

Das 64'er EPROM-Programmiergerät

Werfen Sie einen Blick hinter die Kulissen der EPROM-Programmierung! Wir zeigen Ihnen, wie Sie sich mit wenig Aufwand ein eigenes EPROM-Programmiergerät bauen können.

Unser EPROM-Programmiergerät für den C 64 soll preiswert, zuverlässig, einfach zu bauen und übersichtlich aufgebaut sein. Zudem soll es verschiedene EPROMs, vor allem aber die gängigsten Typen, programmieren können. Dazu soll zunächst auf die Problemstellung, die der Bau eines EPROM-Programmiergerätes aufwirft, eingegangen werden. Daraus wird dann ein Konzept für das 64'er EPROM-Programmiergerät entwickelt und anschließend in eine konkrete Schaltung umgewandelt. Bevor aber die technischen Details beschrieben werden, ist es lohnend, einige Grundbegriffe, die in diesem Zusammenhang stehen, zu erläutern.

Datenbus: Der Datenbus besteht aus acht Leitungen, auf denen ein komplettes Byte (8 Bit) zwischen dem Computer und dem Eprommer in beiden Richtungen (bidirektional) übertragen werden kann. Bezeichnung: D0 bis D7.

Adreßbus: Bis zu 16 Leitungen, auf denen der Computer dem EPROM signalisiert, welches Datenbyte gelesen oder geschrieben werden soll.

Kontrollbus: Diese Leitungen steuern am EPROM die Schreib-, Lese- und Sperrfunktion (CS, CE, PGM, Tri State).

Spannungsversorgung: Jedes EPROM braucht zum Betrieb eine Spannung von 5 Volt und einen Masseanschluß (U_B und GND). Außerdem wird zum Programmieren eine Spannung von 12,5 bis 25 Volt (U_P) benötigt.

TTL-Pegel: Man versteht darunter logische Zustände, die durch bestimmte Spannungspegel dargestellt werden. Logisch »0« wird durch eine Spannung von 0 Volt, logisch »1« durch eine Spannung von 5 Volt dargestellt. Man kann Bausteine, die mit TTL-(Transistor-Transistor-Logik-)Pegeln arbeiten, ohne auf Spannungen zu achten, nur nach

logischen Gesichtspunkten miteinander verschalten.

Tri State: Im Tri State-Zustand schaltet ein Baustein alle Ein- und Ausgänge in einen hochohmigen Zustand. Dadurch wird der Baustein für den Bus praktisch »unsichtbar«.

Auf die Funktion eines EPROMs braucht an dieser Stelle nicht mehr ausführlich eingegangen zu werden, denn darüber haben wir bereits im 64'er, Ausgabe 8/85, berichtet. Außerdem wird darauf in dieser Ausgabe in dem Artikel »Modul-Generator« eingegangen. Deshalb nur kurz folgende Zusammenfassung: Ein EPROM ist ein Erasable Programmable Read Only Memory, also ein Festwertspeicher, der programmiert und auch wieder gelöscht werden kann. Zum Löschen benötigt man UV-Licht, wobei die Strahlung einer preiswerten Höhensonne vollkommen ausreicht. Um programmierte EPROMs auf dem C 64 zu verwenden, benötigt man geeignete Steckplatinen (Module) für den Expansion-Port (siehe auch 64'er, Ausgabe 10/85) oder einen Umsetzsockel für Betriebssystem oder Basic-ROM. Doch nun zur eigentlichen Problematik eines EPROM-Programmiergerätes. Betrachten wir die Vor- und Nachteile verschiedener Methoden der Realisierung.

Die Versorgungsspannung: Zur Versorgung des EPROMs und der Logikbausteine des EPROM-Programmiergerätes benötigt man +5 Volt und Masse (U_B und GND). Beide Anschlüsse stehen sowohl am User-Port als auch am Expansion-Port zur

Verfügung. Man braucht nur darauf zu achten, daß die Schaltung nicht mehr als 300 mA Strom verbraucht, da sonst die Spannungsversorgung des ganzen Computers zusammenbricht.

Die Programmierspannung: Um Daten in ein EPROM zu programmieren, benötigt man einen Spannungsimpuls für die Zeitdauer von 50 ms. Die Spannung beträgt dabei je nach verwendetem EPROM-Typ 24, 21 oder 12,5 Volt. Da am C 64 aber keine dieser Spannungen vorliegt, muß diese durch ein paar Tricks erzeugt werden. Dabei gibt es mehrere Möglichkeiten, von denen wir nur die drei praktikabelsten betrachten wollen. Am einfachsten ist es, ein externes Netzteil anzuschließen, das mindestens 24 Volt liefert. Alle anderen Spannungen könnte man daraus problemlos gewinnen. Diese Lösung würde aber die Anwendung des EPROM-Programmiergerätes etwas umständlich machen, und Kabel liegen auf dem Computertisch meistens sowieso schon genug herum. Zudem ist diese Lösung auch nicht unbedingt die preiswerteste. Verwendet man die am Computer zur Verfügung stehenden Spannungen, bieten sich drei weitere Methoden an. Nimmt man einen Transformator, kann man mit einem Trick arbeiten.

Der Trick mit dem Trafo

Normalerweise wird ein Transformator dazu verwendet, um aus hohen Spannungen niedrige zu erzeugen (beispielsweise bei einem Netzgerät). Man kann den Effekt aber auch umdrehen und so aus niedriger Spannung eine oder mehrere höhere erzeugen (Zündspule). Voraussetzung zum Betrieb eines Transformators ist immer eine Wechselspannung. An Pin 10 und 11 des User-Port steht diese Wechselspannung zur Verfügung.

Eine weitere Methode, die Kaskade (Bild 1), findet auch in der Hochspannungstechnik Verwendung. Das Verfahren beruht darauf, daß man Kondensatoren parallel auflädt und sie dann in Serie entlädt. So kann man Spannungsimpulse gewinnen, die eine Ausgangsspan-

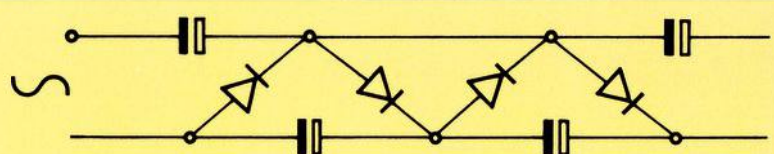


Bild 1. Die Schaltung einer Kaskade

nung nach der Anzahl der Kondensatoren vervielfältigen. Verwendet man zum Betreiben der Kaskade Wechselstrom, so kann man am Ausgang eine kontinuierlich vervielfältigte Spannung entnehmen. Die letzte Möglichkeit besteht darin, einen fertigen DC-/DC-Wandler zu verwenden. Diese Methode ist zwar die beste, aber normalerweise auch eine der teuersten.

Der Adreßbus: Jedes Byte, das in einem EPROM gespeichert oder aus diesem gelesen werden soll, muß an den Adreßeingängen des EPROMs angesprochen werden. Dazu hat jedes EPROM die Eingänge A0 bis AX, je nach der Kapazität des EPROMs. An diesen Eingängen legt man zum Programmieren binär codiert (mit TTL-Pegeln) die Nummer des Bytes an, welches man im EPROM ansprechen will. Zum Auslesen reicht es, wenn man das EPROM direkt an den Systembus anschließt (was man bei Steckmodulen am Expansion-Port auch macht). Zum Programmieren muß man die selektierte Adresse aber für mindestens 50 ms (Dauer des Programmierimpulses) stabil halten. Dazu ist es unumgänglich, die Adresse zwischenspeichern. Auch hier gibt es wieder mehrere Möglichkeiten.

Die erste Möglichkeit besteht in der Verwendung eines Binärzählers. Binärzähler haben einen Impulseingang und mehrere Zählgänge. Jedesmal, wenn am Eingang eine Impulsflanke auftritt, wird die an den Ausgängen durch TTL-Pegel dargestellte Zahl um eins erhöht. Man kann durch Koppeln von Binärzählern jede beliebige Zahl (als Spannungspegel) realisieren. Will man, wie wir nicht nur zählen, sondern auch bestimmte Zahlen darstellen, so benötigt man unbedingt einen Binärzähler mit sogenanntem »Clear«-Eingang. Gibt man einen Impuls auf diesen Eingang, wird der Zähler auf Null gestellt und man kann durch eine entsprechende Zahl von Impulsen am Zählengang jede beliebige Zahl (Adresse) an den Ausgängen herstellen. Der Vorteil dieser Anwendung ist, daß man nur drei Steuerleitungen zur Darstellung beliebiger Adressen braucht. Der Nachteil besteht darin, daß die gewünschten Adressen Impuls für Impuls hochgezählt werden müssen. Da am User-Port des C 64 nur eine begrenzte Anzahl von Leitungen zur Verfügung steht, eignet sich diese Methode hauptsächlich für Programmiergeräte, die am User-Port betrieben werden.

MODE SELECTION							
Mode	Pins	CE (20)	OE (22)	PGM (27)	V _{PP} (1)	V _{CC} (28)	Outputs (11-13, 15-19)
Read		L	L	H	V _{CC}	V _{CC}	D _{OUT}
Standby		H	X	X	V _{CC}	V _{CC}	High Z
Program		L	X	L	V _{PP}	V _{CC}	D _{IN}
Program Verify		L	L	H	V _{PP}	V _{CC}	D _{OUT}
Program Inhibit		H	X	X	V _{PP}	V _{CC}	High Z

X kann sowohl L als auch H sein

Bild 2. Zusammenhang zwischen Betriebsart und Zustand der Steuerleitungen

Das ganze kann man auch mit Registern machen. Ein Register ist ein Speicher, in dessen Eingänge man Daten schreiben kann und an dessen Ausgängen die Daten so lange präsent bleiben, bis sie gelöscht oder neu geschrieben werden. Man kann somit beliebige Zahlen direkt zwischenspeichern. Der Vorteil liegt in der durch die Datenübergabe direkten Adressierung. Der Nachteil besteht darin, daß man mindestens fünf, normalerweise aber zehn Steuerleitungen braucht. Für den User-Port ist diese Möglichkeit daher ungeeignet, für den Expansion-Port aber geradezu prädestiniert.

Statt der Register oder dem Binärzähler kann man aber auch einen Input/Output-Baustein verwenden. Da diese intelligenten, hochintegrierten Bausteine in der letzten Zeit erheblich im Preis gesunken sind, sollte man diese Methode nicht gleich beiseite legen. Mit einem 6821 Baustein hat man dann bis zu 20 Leitungen zur Verfügung, die bidirektional arbeiten und frei programmierbar sind. Man kann zudem die dort gespeicherte Adresse zu Kontrollzwecken wieder auslesen.

Der Datenbus: Er dient dem Austausch von Daten zwischen dem EPROM und dem Mikroprozessor 6510 in beiden Richtungen, er muß also bidirektional arbeiten. Das erscheint sinnvoll, denn wir wollen unser EPROM ja programmieren, auslesen und verifizieren. Zum Programmieren müssen die Daten, ebenso wie die Adressen, auch mindestens 50 ms stabil stehen. Das macht den direkten Anschluß an den Systembus des C 64 aber unmöglich, denn der C 64 kann eine Adresse nicht so lange stabil halten. Aber auch hier gibt es mehrere Möglichkeiten, das Problem zu lösen — betrachten wir zwei davon.

Zunächst kann man Register in Verbindung mit einem Bustreiber verwenden. Wie schon beim Adreßbus erläutert, kann man Register verwenden, um ein zu programmierendes Byte dem EPROM zur Verfü-

gung zu stellen. Da das Register jedoch nur in eine Richtung arbeitet, kann man die Datenleitungen des EPROM mit Hilfe eines Tri State-Bustreibers in der anderen Richtung (das heißt, vom EPROM zur CPU) direkt mit dem Systembus verbinden. Diese Methode ist aber nur am Expansion-Port möglich. Wesentlich flexibler ist dagegen ein Konzept, das einen Input/Output-Baustein verwendet. Am User-Port kann man für den bidirektionalen Datenaustausch den eingebauten I/O-Baustein 6526, dessen acht Datenleitungen herausgeführt sind, verwenden. Am Expansion-Port muß man einen I/O-Baustein installieren, von dem man ein 8 Bit Register zum Datenaustausch verwenden kann. Die restlichen Leitungen könnte man zum Beispiel zum Steuern des Kontroll-Busses oder niederwertigen Adreß-Byte verwenden.

Die Kontroll-Leitungen: Dieser Bus dient zur Steuerung des EPROM-Zustandes. Man kann damit das EPROM auf Programmieren, Lesen oder Stand-By schalten. Zudem dient dieser Bus dazu, die Programmiervoltage ein- und auszuschalten. Für den Aufbau des EPROM-Programmiergerätes sind die Signale des Kontroll-Bus besonders wichtig, im Bild 2 sind deshalb die Zustände der Steuerleitungen in Abhängigkeit von der Betriebsart beispielsweise bei einem 2764 EPROM abgebildet.

OE (Output Enable): Eine logische »0« an diesem Eingang des EPROM ermöglicht in Verbindung mit den Signalen CE und PGM ein Lesen des EPROM-Inhaltes. Während des Programmierzyklus ist der Zustand dieser Leitung ohne Bedeutung (kann 0 oder 1 sein).

CE (Chip Enable): Ist diese Leitung logisch »1«, so befindet sich das EPROM im Stand-By-Modus, das heißt, alle Ein- und Ausgänge werden in einen hochohmigen Zustand geschaltet. Dies ermöglicht den direkten Anschluß des EPROM an den Systembus. Eine logische »0« er-

2532	2732A	2764A	27128A	27256	27512	PIN	PIN	27512	27256	27128A	2764A	2732A	2532
		V _{PP}	V _{PP}	V _{PP}	A ₁₅	1	28	V _{CC}	V _{CC}	V _{CC}	V _{CC}		
		A ₁₂	A ₁₂	A ₁₂	A ₁₂	2	27	A ₁₄	A ₁₄	PGM	PGM		
A ₇	A ₇	A ₇	A ₇	A ₇	A ₇	3	26	A ₁₃	A ₁₃	A ₁₃	NC	V _{CC}	V _{CC}
A ₆	A ₆	A ₆	A ₆	A ₆	A ₆	4	25	A ₈	A ₈	A ₈	A ₈	A ₈	A ₈
A ₅	A ₅	A ₅	A ₅	A ₅	A ₅	5	24	A ₉	A ₉	A ₉	A ₉	A ₉	A ₉
A ₄	A ₄	A ₄	A ₄	A ₄	A ₄	6	23	A ₁₁	A ₁₁	A ₁₁	A ₁₁	A ₁₁	V _{PP}
A ₃	A ₃	A ₃	A ₃	A ₃	A ₃	7	22	OE/V _{PP}	OE	OE	OE	OE/V _{PP}	CE
A ₂	A ₂	A ₂	A ₂	A ₂	A ₂	8	21	A ₁₀	A ₁₀	A ₁₀	A ₁₀	A ₁₀	A ₁₀
A ₁	A ₁	A ₁	A ₁	A ₁	A ₁	9	20	CE	CE	CE	CE	CE	A ₁₁
A ₀	A ₀	A ₀	A ₀	A ₀	A ₀	10	19	O ₇	O ₇	O ₇	O ₇	O ₇	O ₇
O ₀	O ₀	O ₀	O ₀	O ₀	O ₀	11	18	O ₆	O ₆	O ₆	O ₆	O ₆	O ₆
O ₁	O ₁	O ₁	O ₁	O ₁	O ₁	12	17	O ₅	O ₅	O ₅	O ₅	O ₅	O ₅
O ₂	O ₂	O ₂	O ₂	O ₂	O ₂	13	16	O ₄	O ₄	O ₄	O ₄	O ₄	O ₄
GND	GND	GND	GND	GND	GND	14	15	O ₃	O ₃	O ₃	O ₃	O ₃	O ₃

Bild 3. Die Anschlußbelegungen der verwendeten EPROM-Typen

möglicht den Zugriff sowohl im Programmier- als auch im Lesemodus.

PGM (Programm): Dieser Eingang muß während des Programmiermodus logisch »0« sein. Will man das EPROM auslesen oder bei anliegender Programmiervspannung verifizieren, so muß man diesen Eingang auf logisch »1« schalten.

U_p (Programmiervspannung): An diesen Eingang müssen während des Programmierens 25 Volt Programmiervspannung anliegen. Betreibt man das EPROM in einem Computer (als fertig programmiertes ROM), so muß an diesen Eingang eine Spannung von 5 Volt angelegt werden.

U_b (Betriebsspannung): An diesem Pin des EPROM muß die Betriebsspannung von 5 Volt (105 mA während des Betriebs, 21 mA im Stand-By-Modus) angelegt werden.

GND (Ground): Masseanschluß des EPROM (Betriebs Erde).

Der Programmiervorgang (am Beispiel eines 2764)

Nach dem Löschen, beziehungsweise bei einem neuen EPROM, haben alle Bits den Wert »1«. Man kann also testen, ob ein EPROM vollständig gelöscht ist, indem man alle verfügbaren Bytes mit hexadezimal »FF« vergleicht. Durch das Programmieren wird später nichts anderes gemacht, als die entsprechenden Bits auf »0« zu setzen. Der Programmierzklus wird gestartet, indem man die Programmiervspannung an U_p anlegt. Die Adresse des zu programmierenden Bytes wird an den Adreß-, die Daten an den Dateneingängen in Form von TTL-Pegeln angelegt. Das EPROM muß dabei durch den CE-Eingang selektiert sein (logisch 0). Dann wird am PGM-Eingang ein Impuls (TTL logisch 0)

von 50 ms gegeben. Der Impuls darf auf keinen Fall länger als 55 ms sein, da sonst das EPROM zerstört wird. Die Länge des Impulses kann man durch Software steuern, sollte sie aber auf Seiten der Hardware unbedingt durch den Einsatz eines Monoflops absichern. Nach diesem Vorgang ist das adressierte Byte mit der am Datenbus angelegten Information fest programmiert. Nach der vollständigen Programmierung eines EPROMs sollte man dieses vor unbeabsichtigtem Löschen durch Aufkleben einer Silberfolie auf das EPROM-Fenster schützen (auch Tageslicht enthält UV-Strahlung).

Die Frage der EPROM-Typen

Es gibt EPROM-Programmierer, die nahezu jeden EPROM-Typ programmieren können. Da unser EPROM-Brenner jedoch für den Einsatz am C 64 gedacht ist, genügt es, wenn er die EPROM-Typen programmieren und lesen kann, die an diesem Computer am häufigsten Anwendung finden. Das hat den Vorteil, daß man den EPROM-Programmierer kostengünstig herstellen kann. Meistens benötigt man EPROMs, die das Betriebssystem ersetzen, statt des Basic-ROMs eingesetzt werden, oder als Steckmodul im Bereich \$8000 bis \$A000 eingeblendet werden. Da alle diese Bereiche 8 KByte Speicher umfassen (wie auch die Betriebssystem-ROMs der 1541 Floppy), ist dafür vor allem der Typ 2764 geeignet, der in letzter Zeit auch noch sehr billig geworden ist. Für größere Anwendungen (lange Programme, 16 KByte Steckmodule, Einsatz von Umschaltmodulen gibt es bis 128 KByte) kommt noch der Typ 27128 (16 KByte) oder 27256 (32 KByte) in Frage. Zum

Austausch des Zeichensatzes kann man dann noch 4 KByte EPROMs vom 2532 oder 2732 verwenden. Der Einsatz weiterer Typen ist im Regelfall nicht nötig und auch wirtschaftlich unsinnig, da vor allem die 27XX-Reihe zur Zeit fast überall sehr preiswert gehandelt wird. Bislang sind wir noch nie auf ein Problem gestoßen, das sich nicht mit den oben genannten EPROM-Typen (Bild 3) lösen ließe.

Das Konzept des 64'er EPROMmers

Um eine größtmögliche Flexibilität und Bedienerfreundlichkeit zu gewährleisten, wurde der 64'er EPROMmer zum Anschluß an den Expansion-Port entwickelt (Bild 4). Der Daten und Adreßverkehr, sowie die Steuerung des Kontrollbusses erfolgt über zwei I/O-Bausteine vom Typ 6821. Diese werden im C 64 im Adreßbereich \$DF00—\$DF03 (Baustein Nummer 1) und \$DF04—\$DF07 (Baustein Nummer 2) decodiert. Beide Bausteine sind parallel am Datenbus (D0 bis D7) und an den Adreßleitungen A0 und A1 des C 64 angeschlossen. Die Decodiereingänge CS0/CS1/CS2 der Bausteine sind folgendermaßen verschaltet: CS0 sind bei Baustein Nummer 1 direkt, bei Baustein Nummer 2 über einen Inverter 7404 mit der Adreßleitung A2 verbunden. CS1 ist bei beiden Bausteinen mit der Adreßleitung A3 und CS2 mit der Decodierleitung IO2 des Computers verbunden. Die IRQ-Ausgänge der beiden 6821 bleiben bei dieser Anwendung, da sie nicht gebraucht werden, unverschaltet. Weiterhin sind die Leitungen φ2 (Systemtakt), R/W (lesen/schreiben), Reset, +5 Volt und GND des C 64 mit beiden 6821 zur Steuerung des Datentau-

sches verbunden. Der I/O Port A des 6821 Nummer 1 übernimmt den Datenverkehr zwischen EPROM und Computer. Da die Datenleitungen aller verwendeten EPROMs gleich sind, können diese Verbindungen fest verdrahtet werden. Port B des gleichen IC steuert die Kontroll-Logik zum Lesen oder Programmieren der verschiedenen EPROM-Typen. Der 6821 Baustein Nummer 2 steuert den Adreßbus mit beiden Ports. Dadurch wird die Bearbeitung von EPROMs bis zu 64 KByte ermöglicht. Der größte Teil dieser Leitungen kann auch fest mit dem Texttoolsockel (Nullkraftsockel), der der Aufnahme der EPROMs dient, verbunden sein.

Dies sind die Leitungen A0 bis A10, die anderen Adreßleitungen sind bei jedem EPROM-Typ spezifisch und müssen über den Kontrollbus gesteuert werden. Bevor nun der Kontrollbus beschrieben wird, der die aufwendigste Verschaltung des EPROM-Programmiers darstellt, muß das Problem der Spannungsvorsorgung geklärt sein. Die stabilisierte Versorgungsspannung von 5 Volt für die gesamte Logik ist am Expansion-Port verfügbar. Die Programmierspannung wird aus einem DC-/DC-Wandler der Firma TDK (CE 0072) gewonnen. Dieser preiswerte Wandler liefert am Ausgang eine Gleichspannung von 31 Volt. Sie wird durch einen Kondensator von

100 μ F geglättet und über ein Transistor Zehner-Dioden-Glied auf 27 Volt vorgeregelt. Die endgültige Regelung auf die erforderlichen Programmierspannungen erfolgt nach nochmaliger Glättung durch einen Kondensator von 10 μ F direkt am EPROM durch Zehner-Dioden. Die Spannungen für die verschiedenen EPROM-Typen werden über den Kontrollbus geschaltet. Der Kontrollbus hat somit folgende Aufgaben:

1. Die Verschaltung des Adreßbusses (A11 bis A15) für die verschiedenen EPROM-Typen.
2. Die Steuerung der verschiedenen Programmierspannungen.
3. Die Bedienung der EPROM-Eingänge der verschiedenen EPROM-Typen zum Lesen oder Programmieren.
4. Die Erzeugung des Programmier-Impulses von 50 ms.
5. Die Bereitstellung der Versorgungsspannung für 24 Pin-EPROMs an Pin 26 des Texttoolsockels.

Diese Aufgaben werden mit Hilfe von Multiplexern und Transistoren (siehe Schaltpläne Bild 4 und Bild 5) erfüllt. Die Logik wurde dabei so geschaltet, daß für jeden EPROM-Typ eine Zahlenkombination in Port B geschrieben wird und das Umschalten zwischen Lesen und Schreiben über ein Bit gesteuert werden kann. Der verwendete 74LS139 Multiplexer ist mit seinen Eingängen mit Leitung B0 und B1 des 6821 Nummer 1 verbunden. So kann man mit einer Zahl zwischen Null und Vier eine bestimmte Programmierspannung auswählen. Der Enable-Eingang des Multiplexers, der die Auswahl auf die Ausgänge schaltet, wird von einem Monoflop 74LS123 gesteuert.

Bild 4. Der Blockschaltplan des 64'er EPROM-Programmiergeräts

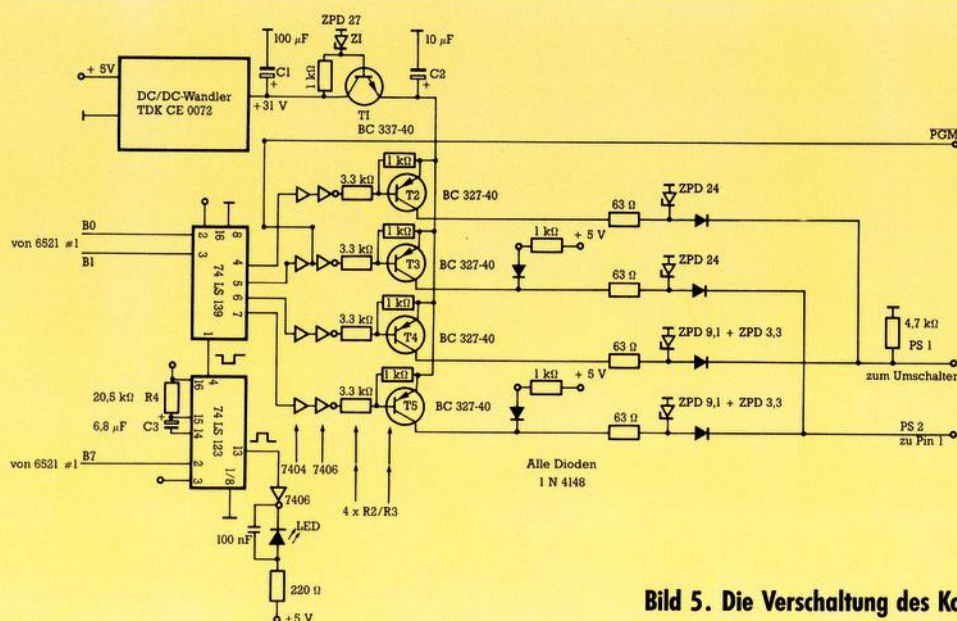
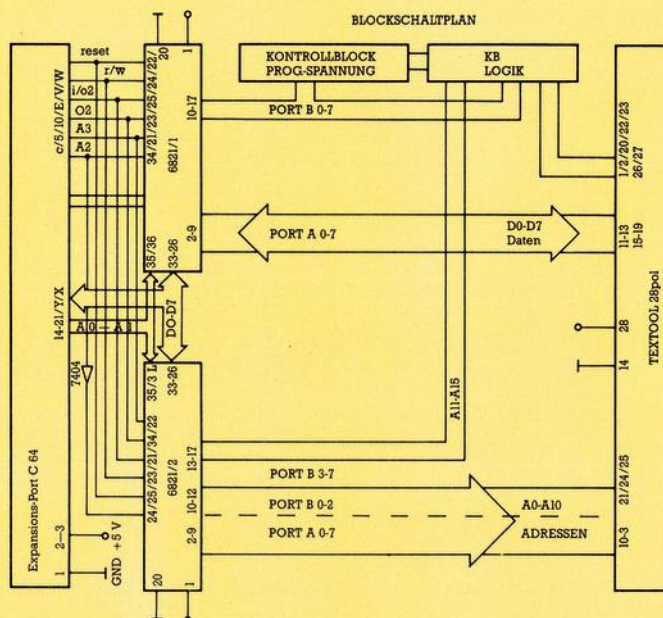
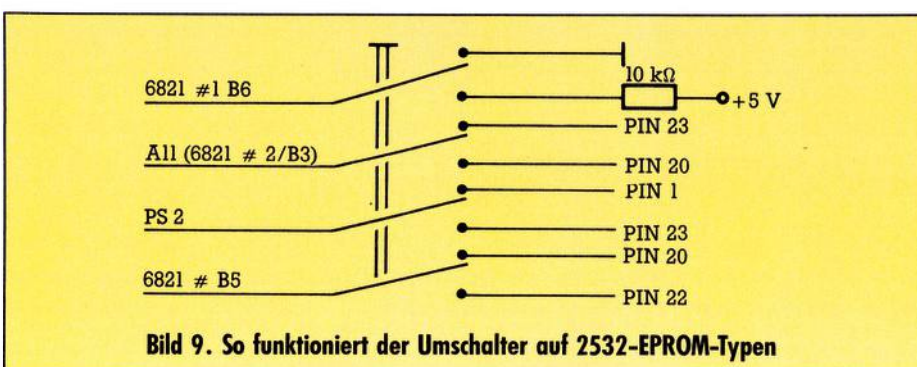
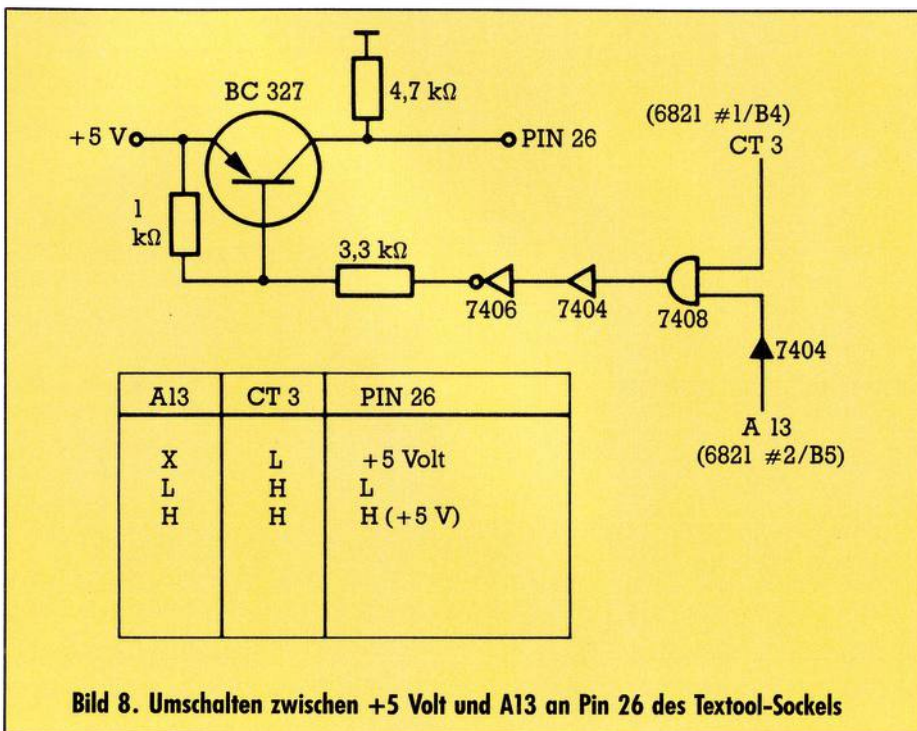
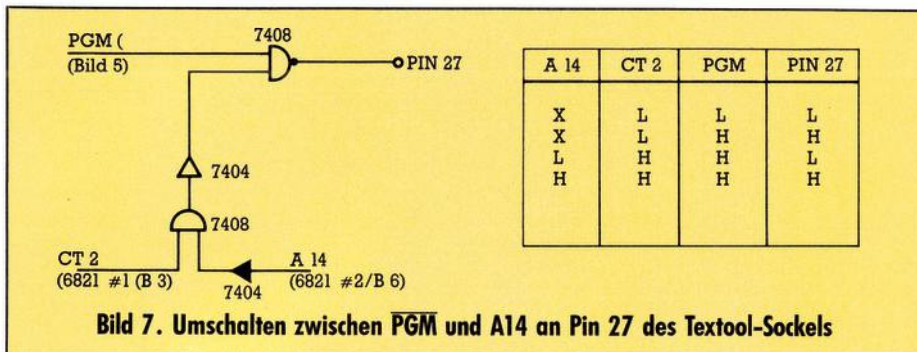
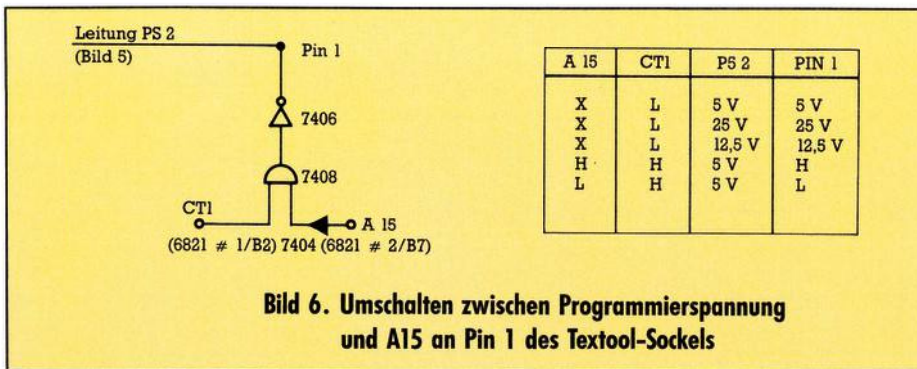


Bild 5. Die Verschaltung des Kontroll-Busses



Dieses Monoflop ist mit Hilfe eines Kondensators von $6,8 \mu\text{F}$ und eines Widerstandes von $20,5 \text{ k}\Omega$ auf 50 ms eingestellt. So ist gewährleistet, daß ein programmierter Spannungsimpuls, selbst bei Ausfall des Computers oder Auftreten eines Interrupts, genau 50 ms dauert (ein längerer Impuls könnte zur Zerstörung des EPROMs führen). Zum Einschalten der verschiedenen Programmiervspannungen dienen Transistoren vom Typ BC327-40, die die Programmiervspannung durchschalten, wenn ihre Basis vom Multiplexer über den angeschlossenen Treiber 7406 auf Masse gezogen wird. Nach dem Passieren des Schalttransistors werden die verschiedenen Spannungen über einen Vorwiderstand mit Zehnerdiode geregelt. Die logischen Zustände bei abgeschalteter Programmiervspannung werden durch Dioden mit Vorwiderstand (logisch »1«) oder Widerstände, die mit Masse verbunden sind, (logisch »0«) geregelt. Weiterhin wird das Signal PGM für die Typen 2764 und 27128 aus dem Multiplexer, Leitung 5, gewonnen. Der Monoflop steuert über einen zweiten Ausgang eine LED, die für die Dauer des Programmierimpulses erlischt und so den Programmiervorgang durch Blinken anzeigt. Außer der Ansteuerung der verschiedenen Programmiervspannungen müssen noch Pin 1, Pin 26 und 27 des Textool-Sockels mit verschiedenen EPROM-Typen beschaltet werden.

Diese sind:

Pin 1: Umschaltung zwischen A15 (27512) und U_p (12,5 oder 25 Volt für 2764/27128/27256)

Pin 27: Umschaltung zwischen A14 (27256/27512) und PGM (2764/27128)

Pin 26: Umschaltung zwischen A13 und U_p (2732/2532)

Die Umschaltung dieser Pins ist in den Bildern 6, 7 und 8 dargestellt.

Die Funktion der Steuerleitungen (CT1 bis CT3), die aus dem 6821 Nummer 1 Port B kommen, kann man den Logikdiagrammen entnehmen. Die Leitungskreuzungen für den seltener verwendeten Typen 2532 wurden mit einem Umschalter, dessen Stellung am 6821 Nummer 1 B6 ausgelesen werden kann, realisiert (Bild 9).

Nachdem hiermit nun die Grundlagen des 64'er EPROM-Programmiergerätes beschrieben wurden, werden wir uns in der nächsten Ausgabe der Bauanleitung, den Bestückungsplänen und dem Platinenlayout widmen.

(Michael Frank/aw)