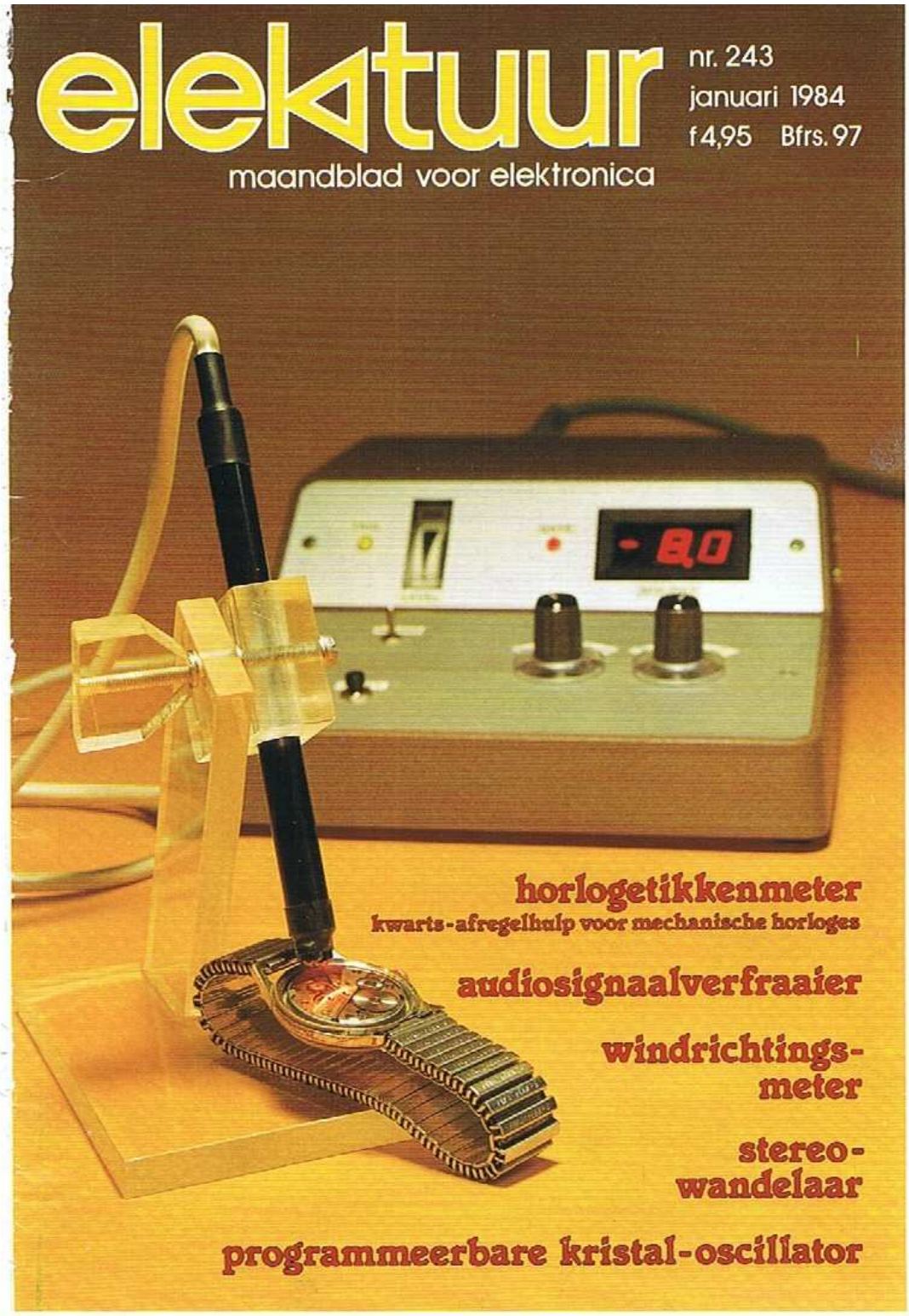
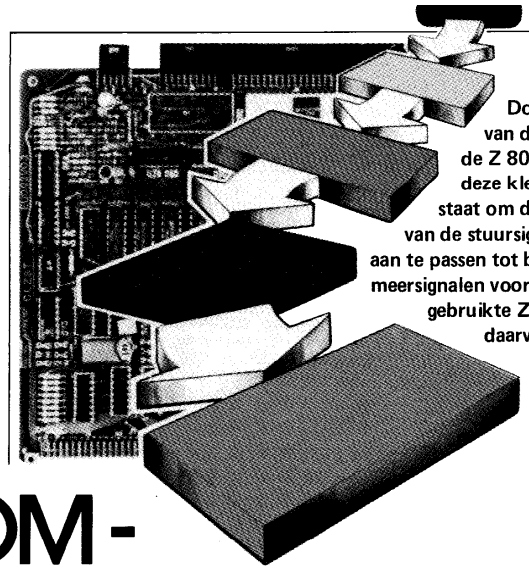


EPROM programmer for Z80, English, Deutsch, Nederlands

EPROM-programmer
voor Z 80
elektuur januari 1984

B. Barink



Door gebruik te maken van de $\overline{\text{WAIT}}$-ingang van de Z 80-microprocessor is deze kleine schakeling in staat om de bijzondere volgorde van de stuursignalen van de Z 80 aan te passen tot bruikbare programmeersignalen voor een 2716. Het gebruikte Z 80-systeem moet daarvoor wel zijn voorzien van een statisch RAM-geheugen.

EPROM-programmer voor Z 80

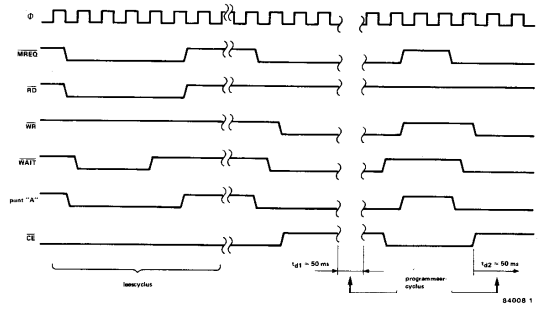
programmeerschakeling voor 2716-EPROM's

Om een 2716-EPROM te kunnen programmeren moeten de volgende signalen op de aansluitpennen van deze EPROM worden gezet: Op pen $\overline{\text{OE}}$ (output enable) moet een "1"-nivo komen te staan, data en adres moeten gedurende de programmeercyclus stabiel blijven, pen V_{pp} moet een programmeerspanning van 25 V krijgen en tenslotte moet pen $\overline{\text{CE}}$ (chip enable) logisch één zijn gedurende minstens 50 ms. Niet moeilijk, maar al die voorwaarden moeten wel strikt worden aangehouden. Verder moet de snelheid van de processor worden aangepast en moeten we letten op enkele bijzonderheden in de volgorde van de kontrolesignalen. Zo is bijvoorbeeld in figuur 1 te zien dat bij een lees-cyclus het $\overline{\text{RD}}$-signaal (read) op hetzelfde moment verschijnt als het $\overline{\text{MREQ}}$-signaal (memory request). Bij een schrijfcyclus is er een vertraging van een klokperiode tussen het verschijnen van $\overline{\text{MREQ}}$ en het logisch nul worden van het $\overline{\text{WR}}$-signaal (write). Daar moet rekening mee worden gehouden, want het programmeren bestaat in feite uit een verlengde schrijf-cyclus. Verder moet de EPROM ergens in het adresseerbare geheugenbereik worden gezet. Daarbij is een adresdekoder nodig (hier niet getekend) die het enable-signaal voor de EPROM levert.

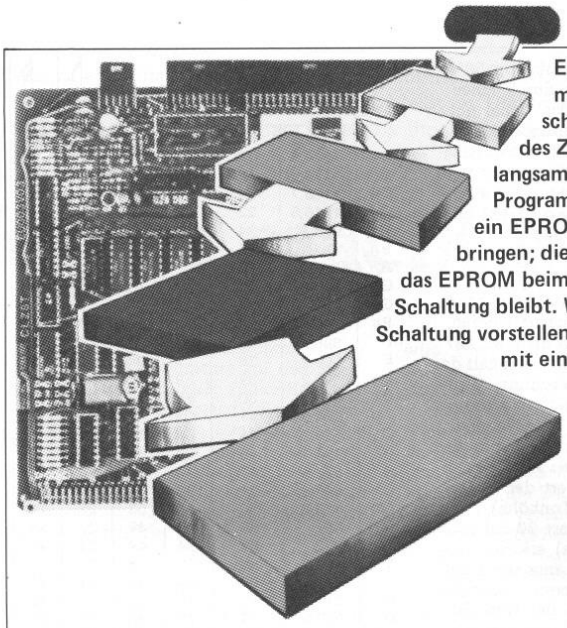
De schakeling en zijn werking

Het adresdekodeersignaal moet komen te staan op punt A in het schema van figuur 2 (logisch nul-nivo). Als het signaal wordt gemaakt uit een kombinatie van de adreslijnen en de $\overline{\text{MREQ}}$-lijn, dan kunnen deze samengevoegd worden door middel van poort N7. Ook een eventueel reeds aanwezig $\overline{\text{ADDRESS}}$-signaal kan aan deze poort worden toegevoegd. Tenslotte kan in sommige gevallen ook het $\overline{\text{PE}}$-signaal (program enable) worden gebruikt als enable-signaal voor de EPROM.

1



Figuur 1. Tijdvolgordediagram van de benodigde Z 80-signalen voor het lezen en schrijven in het geheugen. Let op dat $\overline{\text{WR}}$ iets later dan $\overline{\text{MREQ}}$ verschijnt, terwijl $\overline{\text{RD}}$ tegelijk met $\overline{\text{MREQ}}$ schakelt. Door een vertragingsschakeling wordt het $\overline{\text{WAIT}}$-signaal "0" gehouden als de EPROM wordt geadresseerd, zowel bij een lees- als een schrijfcyclus.



Es ist schon einigermaßen schwierig, die schnellen Steuerbefehle des Z80 und die wesentlich langsameren (und strikten) Programmieranweisungen für ein EPROM unter einen Hut zu bringen; dies umso mehr, wenn das EPROM beim Programmieren in der Schaltung bleibt. Wir wollen eine kleine Schaltung vorstellen, die dieses Problem mit einer geschickten Manipulation des $\overline{\text{WAIT}}$-Eingangs löst.

2716 an Ort und Stelle programmieren

EPROMer für Z 80

B. Barink

Zum Programmieren eines EPROMs 2716 müssen folgende Voraussetzungen erfüllt sein: Der Anschluß $\overline{\text{OE}}$ (output enable = Ausgang-Freigabe) muß logisch 0 sein, die Daten- und Adreßleitungen müssen einen stabilen Pegel haben, der Anschluß V_{CC} muß von 5 V auf eine Programmierspannung von 25 V gelegt werden, und schließlich muß der Anschluß $\overline{\text{CE}}$ (chip enable = Baustein-Freigabe) für 50 ms auf logisch 1 umspringen.
Außerdem muß der Takt des Prozessors verlangsamt werden; dabei den zeitlichen Ablauf der Steuersignale beachten. Das Impulsdiagramm (Bild 1) zeigt, daß das Signal $\overline{\text{RD}}$ (read = Lesen) während eines Lesezyklus gleichzeitig mit dem Speicherzugriffs-Signal MREQ (memory request = Speicher bereit) auftritt. Beim Einschreiben erscheint das Signal MREQ einen Taktimpuls, bevor der Pegel des Signals $\overline{\text{WR}}$ (write = Schreiben) auf logisch 0 abfällt. Das muß man sich klarmachen, denn Programmieren ist auch nichts weiter als ein verlängerter Einschreib-Prozeß. Um überhaupt auf ein EPROM zugreifen zu können, muß man mit den Programmierbefehlen natürlich einen adressierbaren Bereich "ansprechen". Außerdem wird eine Adreßdekodierung gebraucht (die hier nicht weiter besprochen werden soll), die ein Freigabe-Signal für die vom EPROM besetzte Speicherzone erzeugt.

1

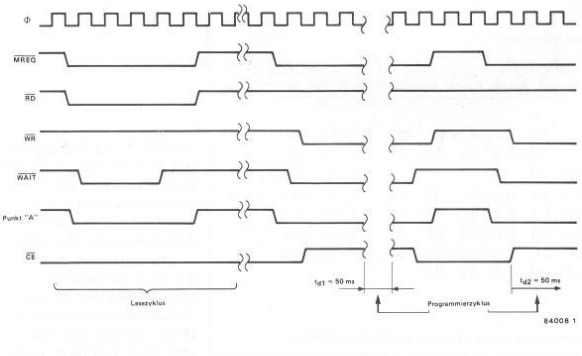


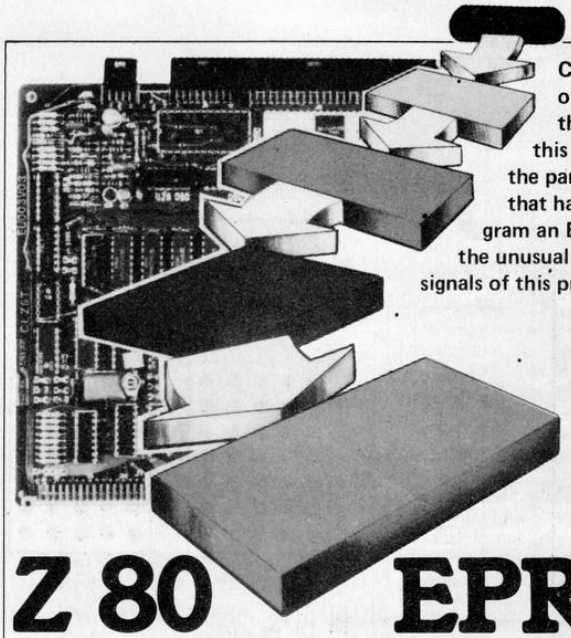
Bild 1. Impulsdiagramm für die Steuersignale des Z 80 während eines Schreib- und eines Lesevorgangs. Besonders auffällig ist, daß $\overline{\text{WR}}$ erscheint, während $\overline{\text{RD}}$ gleichzeitig mit $\overline{\text{MREQ}}$ auftritt. Mit Hilfe einer Warteschleife wird die Leitung $\overline{\text{WR}}$ logisch 0, sobald das EPROM adressiert ist, und das auch im Verlauf eines Schreibzyklus.

Schaltung und Impulsdiagramm

Das Signal zur Adreßdekodierung bewirkt, daß Punkt "A" (im Schaltbild links oben) logisch 0 wird. Wenn dieses Signal erzeugt wird, ohne die Adreßleitungen mit der Leitung MREQ zu verbinden, muß diese Verknüpfung von N7 übernommen werden. Im umgekehrten Fall kann das Dekodierungssignal, hier ADDRESS genannt, so wie es ist, auf Punkt "A" gegeben werden. Wir kommen später noch auf das Signal $\overline{\text{PE}}$ (program enable = Freigabe zum Programmieren) zurück; dieses Signal kann in einigen Anwendungen das Freigabe-Signal ersetzen.

Schreibzyklus

Wenn das EPROM adressiert ist, erzeugt der logische Pegel an Punkt "A" des EPROMers eine abfallende Flanke an N3, wodurch



Careful manipulation of the $\overline{WAIT}$ input of the Z 80 is what enables this little circuit to fulfil the particular conditions that have to be met to program an EPROM *in situ* given the unusual timing of the control signals of this processor.

B. Barink

Z 80 EPROM programmer

any Z 80 system with static RAM can be used to program 2716 EPROMs

In order to program a 2716 EPROM there are several conditions that have to be met. The $\overline{OE}$ (Output Enable) pin must be 'high', the levels on the address and data lines must be stable, the potential on pin V_{pp} must rise from 5 V to the programming voltage of 25 V and finally, the $\overline{CE}$ (Chip Enable) pin must go 'high' for 50 ms. There is nothing really unusual there but a certain amount of care is needed as the speed of the processor must be slowed down and the peculiarities of the timing of the control signals must be taken into account. It is notable, looking at figure 1, that the $\overline{RD}$ (read) signal appears at the same time as the memory validation signal $\overline{MREQ}$ (memory request), whereas during a write operation there is a delay of one clock cycle between the appearance of $\overline{MREQ}$ and the transition to 'low' of the $\overline{WR}$ signal (write). This is important for us as the programming consists of a prolonged write operation. However, to be able to access the EPROM it must be located somewhere in the addressable area. An address decoding (not represented here) is needed to supply a validation signal for the memory zone occupied by the EPROM.

The circuit and its timing

The address decoding signal must set point 'A' in figure 2 logic 'low'. If this signal has been generated without combining the address lines with the $\overline{MREQ}$ line, they can still be combined using OR gate N7. If these signals have already been combined, the

1

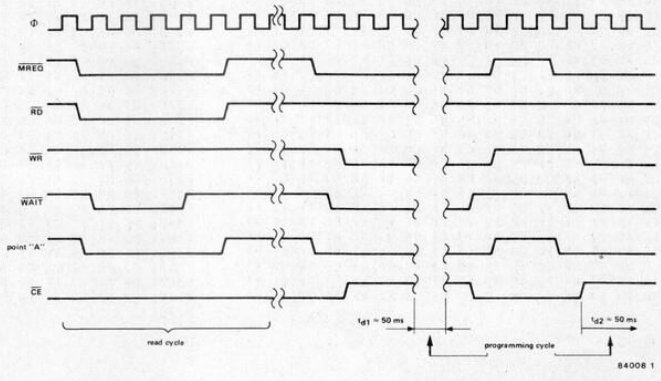


Figure 1. This is the timing diagram for the Z 80 control signals during read and write cycles. It is notable that there is a significant time delay between the appearance of $\overline{MREQ}$ and $\overline{WR}$, whereas $\overline{MREQ}$ and $\overline{RD}$ appear simultaneously. A wait circuit is used to set the $\overline{WAIT}$ line 'low' as soon as the EPROM is addressed, even during a write cycle.

decoding signal, called $\overline{\text{ADDRESS}}$ here can be applied directly to point 'A'. We will return later to the $\overline{\text{PE}}$ (program enable) signal which could, in certain applications, take the place of a validation signal.

Write cycle

When the EPROM is addressed, the logic level applied to point 'A' of the programmer produces a falling edge at the output of N3, which triggers monostable MMV1. A calibrated 50 ms pulse then appears at pin 8 of this IC and is used as a programming pulse at the $\overline{\text{CE}}$ input of the EPROM. This same pulse sets the $\overline{\text{WAIT}}$ input of the Z 80 'low' via N1 and N5 so that the address word and the data word on the buses remain stable. As the $\overline{\text{RD}}$ line is 'high', input $\overline{\text{OE}}$ of the EPROM is also 'high'. At the same time T1 is turned off, T2 saturates and the potential at pin V_{pp} of the EPROM goes from 5 V to 25 V.

None of this will happen, however, if the $\overline{\text{WR}}$ signal is not delayed, as we mentioned at the beginning of this article. In fact the output of OR gate N3 cannot go 'low' unless the $\overline{\text{WR}}$ line is also 'low'. Also the delay introduced by monostable MMV1 must be taken into account. This is the reason for adding a circuit to introduce a 'wait' of several cycles. It consists of a series of flip-flops FF1 . . . FF4, which hold the $\overline{\text{WAIT}}$ pin of the Z 80 'low' immediately after point 'A' goes 'low'. The maximum delay between the time that the $\overline{\text{WAIT}}$ input should go 'low' (making the address and data words on the buses stable) and the time when the 'low' appears on the $\overline{\text{WR}}$ line is about 150 ns. A few dozen ns delay introduced by MMV1 must be added to this. With the four flip-flops we gain three wait cycles, or 750 ns with a 4 MHz clock. As the timing diagram of figure 1 shows, the $\overline{\text{WAIT}}$ input goes 'low' just after $\overline{\text{MREQ}}$, even though the $\overline{\text{WR}}$ line is still 'high'. As soon as the 50 ms $\overline{\text{CE}}$ pulse arrives, the address and data buses are fixed and remain so for the duration of the programming.

Read cycle

The wait circuit is triggered by the address decoding signal, so it also works during the read cycles of the EPROM. This gets over the problem of EPROMs whose access time is normally too long (450 ns). The monostable, on the other hand, is not activated, so $\overline{\text{CE}}$ remains 'low', as the first part of the timing diagram shows. $\overline{\text{OE}}$, however, goes 'low' as soon as $\overline{\text{RD}}$ does. Then all the conditions required for the EPROM to put data onto the bus are met. In order to retain the normal reading speed the wait cycles must be cancelled. This is easily done by linking pin 6 of N4 ($\overline{\text{OE}}$) with pin 4 (PR) of flip-flop FF1, which will then no longer be connected to +5 V.

Programming *in situ*

This is not a totally autonomous EPROM programmer. It is, in fact, an auxiliary circuit in which the EPROM socket has wire wrapping terminals. There are, of course, a few links that have to be wired in: PHI/EX (the clock), $\overline{\text{WAIT}}$, $\overline{\text{RD}}$, $\overline{\text{WE}}$, the address decoding signal (or $\overline{\text{PE}}$) and finally the programming potential of 26 V (not 25 V as there will be some voltage dropped across D3 and T2). Make sure that the address decoding signal ($\overline{\text{ADDRESS}}$) does not contain the $\overline{\text{RD}}$ signal as its presence would prevent any writing, and therefore programming, from taking place.

The programming unit of the polyphonic synthesizer is a nice example of programming *in situ*. If you look at the circuit diagram in the relevant article you will see what we mean. In this case there is no need even to fit a special socket for the EPROM as it takes the place of RAM IC9. The 4071 (IC6) is removed from its socket and the signals for the EPROM are then applied to the pins as follows:

pin 10 (IC6): $\overline{\text{OE}}$ (pin 20 of the EPROM)
pin 11 (IC6): V_{pp} (pin 21 of the EPROM)
pin 4 (IC6): $\overline{\text{CE}}$ (pin 18 of the EPROM)

The clock signal PHI/EX is available at pin 27a of the μP bus, as are $\overline{\text{RD}}$ (at 31c) and $\overline{\text{WR}}$ (31a). Signal $\overline{\text{PE}}$ is available at the output of N10. The $\overline{\text{WAIT}}$ signal is applied to pin 5c of the 64-way connector. Then, whenever the potential of 26 V is present, every operation to write to memory (store enable) causes the EPROM to be programmed.

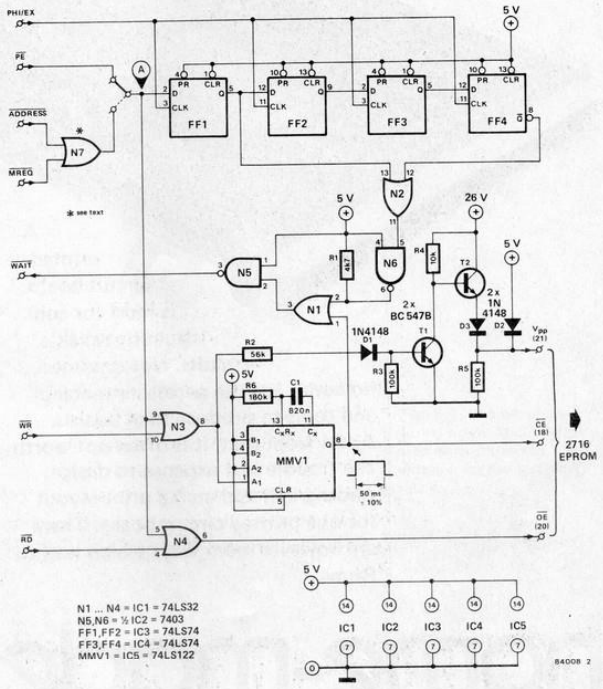


Figure 2. The circuit diagram for the Z 80 2716 EPROM programmer consists of a monostable that generates a calibrated 50 ms programming pulse, and a wait circuit that sets the $\overline{\text{WAIT}}$ line 'low' even before the $\overline{\text{WR}}$ signal appears. By mounting this circuit on a piece of veroboard fitted with 24 wire wrap pins, this programmer could be substituted for the EPROM to be programmed on any memory card with address decoding.