					
02			C8	STRR	316			
03			C9					
93	LPSL	331	CA					
			CB					
92	LPSU	330	C1	STRZ	316			
			C2					
			C3					
			AC	SUBA	321			
CO	NOP	349	AD					
			AE					
			AF					

Befehlsvorrat des 2650 A

	MNE-MONIC	DESCRIPTION OF OPERATION	OP CODE R or CC				PSW BITS AFFECTED						BYTES	CYCLES	FOR-MAT*	NOTE	
			3	2	1	0	CC	IDC	C	OVF	SP	II					F
LOAD/STORE	LOD	Z Load register zero	03	02	01	—	•							1	2	Z	1
		I Load immediate	07	06	05	04	•							2	2	I	1
		R Load relative	0B	0A	09	08	•							2	3	R	1,6
		A Load absolute	0F	0E	0D	0C	•							3	4	A	6
	STR	Z Store register zero	C3	C2	C1	—	•							1	2	Z	1
	R Store relative	CB	CA	C9	C8	•							2	3	R	6	
	A Store absolute	CF	CE	CD	CC	•							3	4	A	6	
ARITHMETIC	ADD	Z Add to register zero w/wo carry	83	82	81	80	•	•	•	•				1	2	Z	1
		I Add immediate w/wo carry	87	86	85	84	•	•	•	•				2	2	I	1
		R Add relative w/wo carry	8B	8A	89	88	•	•	•	•				2	3	R	1,6
		A Add, absolute w/wo carry	8F	8E	8D	8C	•	•	•	•				3	4	A	1,6
	Z Subtract from register zero w/wo borrow	A3	A2	A1	A0	•	•	•	•					1	2	Z	1
	I Subtract immediate w/wo borrow	A7	A6	A5	A4	•	•	•	•					2	2	I	1
	R Subtract relative w/wo borrow	AB	AA	A9	A8	•	•	•	•					2	3	R	1,6
A Subtract absolute w/wo borrow	AF	AE	AD	AC	•	•	•	•					3	4	A	1,6	
DAR	Decimal adjust register	97	96	95	94	•							1	3	Z	1,10	
LOGICAL	AND	Z AND to register zero	43	42	41	—	•							1	2	Z	1
		I AND immediate	47	46	45	44	•							2	2	I	1
		R AND relative	4B	4A	49	48	•							2	3	R	1,6
		A AND absolute	4F	4E	4D	4C	•							3	4	A	1,6
	IOR	Z Inclusive-OR to register zero	63	62	61	60	•							1	2	Z	1
		I Inclusive-OR immediate	67	66	65	64	•							2	2	I	1
		R Inclusive-OR relative	6B	6A	69	68	•							2	3	R	1,6
		A Inclusive-OR absolute	6F	6E	6D	6C	•							3	4	A	1,6
	EOR	Z Exclusive-OR to register zero	23	22	21	20	•							1	2	Z	1
		I Exclusive-OR immediate	27	26	25	24	•							2	2	I	1
R Exclusive-OR relative		2B	2A	29	28	•							2	3	R	1,6	
A Exclusive-OR absolute		2F	2E	2D	2C	•							3	4	A	1,6	
ROTATE/COMPARE	COM	Z Compare to register zero arithmetic/logical	E3	E2	E1	E0	•							1	2	Z	2
		I Compare immediate arithmetic/logical	E7	E6	E5	E4	•							2	2	I	3
		R Compare relative arithmetic/logical	EB	EA	E9	E8	•							2	3	R	3,6
	A Compare absolute arithmetic/logical	EF	EE	ED	EC	•								3	4	A	3,6
	RRR	Rotate register w/wo carry	53	52	51	50	•	•	•	•				1	2	Z	1
RRL	Rotate register left w/wo carry	D3	D2	D1	D0	•	•	•	•				1	2	Z	1	
BRANCH	BCT	R Branch on condition true relative	1B	1A	19	18								2	3	R	7,8
		A Branch on condition true absolute	1F	1E	1D	1C								3	3	B	7,8
	BCF	R Branch on condition false relative	—	9A	99	98								2	3	R	7
		A Branch on condition false absolute	—	9E	9D	9C								3	3	B	7
	BRN	R Branch on register non-zero relative	5B	5A	59	58								2	3	R	7,8
		A Branch on register non-zero absolute	5F	5E	5D	5C								3	3	B	7,8
	BIR	R Branch on incrementing register relative	DB	DA	D9	D8								2	3	R	7,8
		A Branch on incrementing register absolute	DF	DE	DD	DC								3	3	B	7,8

MNE-MONIC	DESCRIPTION OF OPERATION	OP CODE R or CC	PSW BITS AFFECTED							BYTES	CYCLES	FOR- MAT*	NOTE			
			3	2	1	0	CC	IDC	C					OVF	SP	II
BRANCH (Cont'd)	BDR { R A	Branch on decrementing register relative	FB	FA	F9	F8							2	3	R	7,8
		Branch on decrementing register absolute	FF	FE	FD	FC								3	3	B
	ZBRR	Zero branch relative, unconditional	9B	—	—	—							2	3	ER	6
	BXA	Branch indexed absolute, unconditional	9F	—	—	—							3	3	EB	5,6
SUBROUTINE BRANCH/RETURN	BST { R A	Branch to subroutine on condition true, relative	3B	3A	39	38				•			2	3	R	7,8
		Branch to subroutine on condition true, absolute	3F	3E	3D	3C				•				3	3	B
	BSF { R A	Branch to subroutine on condition false, relative	—	BA	B9	B8				•			2	3	R	7
		Branch to subroutine on condition false, absolute	—	BE	BD	BC				•			3	3	B	7
	BSN { R A	Branch to subroutine on non-zero register, relative	7B	7A	79	78				•			2	3	R	7,8
		Branch to subroutine on non-zero register, absolute	7F	7E	7D	7C				•			3	3	B	7,8
	ZBSR	Zero branch to subroutine relative, unconditional	BB	—	—	—							2	3	ER	6
	BSXA	Branch to subroutine, indexed, absolute unconditional	BF	—	—	—							3	3	EB	5,6
	RET { C E	Return from subroutine, conditional	17	16	15	14				•			1	3	Z	8
		Return from subroutine and enable interrupt, conditional	37	36	35	34				•	•		1	3	Z	8
MISC. INPUT/OUTPUT	WRD	Write data	F3	F2	F1	F0						1	2	Z		
	REDD	Read data	73	72	71	70	•					1	2	Z	1	
	WRTC	Write control	B3	B2	B1	B0						1	2	Z		
	REDC	Read control	33	32	31	30	•					1	2	Z	1	
	WRTE	Write extended	D7	D6	D5	D4						2	3	I		
	REDE	Read extended	57	56	55	54	•					2	3	I	1	
MISC.	HALT	Halt, enter wait state	—	—	—	40						1	1	E		
	NOP	No operation	—	—	—	C0						1	1	E		
	TMI	Test under mask immediate	F7	F6	F5	F4	•					2	3	I	4	
PROGRAM STATUS	LPS { U L	Load program status, upper	92				•	•	•	•	•	•	1	2	E	
		Load program status, lower	93				•	•	•	•	•	•	1	2	E	9
	SPS { U L	Store program status, upper	12				•						1	2	E	1
		Store program status, lower	13				•						1	2	E	1
	CPS { U L	Clear program status, upper, masked	74							•	•	•	2	3	EI	
		Clear program status, lower, masked	75				•	•	•	•			2	3	EI	9
	PPS { U L	Preset program status, upper, masked	76							•	•	•	2	3	EI	
		Preset program status, lower, masked	77				•	•	•	•			2	3	EI	9
	TPS { U L	Test program status, upper, masked	B4				•						2	3	EI	4
		Test program status, lower, masked	B5				•						2	3	EI	4

Literatur

Bernstein, H.: TV-Computerspiele. Stuttgart 1980

Fischer, Othmar: Mikroprozessoren für Anfänger. Programmieren [des 2650] leicht und schnell erlernbar. Wien/München 1982.

Gäßler, Reinhard: Dem Mikrocomputer aufs Bit geschaut. Elo H.12/1978 bis H.8/1980. Als Sonderheft 3. Aufl. München 1980.

Hatzenbichler, Johann: Mikrocomputer-Programmierbeispiele mit dem Mikroprozessor 2650. München 1978.

Herchen, Hans-Alfred: Computereien. Heiteres aus der Welt der Computer und Elektronik. Frankfurt/M. 1983.

Holmes, P.: TV-Spielcomputer. Spielen mit dem Mikroprozessor [2650]. Gangel 1982.

Nührmann, Dieter: Schlüssel zum Mikrocomputer. München 1983.

Osborne, Adam: Einführung in die Mikrocomputer-Technik. 4. Auflage München 1982.

Sacht, Hans-Joachim: μ P-Programmierfibel für 2650/6502/6800/8080-85. Würzburg 1980.

Schnell, Gerhard/Hoyer, Konrad: Mikrocomputerfibel. 2. Auflage Braunschweig 1983.

VALVO: Mikroprozessoren, Mikrocomputer und Peripherieschaltungen (Datenbuch). Hamburg 1982.

VALVO: Integrierte Logikschaltungen (Datenbuch). Hamburg 1982.

VALVO/signetics: Mikroprozessor 2650 A (Handbuch). Hamburg 1980.

Sachverzeichnis

A

A (Anfang) 236
A (ask) 235
Abgleich (A/D-Wandler) 209
abisolieren 17
Abisolierzange 17
Ablaufsteuerung 120
Ableitwiderstand 181, 184
absolute Adressierung 310
abtrennen, elektronisch 132, 169
Adapter 90
Addition 97
ADFLOT 132, 141
ADMEM 103, 127, 130, 141, 153, 163, 174, 177
ADPER 103, 127, 130, 174, 177
ADREN 124, 132
Adresse 146
Adressierbereich 164
adressieren 118, 148
Adressierung 147, 150
- relative 250, 252
Adressierungsarten 308
Adreß-Bit 150
Adreßbus 121
Adreßdecoder 155, 181, 217
Adreßeingabe und -anzeige 68
ADVAL 141, 170
A/D-Wandler, s. auch Analog-Digital-Wandler 199
Akkumulator 118
Aktuator 212
ALU 111, 117
Alphabet 215
Ammoniumpersulphat 35

Ampelschaltung 179
analog 40
Analog-Digital-Wandler 68, 72, 181, 189, 190, 193, 199
Analogtechnik 40
Anode, gemeinsame 213
Anwenderprogramm 282
Anzeige 213
Anzeigeeinheit 217
Anzeigezyklus 219
Arbeitsregister 116, 240
Arbeitsspeicher 155
Arbeitstempo 11
ätzen 35
Ätznatron 34
Auffangregister 178, 219
Auffangspeicher 57
Auffrischung 10
Ausgang (TTL) 61, 179, 181
Ausgangsport 173, 179
Auslenkzeit 197
auslöten 89
Aussage 42
Aussagenlogik 42
Auto-RESET 128

B

B (Break-Point) 235
Bandzählwerk 243
Basismaterial 31
Basiszahl 93
Bauelemente (Montage) 15
BCD 214
BCD-Arithmetik 116

Bedienung (Tastatur) 234
Bedienungskomfort 189
Befehlsadreßregister 118
Befehlsdecoder 120
Befehlsformat 313
Befehlshalterregister 120
belichten 32
Benutzerroutine 252
beschichten (Fotolack) 32
Betriebsart, arithmetische 99
- logische 99
Betriebsprogramm, siehe auch Monitor 213
Bewegung 212
Bezugsspannung 200
Biegelehre 184
binär 41
Bit 59, 95
Block 236
bohren 32, 36
Bohrmaschine 36
Boole, G. 44
Break-Point 325
Brückenschaltung 212
Brummspannung 211
buffer 48
Bus 81, 179
Busadapter 90, 187
Busplatte 67, 81
Bussystem 81
- Bustransceiver 160, 180, 217
Byte 95

C

C (Clear Break-Point) 236
Carry-Bit 111, 117
CE 163, 180
CMD 235
Clear 60
Clock 56, 121, 128
Clock-Frequenz 248
Clock-Oszillator 125, 196
Compare-Bit 115
Condition-Code 115, 307

Computer 189, 190
Control-Taste 219, 224, 230, 232
CPU 67, 111
CS 152, 154, 156

D

D (Dump to Tape) 236
DAFLOT 102, 132, 188
Datei 236
Dateneingabe und -anzeige 68, 141
Datenaufzeichnung 242
Datenbus 95, 100, 120, 180, 199
Datentaste 219, 224, 230, 232, 236
Datenverarbeitung 67
Datenwort 59
DBUSEN 124, 132
D/C 124
Decoder 152, 169, 177
dynamisch 159
Dekrementierung 111
D-Flip-Flop 200
digital 40
Digitaltechnik 40
Digitalvoltmeter 190
Differenzierglied 146
DIL-Stecker 185
Diodenbuchse 135
Disjunktion 45
Division 99
Drahtbrücke 15
Drehwinkel 190
Drillbohrer 37
Druck 190
Dualzähler 193
Durchflußmenge 190
Durchlaßspannung 200
duty cycle 58
dynamisch 10

E

E (Enable) 153
E (Ende) 236

E (Examine) 236
EARAM 160
Eccles, W.H. 56
EI 220
Eigenleitung 158
Ein-/Ausgabeverwaltung 120
Eingang 181
Eingangskabel 209, 211
Eingangstrom 64, 181
Eingangswiderstand 191
Eingangsvariable 45
Einpreßwerkzeug 185
Einzel-Clock s. auch -takt 218
Einzelschrittsteuerung 129
Einzeltakteingabe 127, 129
Eisenoxiddkassette 243
Eisen-III-Chlorid 35
E/NE 124
Entkopplung 156
entlöten 22
Entlötlitze 23
entschichten 36
Entwickeln (Platinen) 34
EO 221
EPROM 158
E²PROM 160
ERROR 237
ERROR 2 236
EXAMINE 236, 243, 248
EXOR 44, 52
Experimentierplatte 26, 82

F

F (Fetch) 236
Fädeldraht 16
Fahrradventilschlauch 77
falsch 42
FAMOS 158
fan in 64
fan out 64, 217
Farbcode 17, 109
Federleiste 82
Festwertspeicher 155, 213
FET 191

Flachbandlitze 185
FLAG 98, 240, 244, 248
FLAG-Bit 115
Flanke (Signal) 53, 58
Flip-Flop 44ff.
Flußmittel 19
Fotolack 32
Frequenzzähler 244
FSR 200
Fühler 190
Funktionstafel 44

G

gA 213
Gatter 43
Gegenkopplung 202, 210
gK 213
Gleichspannungsverstärker 202
GND 61
GOTO 235
GS 221

H

H (High) 12, 41
Haarriß 79
HALT 171
hand-shake 129
Hardware 9
Hexadezimalsystem 95
Heyting, A. 44, 47
HF-Einstrahlung 212
HF-Litze 16
High-Efficiency-Anzeige 219
High-Fenster 242, 247
Hilbert, D. 44, 47
Hilfsspeicher 253
HOLD 171
Hysterese 54

I

IAR 118
Immediate-Adressierung 308
Impulsbetrieb 103
Index-Control-Bit 310
indirekte Adressierung 311
Inkrementierung 111
INTACK 123
Interdigit-Carry 116
Interrupt 121, 150
Interrupt-Inhibit-Bit 115, 123
Interruptlogik 120
INTREQ 122
Inverter 47, 50
Isoliernippel 77
-scheibe 77
I/O 121

J

Jordan, F. W. 56

K

Kabelbaum 90
-knoten 90
kalte Lötstellen 21, 89
Kassette 236
Kassetteninterface 68, 135, 154, 240
Kassettenlaufwerk 249
Kassettenrecorder 242
Kathode, gemeinsame 213
Kilo 148
Kilobyte 119
Klarpausspray 33
Knoten (Kabelbaum) 90
Kolophoniumlötendraht 19
Komplement 97
Konjunktion 43
Krokodilklemme 18, 209
Kühlblech 75
Kupfernadel 79

L

L (Low) 41ff.
Label 250
laden 132, 141, 169, 170, 220, 236, 240
Lastfaktor 64
Lasteinheit 65
Latch 57, 59, 199, 200
Laubsäge 26
Lautsprecher 179
LDR 212
LED 101, 179
Leibniz, G. W. 93
Leiterbahnunterbrecher 28
lesen (Daten) 171
- (Zahlen) 96
Leseoperation 191
load 121, 174
Licht 190
LIFO-Speicher 119
Litze 89, 109
Logik 41
Logikplan 43
logische Schaltung 40
löschen (EPROM) 158
Lötendraht 19
löten 18
LötKolben 18, 20
Lötack 19
Löt-nagel 24, 85, 89
Lötstelle, kalte 21
Low 12
LS 65

M

Magnetfeld 190
Maschinenzyklus 129, 147, 172, 218
Master Reset 60
Matrix 156
memory mapped I/O 174
Meßbereich 200, 203
Meßkabel 211
Mikrocomputer 68, 69
Mikroprozessor 10

M/ \overline{O} 123
MON 234
Monitor 161, 213, 214, 252
Mono-Flop 197
Montage (Bauelemente) 15
– (Leiterplatten) 25
– (Schiebeschalter) 105
Morgan, A. de 51
de Morgansches Gesetz 51
MOSFET 158
Multimeter, siehe auch
Vielfachinstrument 230
Multiplex-Betrieb 218
multiplexen 225
Multiplexer 117
Multiplikation 99
Musterprogramm 252

N

NAND 48, 51, 197
Natronlauge 34
NEXT 220, 235, 238
Negation 47
Netzkabel 71
Netzteil 69
Netztransformator 69, 71
Nibble 95, 146, 219
NICHT 47
NOP 146
NOR 49, 51, 197
NOR, wired 49
NOT 47
NTC-Widerstand 212

O

OAR 119
o. c. siehe Open-Collector
ODER 45, 51
 \overline{OE} 63, 154, 156, 163
Offset 203
Offsetabgleich 303, 210
Ohmmeter 30, 76, 88

OPACK 123, 129
Open-Collector-Ausgang 47, 61, 62,
101
Operandenadreibregister 119
Operationsverstärker 191, 193, 202,
210
OPREQ 121, 123
Organisation (Speicher) 150
Ortssender 212
Oszillator 125
Oszillograph 206, 244, 247
Overflow-Bit 98, 111, 116

P

Page 121, 149
Page-Bit 149, 150
PAUSE 122, 129, 170
PC, s. auch Programmzähler 118
Peano, G. 47
Pegel (H-, L-) 100
Peripherie 67, 103, 152, 179
Peripheriegerät 129
Pinzette 22
Platine 31
Port 67, 173, 174, 188
Portadresse 181, 217
Portbaustein 217
Portschaltung 199
Potentiometer 212
prellen 56
Prioritätsencoder 220, 232
Programm 67, 148, 250
Programmarchiv 243
programmieren 13
Programm-Status-Wort 112, 114, 118,
307
Programmzähler 118, 141, 146, 172,
234
– externer 220, 235
PROM 156, 158
Prozessorzyklus, s. auch Maschinen-
zyklus 121
Prüfkabel 209
PSL 115
PSU 115

PSW, s. Programm-Status-Wort
Puffer 48, 179, 224
Pulsbreite 198
Punkt (in der Anzeige) 237

Q

Quittungsverfahren 129

R

RAM 67, 154, 159, 188
RAM-Bereich 234
read 121, 124
Rechenregeln 97
Rechenwerk 111
Rechtecksignal 53
Referenzspannung 200, 202
Register 59, 199
Registeradressierung 308
Register-Bank-Select-Bit 116
Registerinhalt 252
Regelautomatik 242
Regelschwingung 242
relative Adressierung 309
RESET 121, 128, 146, 150, 170, 179,
234, 238
ROM 67, 155, 156
Röntgenstrahlung 158
Rotationsbefehl 111
Routine 213
R-/2R-Widerstandsnetzwerk 191
rücksetzen 55
RUN 235
RUN/WAIT 122
R/W 124

S

Salva, F. 100
Schalt draht 16
Schalter (Montage) 133

Schieberegister 60
Schiebeschalter 105
Schleife 119
schleifen (Bohrer) 38
Schmitt, O. H. 53
Schmitt-Trigger 48, 53
Schnittmenge 44
Schreib-/Lesespeicher, s. RAM 159
Schreiboperation 121
Schutzmaßnahmen 38
Sedezimalsystem 95
Segment 214
Seifenlauge 36
SENSE 242, 247
SENSE-Bit 115
SENSE-Eingang 115
Sensor 190, 212
setzen 55
Sicherung 72
Sicherungshalter 72
Siebensegmentanzeige 161, 164, 213,
218
Signal 103
Signal darstellung 100
Singlestep 218
Skalenendbereich 211
Skalenendwert 200
Software 9, 214, 219
Spannungskomparator 193
Spannungspegel 42, 44
Spannungsregelung 69
Spannungsregler 73, 75, 79
Spannungsteiler 191
Spannungsversorgung 120
Spannungsverstärkung 202, 210
Speicher 67, 132, 141, 188, 220, 240
Speicheradresse 148
Speicherbaustein 149, 152
Speicherbereich 148
Speicherfeld 156
Speicherinhalt 240
speichern 55
Speicherraum 151
Speicherplatz 141, 146
Speicherstelle, s. auch Speicherplatz
118, 174
Speichertyp 154

Speicherzelle 56
Sprung 119
S/R 179, 180
Stack 119
Stackpointer 115, 119
statisch 10, 159
STEP 127, 130
Stellenkomplement 98
Stellenwert 93
Stiftleiste 82
STOP 129
store 121, 174
Störimpuls 65, 225
Störspannung 203
STROBE 219, 221, 224, 232
Subtraktion 97

T

Taktgenerator 10, 193
Taktzeit 130, 172
Tastatur 68, 161, 164, 213, 219
Tastverhältnis 58, 80
Temperatur 190, 200
Thyristor 73
Thyristorschaltenteil 72
Timing 11
Tonbandkassette 236, 240
Transceiver 160, 179, 180
Transformator 69, 71, 74, 75, 77
transparent (Latch) 57, 199, 200
Treppenspannung 200
Trimmerpotentiometer 206
Tri-State-Ausgang 81, 154
Tri-State-Bustransceiver 160
Tri-State-Logik 62
Tri-State-Puffer 179, 199
TTL 40, 61
TTL-Ausgang 43, 65
TTL-Eingang 43, 64
Tunneleffekt 158
 t_w 198

U

Überlastung 218
Übertragungsfehler 249
uL 65
UND 43, 50
unity Load 65
Universalport 173, 179, 181
unterbrechen (Leiterbahnen) 27, 29, 30
Unterprogramm 213
UV-Strahlung 32, 158

V

VCC 61, 79
Verdrahtung 15, 30
verdrillen 18
Vergleichsoperation 115
verzinnen 19, 31
Vielfachmeßinstrument, s. auch
Multimeter 30, 88, 206, 209, 247
Vorverstärker 200
Vorwiderstand 101
 V_u 202, 203

W

Wahrheitstafel 44
WAIT 122, 129, 132, 163, 170, 238
Wasserstoffperoxid 35
WE 161
Widerstand, induktiver 103
Widerstandsbrücke 212
Widerstandsnetzwerk 191, 200
Wired NOR 49
With/Without Carry-Bit 116
Wort 59, 150
wortorientiert 156
write 121, 124
WRITE 128, 174
WRITE/READ 102
WRP 124

Z

Z (Tri-State-Zustand) 64
Zahlensystem 93, 96
Zähler 58
Zahnbürste 28
Zeitfehler 249
Ziffer 215
Zinn absaugen 21
Zinnabsauger 22
Zweierkomplement 276
Zweierkomplementzahl 115
zweiwertig 42