

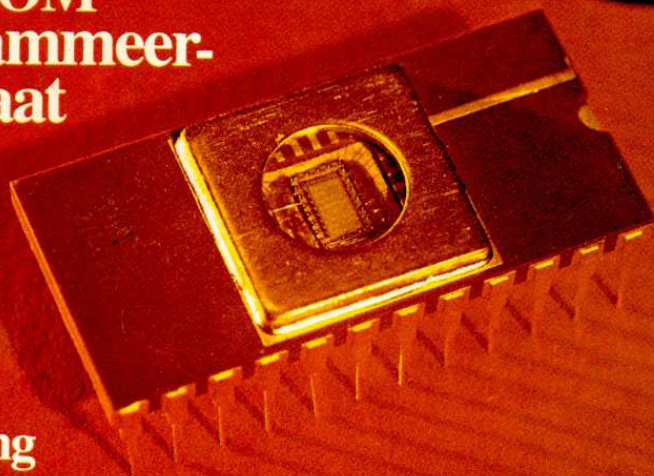
RBB

RADIO BULLETIN

maandblad voor
toegepaste elektronica
jrg 47 • nr. 6 • juni 1978
ned f 3.25 – België F 55

2708

μP EPROM
programmeer-
apparaat



postfading
op elke
recorder

VHF-TV modulator

druppelladen van accu's

6

1978

RB

RADIO BULLETIN

Radio Bulletin is een
maandelijkse uitgave van
uitgeverij De Muiderkring BV,
Nijverheidswerf 17-21, Bussum.
Postadres: postbus 10,
1400 AA Bussum (Holland),
Tel.: 02159-31851, Telex: 15171,
Postgiro 83214,
Bank: Amro-bank, Weesp,
rek. nr. 48.49.54.563.

Redactie
hoofdredacteur: W. Hesselink
eindredacteur: J. G. Arends
technische redacteurs:
D. M. de Boer, J. van de Pol,
D. J. F. Scheper
audioredacteur: W. Jak
redactiesecr.: A. J. Vlaswinkel
techn. adv.: H. B. Stuurman

Telefonisch spreekuur, uitsluitend
over in RB gepubliceerde
schema's:
iedere maandag tussen 16.00 en
17.00 uur op tel. nr. 02159-31851.

Abonnementen
Abonnementsprijs f. 32,50 per vol-
kalenderjaar. Voor een abonne-
ment, dat in de loop van het jaar
wordt opgegeven, geldt een naar
ratio lager tarief. Abonnementen
worden aan het eind van ieder ka-
lenderjaar automatisch verlengd,
tenzij uiterlijk 30 november be-
richt van opzegging is ontvangen.
Betaling van abonnementsgeld
uitsluitend d.m.v. de
toegezonden accept-girokaart.
Teneinde vertraging in de afwik-
keling van correspondentie over
abonnementszaken te voorkom-
en verzoeken wij u vriendelijk in
brieven en telefoongesprekken
steeds uw *abonneenummer* te
vermelden. Dit nummer is afge-
drukt op de adreswikkels van het
blad.

Advertenties
Tarieven worden op aanvraag
verstrekkt. Teksten en illustra-
tiemateriaal dienen uiterlijk op de
6de van de maand, voorafgaande
aan de maand van verschijning,
in het bezit te zijn van de adver-
tentieafdeling: J. J. de Wit en
mv. M. Schram-Sluyk.

RB in België
RB heeft ook een speciale
Belgische editie.
Voor abonnementen en adverten-
ties wordt uitgeverij De Muider-
kring in België vertegenwoordigd
door: Maarten Kluwer's
Internationale Uitgevers Onder-
neming NV,
Generaal Capiaumontstraat 15,
B2600 Berchem-Antwerpen,
Tel. 031-36.05.24,
Giro 000-0925940-75,
Kredietbank 405-3035001-96.



De 2708, 8192 programmeer-
bare bitjes in één IC, goed om
een 1k programma in op te
slaan.

Inhoud

- | | |
|-----|--|
| 121 | 21e internat. elektronicatentoon-
stelling Parijs |
| 123 | nieuw type hoofdtelefoon |
| 124 |  EPROM programmeer-
apparaat met KIM |
| 135 | druppelladen van accu's |
| 139 | postfading op elke recorder |
| 143 | activiteitenrevue |
| 146 | programmeerbare rekenmachines,
deel II |
| 153 | LF-millivoltmeter |
| 156 | lezers peinsden |
| 157 | VHF-televisiemodulator |

Het geheel of gedeeltelijk overnemen van de inhoud van RB zonder toestem-
ming is verboden. Gepubliceerde schakelingen, e.d. kunnen door een Neder-
lands octrooi zijn beschermd, in welk geval de octrooiwet alleen toepassing
voor persoonlijk gebruik toestaat. Voor de gevolgen van onverhoopte fouten in
tekeningen en bouwbeschrijvingen wordt geen aansprakelijkheid aanvaard.

Volgende maand in RB

**synthesizer
voor blazers,**
volledige beschrijving
van een synthesizer, die
kan worden 'bespeeld'
als een blaasinstrument

**verlichtingsautomaat
voor aquarium**

**frequentie- en
impulsteller**
tevens klok en
klokkelijkzetter.

**frequentie- en
capaciteitsmeter**

verschijnt maandelijks
juni 1978
47ste jaargang/nr. 6

EPROM PROGRAMMEER- APPARAAT MET DE KIM

Al eerder hebben we het in RB gehad over geheugen-uitbreiding (november 1977). Toen betrof het een RAM uitbreiding, dus een extra geheugen waarin geschreven en gelezen kan worden. Waarschijnlijk heeft u ook wel eens in stilte gedacht aan extra ROM, waarin u dan de meest voorkomende programma's zet. Het voordeel is dan dat de programma's meteen na inschakelen beschikbaar zijn. Er is echter één maar: Hoe krijg je je programma in ROM? Een ROM wordt tijdens fabricage van een door de klant gewenst programma voorzien. De kosten hiervan zijn echter zó hoog, dat alleen zeer grote series op deze manier gemaakt kunnen worden. Een aantrekkelijk alternatief is de PROM, Programmable Read Only Memory. Dit geheugen kan m.b.v. een PROM-programmer door de gebruiker zelf van de gewenste inhoud worden voorzien. Eenmaal bij de PROM is het nog maar één stap verder naar de EPROM.

Ook de EPROM (Erasable PROM) kan door de gebruiker worden geprogrammeerd. Hij kan echter bovendien met ultraviolet licht weer worden gewist. Hierdoor kunnen we een EPROM met verouderde of achterhaalde programma's gewoon wissen, en opnieuw programmeren. De meest populaire EPROM is op het moment de 2708 (1k byte) die mede door de ontwikkeling van zijn opvolgers (de 2716 en 2732 met resp. 2k en 4k byte) de afgelopen tijd sterk in prijs is gedaald. Het programmeren van zo'n EPROM is een karwei dat uitstekend door de KIM kan worden opgeknapt. In dit eerste deel kunt u lezen hoe de KIM, door toevoeging van een paar eenvoudige schakelingen, de taak overneemt van een dure PROM-programmer. In een volgend artikel zullen we dan enige uitvoeringsvormen beschrijven.

Het programmeren

Volgens de fabrieksspecificaties moeten alle bits van de 2708 worden geprogrammeerd. Een 'lege' 2708 bevat slechts logische enen. Door het programmeren kunnen we hier nullen van maken. De 2708 heeft 10 adreslijnen, A0 t/m A9 (afb. 1). Met deze 10 adreslijnen kunnen we alle $2^{10} = 1024$ adressen bereiken. Op ieder adres staan 8 bits. We hebben 8 datalijnen. Deze datalijnen zijn normaal op de databus van onze KIM aangesloten. Eenmaal geselecteerd (een '0' op CS/WE, pin 20) fungeren deze datalijnen als uitgang.

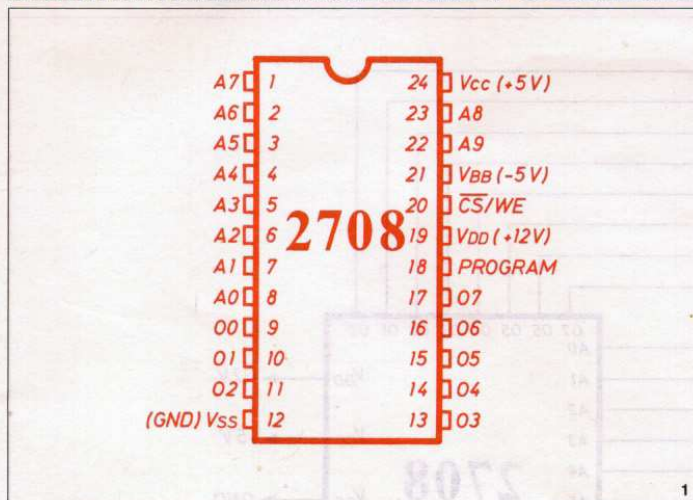
Bij het programmeren worden deze 8 datalijnen als ingang gebruikt. Zolang het programmeren duurt moet pin 20 (CS/WE) op +12 V worden gehouden.

Het programmeren geschiedt door pulsen van 26 V op de programmeer-ingang te zetten. De programmeer-ingang moet actief naar 26 V worden geschakeld en ook actief naar aarde. De flanken moeten ongeveer 1 μ s duren. De schakeling van afb. 2 blijkt goed te voldoen. De 7406 is een 6-voudige TTL inverter met open collectoruitgang. Op deze uitgang mag 30 V worden gezet.

De transistor moet een snel type zijn met een geringe storage. Een 2N2905 is hier te traag. Tijdens zo'n programmeerpuls mogen data en adreslijnen niet veranderen. Alle adressen worden nu achter elkaar geprogrammeerd met elk een programmeerpuls van 1 ms.

Het programmeren van de hele ROM duurt dan ongeveer $1000 \times 1 \text{ ms} = 1 \text{ s}$.

Deze cyclus wordt 100 x herhaald en dan is de EPROM ingelezen. Het hele programmeerproces wordt bestuurd door onze KIM.



1 Bovenaanzicht van de 2708 EPROM

2 De interface tussen de KIM en de te programmeren EPROM

De aansluitingen bij het programmeren

We hebben gezien dat voor het programmeren 10 adreslijnen, 8 datalijnen, 1 chipselectlijn en 1 programmeerlijn, dus totaal 20 aansluitingen nodig zijn. Onze KIM heeft 15 I/O lijnen. Er moeten dus nog 5 lijnen ergens anders vandaan komen. Hiervoor gebruiken we nog 5 I/O lijnen van het keyboard/display. We kunnen het display dus niet laten werken tijdens het programmeren. Maar

dit was anders ook uitgesloten omdat één keer door de SCAND loop van het monitorprogramma 4 milliseconden kost, terwijl de programmeerpulsen maximaal 1 ms mogen duren. De aansluitingen van de 2708 bij het programmeren staat in afb. 3. We zien dat de datalijnen zijn aangesloten op PPA (peripheral port A), de 5 laagste adreslijnen op PBO ... PB4 en de 5 hoogste adreslijnen op COLA ... COLE. De inverter in de CS/WOE lijn dient om het

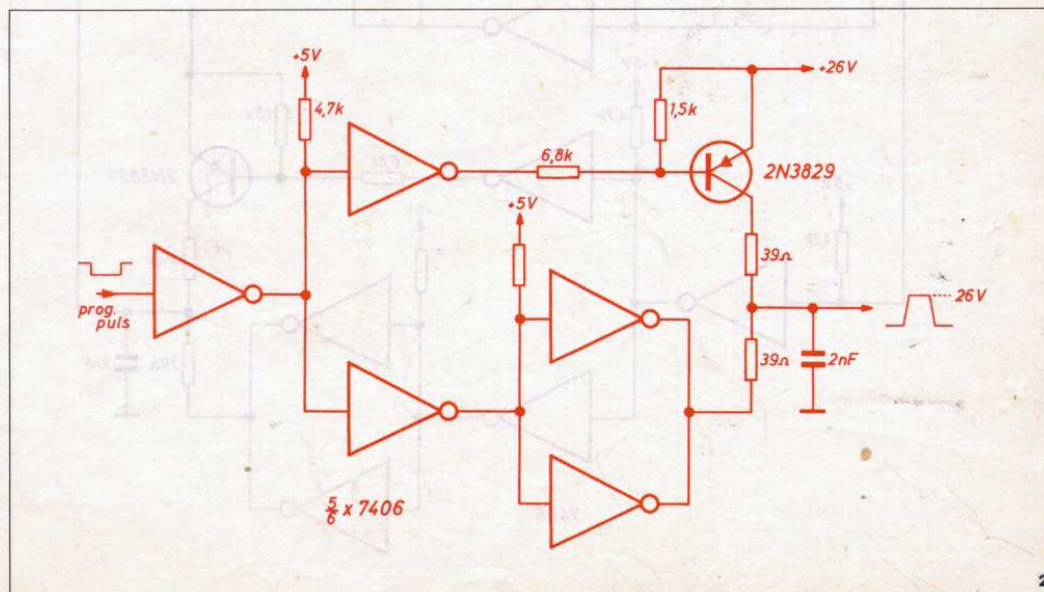
logische '1' niveau op 12 V te brengen. Tijdens het programmeren moet PB5 dus '0' blijven, waardoor CS/WE 12 V wordt. Bij het uitlezen (de inhoud van de 2708 moet worden gecontroleerd) moet PB5 '1' worden, waardoor CS/WE '0' wordt. De programmeerpulsen worden gemaakt d.m.v. PB7. De adressen worden geteld met zero-page-adressen FA en FB. Dit heeft als voordeel dat bij een eventuele fout de displaybuffer reeds gevuld is met het adres van de fout. De plaatsen 1780 t/m 1788 worden ook gebruikt door ons hulpprogramma. Dus deze adressen moeten we verder vermijden.

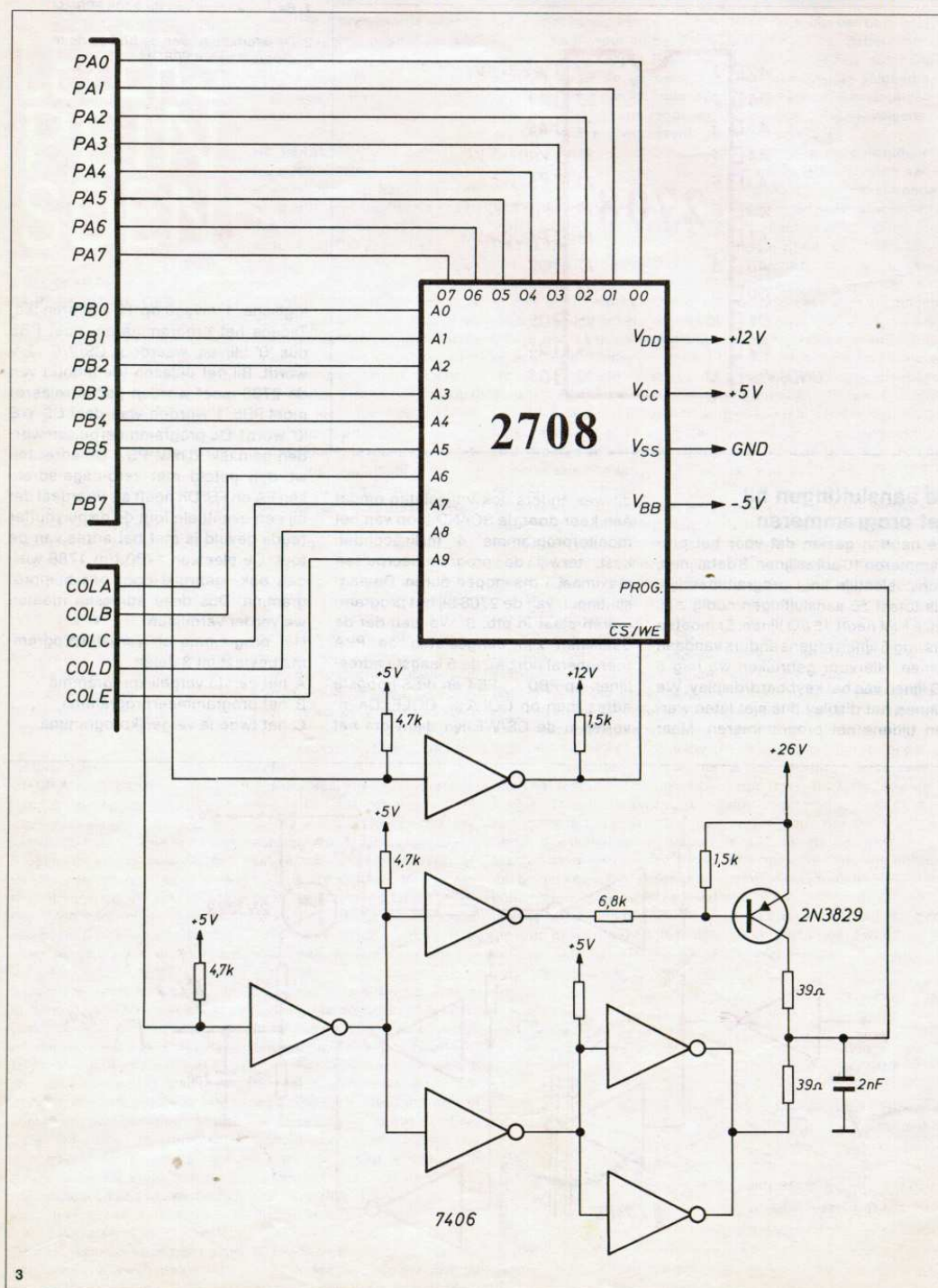
Het programma en vergelijkprogramma bestaat uit 3 delen:

A het eerste vergelijkprogramma

B het programmeerprogramma

C het tweede vergelijkprogramma.





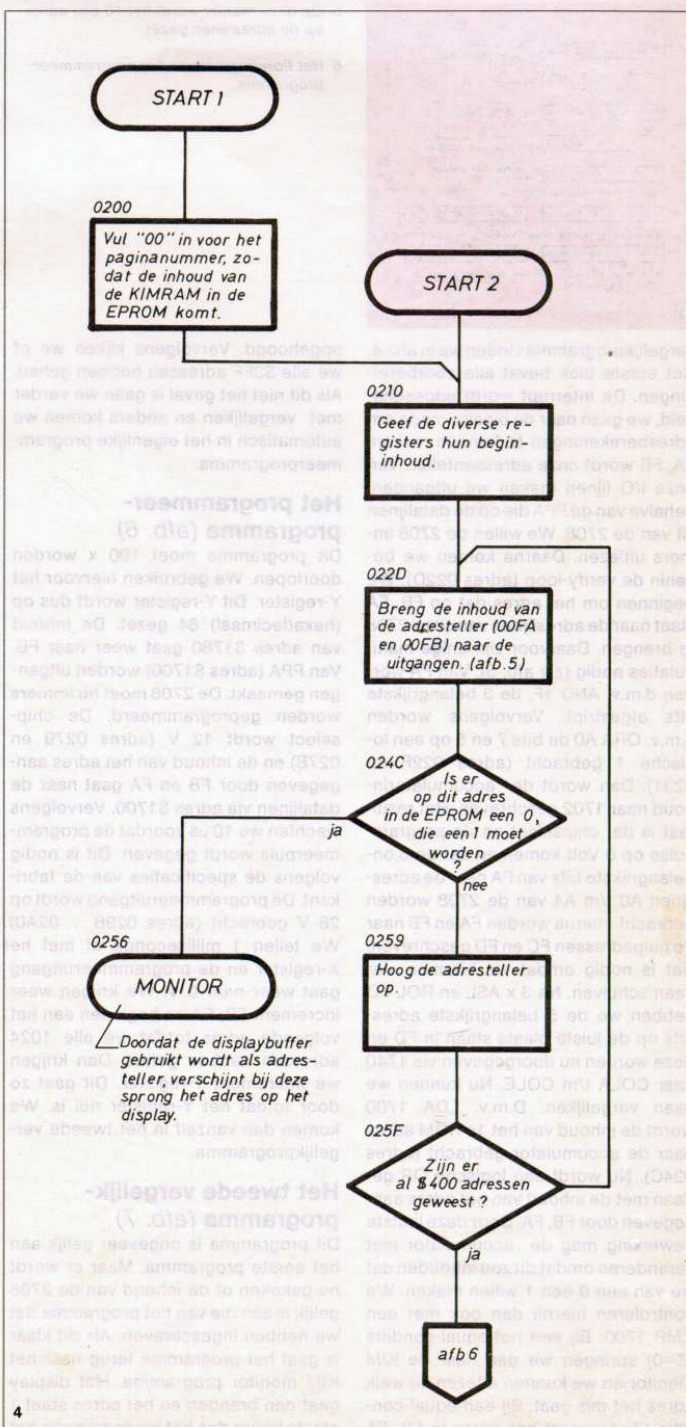
3 Het complete programmeerapparaat, zoals het op de KIM kan worden aangesloten

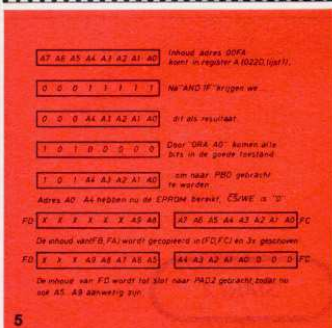
4 Flowdiagram van het eerste vergelijkprogramma. Wanneer nergens een '0' in een '1' verandert, gaan we door naar het programmeerprogramma. (afb. 6)

Het eerste vergelijkprogramma

Voordat het eigenlijke programmeren begint gaan we eerst kijken of het wel mogelijk is ons programma in de 2708 te schrijven. Er geldt namelijk de regel dat we nooit van een 0 een 1 mogen maken. Als de 2708 goed is gewist is het programmeren natuurlijk altijd mogelijk, maar het komt weleens voor dat er een 0 blijft 'hangen'. Verder hebben we de mogelijkheid ons programma te 'updaten'. Dit betekent dat we een bestaand programma in de 2708 kunnen uitbreiden, mits er op de niet gebruikte adressen 'SFF' staat. We schrijven daartoe eerst de inhoud van onze 2708 die op de programmeerplaats staat naar een stuk 1k RAM geheugen. Het nieuwe stuk programma wordt bij de RAM geschreven, en vervolgens programmeren we de EPROM opnieuw. Zolang er van een '0' géén '1' gemaakt hoeft te worden, is het niet nodig de EPROM te wissen.

We moeten dus voor elk adres controleren of er niet van een 0 een 1 gemaakt zou worden. Ons vergelijkprogramma start op adres 0210 + n. x 400. Deze adresnotatie lijkt misschien vreemd maar geeft aan dat het programmeerprogramma relocatable is. Dat betekent dat het overall neergezet kan worden in de adresruimte van de KIM. Deze verplaatsbaarheid is bereikt door in het programma geen eigen subroutines te gebruiken en geen JMP's, omdat we hierbij aangewezen zijn op absolute adressen. Voordat we ons programma starten moeten we op adres 1780 de eerste 8 bits (het bladzijdenummer) vermelden van het programma dat we in de ROM willen schrijven. Als we starten op 0200 + n. 400 wordt er 00 genoteerd op 1780 en daarna gaan we verder met 0210. We programmeren dan dus vanuit de KIM-RAM. Het flowdiagram van het eerste





5

vergelijkprogramma vinden we in afb. 4. Het eerste blok bevat alle voorbereidingen. De interrupt wordt uitgeschakeld, we gaan naar de binaire mode om adresberekeningen te kunnen maken. FA, FB wordt onze adressenteller. Van onze I/O lijnen maken we uitgangen, behalve van de PPA die op de datalijnen zit van de 2708. We willen de 2708 immers uitlezen. Daarna komen we bovenin de verify-loop (adres 022D). We beginnen om het adres dat op FB, FA staat naar de adreslijnen van onze 2708 te brengen. Daarvoor zijn enige manipulaties nodig (zie afb. 5). Van FA worden d.m.v. AND 1F, de 3 belangrijkste bits afgestript. Vervolgens worden d.m.v. OR A0 de bits 7 en 5 op een logische 1 gebracht (adres 022F en 0231). Dan wordt de accumulatorinhoud naar 1702 geschreven. Het resultaat is dat chipselect en de program-pulse op 0 Volt komen en dat de 5 onbelangrijkste bits van FA naar de adreslijnen A0 t/m A4 van de 2708 worden gebracht. Hierna worden FA en FB naar de hulpadressen FC en FD geschreven. Dat is nodig omdat we hiermee wat gaan schuiven. Na 3 x ASL en ROL FD hebben we de 5 belangrijkste adresbits op de juiste plaats staan in FD en deze worden nu doorgegeven via 1740 naar COLA t/m COLE. Nu kunnen we gaan vergelijken. D.m.v. LDA 1700 wordt de inhoud van het 1e ROM adres naar de accumulator gebracht (adres 024C). Nu wordt een logische OR gedaan met de inhoud van het adres aangegeven door FB, FA. Door deze laatste bewerking mag de accumulator niet veranderen omdat dit zou inhouden dat we van een 0 een 1 willen maken. We controleren hierna dan ook met een CMP 1700. Bij een not-equal-conditie (Z=0) springen we dan naar de KIM monitor en we kunnen aflezen bij welk adres het mis gaat. Bij een equal-conditie (Z=1) wordt het adres in FB, FA

5 Op deze manier wordt het 10 bits adres op de adreslijnen gezet

6 Het flowdiagram van het programmeerprogramma.

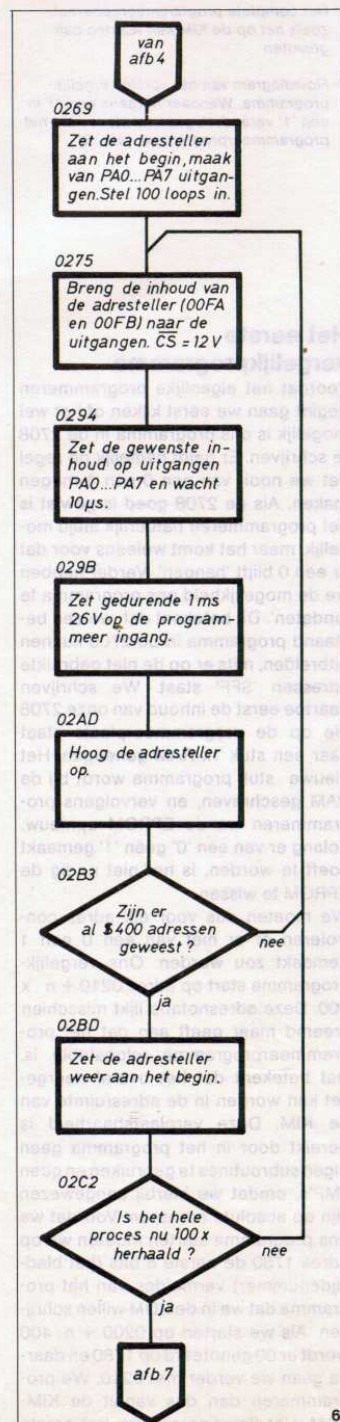
opgehoogd. Vervolgens kijken we of we alle \$3FF adressen hebben gehad. Als dit niet het geval is gaan we verder met vergelijken en anders komen we automatisch in het eigenlijke programmeerprogramma.

Het programmeerprogramma (afb. 6)

Dit programma moet 100 x worden doorlopen. We gebruiken hiervoor het Y-register. Dit Y-register wordt dus op (hexadecimaal) 64 gezet. De inhoud van adres \$1780 gaat weer naar FB. Van PPA (adres \$1700) worden uitgangen gemaakt. De 2708 moet nu immers worden geprogrammeerd. De chipselect wordt 12 V (adres 0279 en 027B) en de inhoud van het adres aangegeven door FB en FA gaat naar de datalijnen via adres \$1700. Vervolgens wachten we 10 μ s voordat de programmeerpuls wordt gegeven. Dit is nodig volgens de specificaties van de fabrikant. De programmeeruitgang wordt op 26 V gebracht (adres 029B... 02A0). We tellen 1 milliseconde af met het X-register en de programmeeruitgang gaat weer naar 0 V. We krijgen weer increment FB, FA en beginnen aan het volgende adres totdat we alle 1024 adressen hebben gehad. Dan krijgen we decrement Y-register. Dit gaat zo door totdat het Y-register nul is. We komen dan vanzelf in het tweede vergelijkprogramma.

Het tweede vergelijkprogramma (afb. 7)

Dit programma is ongeveer gelijk aan het eerste programma. Maar er wordt nu gekeken of de inhoud van de 2708 gelijk is aan die van het programma dat we hebben ingeschreven. Als dit klaar is gaat het programma terug naar het KIM monitor programma. Het display gaat dan branden en het adres staat 1 plaats hoger dan het eindadres van het



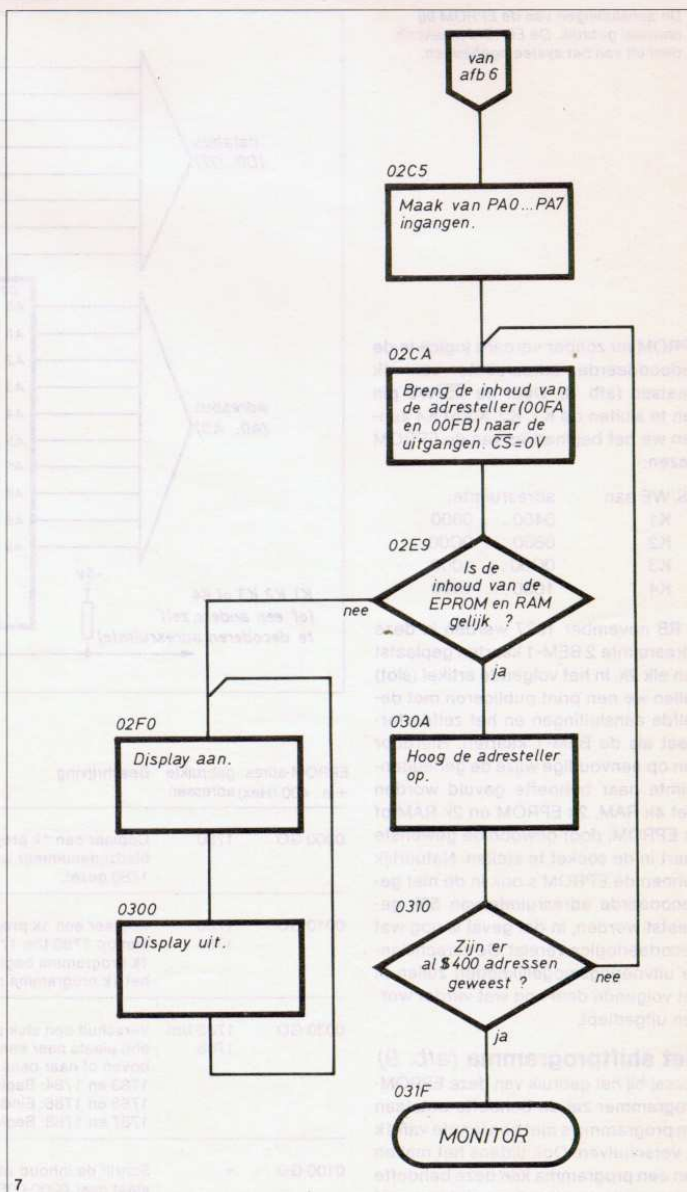
6

7 Het flowdiagram van het tweede vergelijkprogramma. Hier wordt gecontroleerd of de EPROM goed is ingeschreven.

programma dat ingeschreven is. Als er een fout wordt geconstateerd door het 2e vergelijkprogramma, gaat het display knipperen. Dit knipperen gebeurt door afwisselend 256 keer door 'SCANDS' te gaan en daarna een dubbele vertragsingsloop te doorlopen. Voordat we naar SCANDS gaan moeten we eerst de I/O port 1740 weer goed zetten. Dit gebeurt met de subroutine 'INITS' van het monitorprogramma. Op F9 wordt de inhoud van 1700 geschreven. We kunnen dus ook zien wat er in de ROM gekomen is. Als we vervolgens op RESET drukken, kunnen we zien wat er op die plaats had moeten staan. Bij de oude 2708 van de firma INTEL krijgen we hier vaak foutmeldingen. Dat komt omdat dit IC bij het programmeren erg warm wordt. Als we deze 2708 later uitlezen blijkt het er altijd goed in te staan. Misschien zou een sterk ventilatortje hier helpen. Goede resultaten kreeg ik altijd met de TMS 2708 en de TMS 27L08 van Texas Instruments.

Het terugschrijfprogramma (lijst 2)

Als we bij een geprogrammeerde 2708 data willen inschrijven kunnen we dat alleen doen op een gedeelte waar alleen enen staan (FF). We zetten de ROM dan op de programmeerplaats en we schrijven de ROM naar een stuk 1k RAM geheugen. Daartoe maken we gebruik van het terugschrijfprogramma. Dit programma heeft 2 startadressen nl. 100 en 110. Als we starten op 100, wordt adres \$1780 op 00 gezet. We schrijven dan de inhoud van de EPROM welke op de programmeerplaats staat naar de 1K RAM van de KIM. Het programma is vrijwel gelijk aan het 1e vergelijkprogramma, alleen wordt hier de inhoud van de EPROM overgeschreven naar de geheugenplaats aangegeven door de inhoud van de adressen \$FB



en \$FA. Aan het eind van dit programma springen we weer naar de KIM monitor.

Door het terugschrijfprogramma te starten terwijl de programmeerplaats leeg is, komt er alleen \$FF in de RAM te staan. Hierdoor zullen na het intypen van een programma alle niet gebruikte geheugenplaatsen op \$FF staan, zodat

we de mogelijkheid hebben het ongebruikte deel van de EPROM later nog eens te programmeren.

De aansluitingen van de EPROM bij normaal gebruik

Eenmaal geprogrammeerd bevat de EPROM een programma van maximaal 1k byte (1024 adressen). We kunnen de

8 De aansluitingen van de EPROM bij normaal gebruik. De EPROM maakt nu deel uit van het systeemgeheugen.

EPROM nu zonder verdere logica in de gedecodeerde adresruimte van 4k plaatsen (afb. 8). Door de CS/WE pin aan te sluiten op K1, K2, K3 of K4 kunnen we het beginadres van de EPROM kiezen:

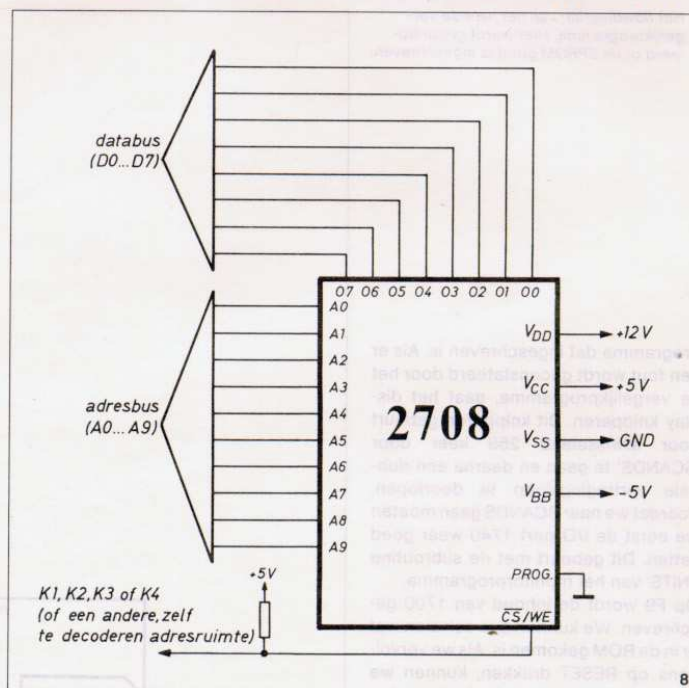
CS/WE aan:	adresruimte:
K1	0400 ... 0800
K2	0800 ... 0C00
K3	0C00 ... 1000
K4	1000 ... 1400

In RB november 1977 werden in deze adresruimte 2 BEM-1 kaarten geplaatst van elk 2k. In het volgende artikel (slot) zullen we een print publiceren met dezelfde aansluitingen en het zelfde formaat als de BEM-1 kaarten. Hierdoor kan op eenvoudige wijze de geheugenruimte naar behoefte gevuld worden met 4k RAM, 2k EPROM en 2k RAM of 4k EPROM, door gewoon de gewenste kaart in de socket te steken. Natuurlijk kunnen de EPROM's ook in de niet gedecodeerde adresruimte van 56k geplaatst worden, in dat geval is nog wat decodeerlogica vereist. De verschillende uitvoeringsmogelijkheden zullen in het volgende deel nog wat verder worden uitgediept.

Het shiftprogramma (afb. 9)

Vooral bij het gebruik van deze EPROM-programmer zal de behoefte ontstaan om programma's met een lengte van 1k te verschuiven. Ook tijdens het maken van een programma kan deze behoefte zich voordoen, zij het nu stukken met een willekeurige lengte. Voordat we het shiftprogramma starten moet daarom bekend zijn:

- Het beginadres van het te verschuiven programma op 1783 en 1784;
 - Het eindadres van het te verschuiven programma op 1785 en 1786;
 - Het nieuwe beginadres van het programma op 1787 en 1788
- (steeds op het eerste adres de laatste



EPROM-adres gebruikte Beschrijving
+ n. 400 (Hex) adressen

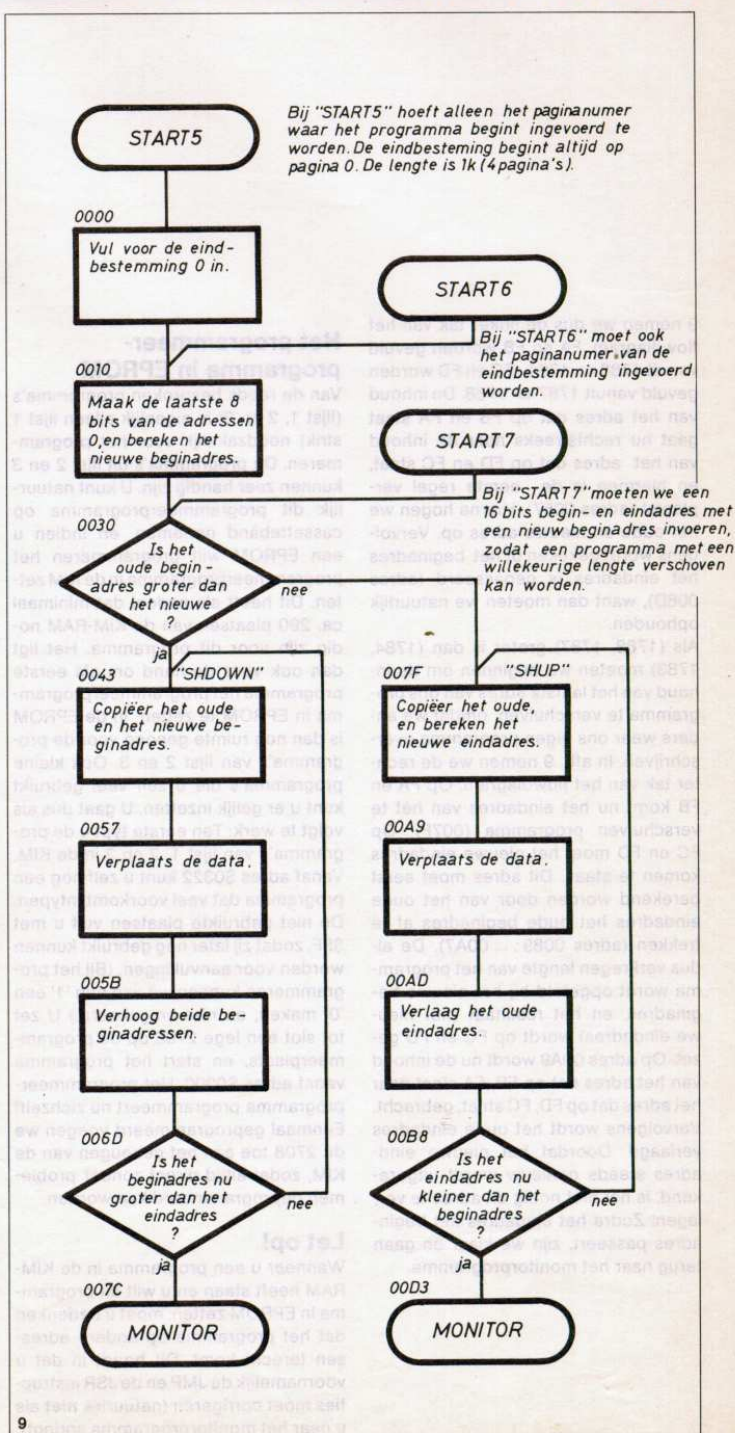
0000 GO	1780	Copieer een 1k programma naar 0000-03FF (KIM-RAM). Het bladzijdenummer waar het 1k programma begint wordt op 1780 gezet.
0010 GO	1780 1781	Copieer een 1k programma naar een ander stuk RAM (schrijf niet op 1780 t/m 1788). Plaats het bladzijdenummer waar het 1k programma begint op 1780 en het bladzijdenummer waar het 1k programma naar toe moet op 1781.
0030 GO	1783 t/m 1788	Verschuif een stuk programma van willekeurige lengte van de ene plaats naar een andere plaats. We kunnen hiermee naar boven of naar beneden schuiven. 1783 en 1784: Beginadres van het te verschuiven programma. 1785 en 1786: Eindadres van het te verschuiven programma. 1787 en 1788: Beginadres waar het programma naar toe moet.
0100 GO	-	Schrijf de inhoud van een 2708 die op de programmeerplaats staat naar 0000-03FF.
0110 GO	1780	Schrijf de inhoud van een 2708 die op de programmeerplaats staat naar een stuk RAM. Plaats het bladzijdenummer waar het programma naar toe moet op 1780.
0200 GO	-	Programmeer een 2708 die op de programmeerplaats staat. Het programma dat wordt overgenomen staat op 0000-03FF.
0210 GO	1780	Programmeer een 2708 die op de programmeerplaats staat. Plaats het bladzijdenummer waar het programma begint op 1780.

9 Het flowdiagram van het verplaats-programma.

cijfers, en op het tweede adres de eerste cijfers.) Het eindadres van het nieuwe programma volgt natuurlijk uit deze gegevens. De adressen 1783... 1788 gebruiken we ook als adrestellers. Omdat we indirect willen werken moeten we ook gebruik maken van zeropage adressen. Hiervoor nemen we FA, FB, FC en FD. Bij elk adres dat we verschuiven worden deze zeropage adressen weer gevuld met de waarden uit 1783... 1788. Het voordeel hiervan is dat we nu ook in de zeropage programma's kunnen verschuiven zonder dat onze tellers worden overschreven. Het spreekt vanzelf dat we nooit mogen schiften naar het gebied van 1783... 1788.

Ons programma heeft 3 ingangen, nl. 0000, 0010 en 0030. (Natuurlijk ook weer + n. 400.) Als we willen starten op 0030, moeten we vooraf de adressen 1783... 1788 invoeren. Als we starten op 0010, verschuiven we altijd een blok van 1k. We moeten dan vooraf op 1780 aangeven waar het programma begint (paginanummer, hex) en op 1781 waar het programma naar toe moet (paginanummer, hex). Als we starten op 0000, wordt door het programma '00' op adres 1781 gezet. Hierdoor kunnen we een 1k blok uit het geheugen naar de KIM-RAM schrijven.

Als we een programma willen verschuiven zijn er twee mogelijkheden, het programma moet naar boven (lagere adressen, of naar onderen (hogere adressen)). We moeten eerst vaststellen of ons programma naar boven of naar beneden schuift. Dit kunnen we nagaan door beide beginadressen te vergelijken, dus door de inhoud van 1784, 1783 en de inhoud van 1788, 1787 van elkaar af te trekken (adressen 0030... 0041). Stel dat (1784, 1783) is groter dan (1788, 1787), dan moeten we dus omhoog schuiven. In afbeelding



9 nemen we dus de linker tak van het flowdiagram. FA en FB worden gevuld vanuit 1783 en 1784., FC en FD worden gevuld vanuit 1787 en 1788. De inhoud van het adres dat op FB en FA staat gaat nu rechtstreeks naar de inhoud van het adres dat op FD en FC staat, en hiermee is de eerste regel verplaatst (adres 0057). Hierna hogen we het oude en nieuwe adres op. Vervolgens wordt gekeken of het beginadres het eindadres is gepasseerd (adres 006D), want dan moeten we natuurlijk ophouden.

Als (1788, 1787) groter is dan (1784, 1783) moeten we beginnen om de inhoud van het laatste adres van ons programma te verschuiven, omdat we anders weer ons eigen programma overschrijven. In afb. 9 nemen we de rechter tak van het flowdiagram. Op FA en FB komt nu het eindadres van het te verschuiven programma (007F). Op FC en FD moet het nieuwe eindadres komen te staan. Dit adres moet eerst berekend worden door van het oude eindadres het oude beginadres af te trekken (adres 0089 ... 00A7). De aldus verkregen lengte van het programma wordt opgeteld bij het nieuwe beginadres, en het resultaat (het nieuwe eindadres) wordt op FC en FD gezet. Op adres 00A9 wordt nu de inhoud van het adres dat op FB, FA staat naar het adres dat op FD, FC staat, gebracht. Vervolgens wordt het oude eindadres verlaagd. Doordat het nieuwe eindadres steeds opnieuw wordt uitgerekend, is het niet nodig dit adres te verlagen. Zodra het eindadres het beginadres passeert, zijn we klaar en gaan terug naar het monitorprogramma.

Het programmeerprogramma in EPROM

Van de reeds besproken programma's (lijst 1, 2 en 3) is eigenlijk alleen lijst 1 strikt noodzakelijk voor het programmeren. De programma's uit lijst 2 en 3 kunnen zeer handig zijn. U kunt natuurlijk dit programmeerprogramma op cassetteband opnemen, en indien u een EPROM wilt programmeren het programmeerprogramma in de KIM zetten. Dit heeft als nadeel dat minimaal ca. 290 plaatsen van de KIM-RAM nodig zijn voor dit programma. Het ligt dan ook voor de hand om als eerste programma het programmeerprogramma in EPROM te zetten. In de EPROM is dan nog ruimte genoeg voor de programma's van lijst 2 en 3. Ook kleine programma's die u zelf veel gebruikt kunt u er gelijk inzetten. U gaat dus als volgt te werk: Ten eerste typt u de programma's van lijst 1, 2 en 3 in de KIM. Vanaf adres \$0322 kunt u zelf nog een programma dat veel voorkomt intypen. De niet gebruikte plaatsen vult u met SFF, zodat zij later nog gebruikt kunnen worden voor aanvullingen. (Bij het programmeren kunnen we van een '1' een '0' maken, en niet omgekeerd.) U zet tot slot een lege 2708 op de programmeerplaats, en start het programma vanaf adres \$0200. Het programmeerprogramma programmeert nu zichzelf! Eenmaal geprogrammeerd voegen we de 2708 toe aan het geheugen van de KIM, zodat altijd direct zonder problemen geprogrammeerd kan worden.

Let op!

Wanneer u een programma in de KIM-RAM heeft staan en u wilt dit programma in EPROM zetten, moet u bedenken dat het programma op andere adressen terecht komt. Dit houdt in dat u voornamelijk de JMP en de JSR instructies moet corrigeren (natuurlijk **niet** als u naar het monitorprogramma springt).

Bij gebruik van tabellen soms ook de LDA instructies. Wanneer u over geheugenuitbreiding beschikt kunt u het programma het best ontwikkelen op de plaats waar het later in EPROM komt, dan weet u zeker dat het werkt. Ook moet u er om denken dat de EPROM-ruimte niet gebruikt kan worden om tijdelijk data in op te slaan. Sommige hobbyisten hebben de aardige gewoonte om het programma zichzelf te laten veranderen, wat bij de EPROM natuurlijk niet mogelijk is.

Vervolg

In het tweede en laatste deel zullen we nog ingaan op de extra voedingsspanning (-5 V, de 12 V en 5 V hadden we al). Tevens zullen enkele suggesties gegeven worden voor de praktische uitvoering van deze programmer.

Lijst 1

0200	A9 00	START1	LDA,imm	S00	Zet paginanummer op 0
0202	80 80 17		STA,abs	PAG1	
0205	F0 09		BEQ,rel	START2	Ga naar START2
0210	78	START2	SEL,impl		
0211	D8		CLD,impl		
0212	A9 00		LDA,imm	S00	
0214	80 01 17		STA,abs	PADD1	
0217	85 F8		STA,Zpage	INH	
0219	85 FA		STA,Zpage	POINTL	
021B	80 43 17		STA,abs	PBDD2	
021E	AD 80 17		LDA,abs	PAG1	
0221	85 FB		STA,Zpage	POINTH	
0223	A9 1F		LDA,imm	S1F	
0225	80 41 17		STA,abs	PADD2	
0228	A9 FF		LDA,imm	SFF	
022A	80 03 17		STA,abs	PBDD1	
022D	A5 FA	VERIFY	LDA,Zpage	POINTL	
022F	29 1F		AND,imm	S1F	
0231	09 A0		ORA,imm	SA0	
0233	80 02 17		STA,abs	PBD1	
0236	A5 FA		LDA,Zpage	POINTL	
0238	85 FC		STA,Zpage	TEMP	
023A	A5 FB		LDA,Zpage	POINTH	
023C	85 FD		STA,Zpage	TMPX	
023E	A2 03		LDX,imm	S03	
0240	06 FC	SHIFT1	ASL,Zpage	TEMP	
0242	26 FD		ROL,Zpage	TMPX	
0244	CA		DEX,impl		
0245	D0 F9		BNE,rel	SHIFT1	
0247	A5 FD		LDA,Zpage	TMPX	
0249	80 40 17		STA,abs	PAD2	
024C	AD 00 17		LDA,abs	PAD1	
024F	01 FA		ORA,ind,X	POINTL	
0251	C0 00 17		CMP,abs	PAD1	
0254	F0 03		BEQ,rel	GOED	
0256	4C 4F 1C		JMP,abs	START	
0259	E6 FA	GOED	INC,Zpage	POINTL	
025B	D0 02		BNE,rel	GREST1	
025D	E6 FB		INC,Zpage	POINTH	
025F	38	GREST1	SEC,impl		
0260	A5 FB		LDA,Zpage	POINTH	
0262	ED 80 17		SBC,abs	PAG1	
0265	C9 04		CMP,imm	S04	
0267	D0 C4		BNE,rel	VERIFY	
0269	AD 80 17		LDA,abs	PAG1	
026C	85 FB		STA,Zpage	POINTH	
026E	AD 64		LDY,imm	S64	
0270	A9 FF		LDA,imm	SFF	
0272	80 01 17		STA,abs	PADD1	
0275	A5 FA	ST	LDA,Zpage	POINTL	
0277	29 1F		AND,imm	S1F	
0279	09 80		ORA,imm	S80	
027B	80 02 17		STA,abs	PBD1	
027E	A5 FA		LDA,Zpage	POINTL	
0280	85 FC		STA,Zpage	TEMP	
0282	A5 FB		LDA,Zpage	POINTH	
0284	85 FD		STA,Zpage	TMPX	
0286	A2 03		LDX,imm	S03	
0288	06 FC	SHIFT2	ASL,Zpage	TEMP	
028A	26 FD		ROL,Zpage	TMPX	
028C	CA		DEX,impl		
028D	D0 F9		BNE,rel	SHIFT2	
028F	A5 FD		LDA,Zpage	TMPX	
0291	80 40 17		STA,abs	PAD2	
0294	A1 FA		LDA,ind,X	POINTL	
0296	80 00 17		STA,abs	PAD1	
0299	EA		NOP,impl		
029A	EA		NOP,impl		
029B	AD 02 17		LDA,abs	PBD1	
029E	29 7F		AND,imm	S7F	
02A0	80 02 17		STA,abs	PBD1	
02A3	A2 C6		LDX,imm	S06	
02A5	CA	LOOP1	DEX,impl		
02A6	D0 FD		BNE,rel	LOOP1	
02A8	09 80		ORA,imm	S80	
02AA	80 02 17		STA,abs	PBD1	
02AD	E6 FA		INC,Zpage	POINTL	
02AF	D0 02		BNE,rel	GREST2	
02B1	E6 FB		INC,Zpage	POINTH	

02B3	38	GREST2	SEC,impl		
02B4	A5 FB		LDA,Zpage	POINTH	
02B6	ED 80 17		SBC,abs	PAG1	
02B8	C9 04		CMP,imm	S04	
02BB	D0 88		BNE,rel	ST	
02BD	AD 80 17		LDA,abs	PAG1	
02C0	85 FB		STA,Zpage	POINTH	
02C2	88		DEY,impl		
02C3	D0 80		BNE,rel	ST	
02C5	A9 00		LDA,imm	S00	
02C7	80 01 17		STA,abs	PADD1	
02CA	A5 FA	VERG2	LDA,Zpage	POINTL	
02CC	29 1F		AND,imm	S1F	
02CE	09 A0		ORA,imm	SA0	
02D0	80 02 17		STA,abs	PBD1	
02D3	A5 FA		LDA,Zpage	POINTL	
02D5	85 FC		STA,Zpage	TEMP	
02D7	A5 FB		LDA,Zpage	POINTH	
02D9	85 FD		STA,Zpage	TMPX	
02DB	A2 03		LDX,imm	S03	
02DD	06 FC	SHIFT3	ASL,Zpage	TEMP	
02DF	26 FD		ROL,Zpage	TMPX	
02E1	CA		DEX,impl		
02E2	D0 F9		BNE,rel	SHIFT3	
02E4	A5 FD		LDA,Zpage	TMPX	
02E6	80 40 17		STA,abs	PAD2	
02E9	AD 00 17		LDA,abs	PAD1	
02EC	C1 FA		CMP,ind,X	POINTL	
02EE	F0 1A		BEQ,rel	GOED2	
02F0	85 F9		STA,Zpage	INH	
02F2	20 88 1E		JSR,abs	INITS	
02F5	8E 81 17	LOOP3	STX,abs	PAG2	
02F8	20 1F 1F	AAN	JSR,abs	SCANDS	
02FB	EE 81 17		INC,abs	PAG2	
02FE	10 F8		BPL,rel	AAN	
0300	A0 80	UIT	LDY,imm	S80	
0302	88	LOOP2	DEY,impl		
0303	D0 FD		BNE,rel	LOOP2	
0306	E6		INX,impl		
0308	D0 F8		BNE,rel	UIT	
030B	F0 EB		BEQ,rel	LOOP3	
030A	E6 FA	GOED2	INC,Zpage	POINTL	
030C	D0 02		BNE,rel	GREST3	
030E	E6 FB		INC,Zpage	POINTH	
0310	38	GREST3	SEC,impl		
0311	A5 FB		LDA,Zpage	POINTH	
0313	ED 80 17		SBC,abs	PAG1	
0316	C9 04		CMP,imm	S04	
0318	D0 80		BNE,rel	VERG2	
031A	A9 00		LDA,imm	S00	
031C	80 03 17		STA,abs	PBDD1	
031F	4C 4F 1C		JMP,abs	START	

Lijst 2

0100	A9 00	START3	LDA,imm	S00	Zet het paginanummer op 0
0102	80 80 17		STA,abs	PAG1	
0105	F0 09		BEQ,rel	START4	Ga naar START4
0110	78	START4	SEL,impl		
0111	D8		CLD,impl		
0112	A9 00		LDA,imm	S00	
0114	80 01 17		STA,abs	PADD1	
0117	85 F9		STA,Zpage	INH	
0119	85 FA		STA,Zpage	POINTL	
011B	80 43 17		STA,abs	PBDD2	
011E	AD 80 17		LDA,abs	PAG1	
0121	85 FB		STA,Zpage	POINTH	
0123	A9 1F		LDA,imm	S1F	
0125	80 41 17		STA,abs	PADD2	
0128	A9 FF		LDA,imm	SFF	
012A	80 03 17		STA,abs	PBDD1	

012D	A5 FA	COPY	LDA,Zpage	POINTL	
012F	29 1F		AND,imm	\$1F	
0131	09 A0		ORA,imm	\$A0	
0133	8D 02 17		STA,abs	PB01	
0136	A5 FA		LDA,Zpage	POINTL	
0138	85 FC		STA,Zpage	TEMP	
013A	A5 FB		LDA,Zpage	POINTH	
013C	85 FD		STA,Zpage	TEMP	
013E	A2 03		LDX,imm	\$03	
0140	06 FC	SHIFT4	ASL,Zpage	TEMP	
0142	26 FD		ROL,Zpage	TEMP	
0144	CA		DEX,impl		
0145	D0 F9		BNE,rel	SHIFT4	
0147	A5 FD		LDA,Zpage	TEMP	
0149	8D 40 17		STA,abs	PAD2	
014C	AD 00 17		LDA,abs	PAD1	
014F	81 FA		STA,ind,X	POINTL	
0151	E6 FA		INC,Zpage	POINTL	
0153	D0 02		BNE,rel	GREST4	
0155	E6 FB		INC,Zpage	POINTH	
0157	38	GREST4	SEC,impl		
0158	A5 FB		LDA,Zpage	POINTH	
015A	ED 80 17		SBC,abs	PAG1	
015D	C9 04		CMP,imm	\$04	
015F	D0 CC		BNE,rel	COPY	
0161	A9 00		LDA,imm	\$00	
0163	8D 03 17		STA,abs	PB0D1	
0166	4C 4F 1C		JMP,abs	START	

Breng het adres naar PB0...PB4, en COLA...COLE.

Breng de inh. van de EPROM naar de RAM.

Hoog de adresssteller op.

Zolang het verschil tussen de adresssteller en het beginadres geen \$400 is, gaan we terug. PB0...PB5 en PB7 worden weer ingang.

Naar monitorprogramma.

Lijst 3

0000	A9 00	START5	LDA,imm	\$00	
0002	8D 88 17		STA,abs	BNIEMI	
0005	F0 0F		SEC,rel	VERDER	
0010	AD 81 17	START6	LDA,abs	PAG2	
0013	8D 88 17		STA,abs	BNIEMI	
0016	A9 00	VERDER	LDA,imm	\$00	
0018	8D 83 17		STA,abs	BOUDLO	
001B	8D 87 17		STA,abs	BNIEMI	
001E	A9 FF		LDA,imm	\$FF	
0020	8D 85 17		STA,abs	BOUDLO	
0023	AD 80 17		LDA,abs	PAG1	
0026	8D 84 17		STA,abs	BOUDHI	
0029	18		CLC,impl		
002A	D8		CLD,impl		
002B	89 03		ADC,imm	\$03	
002D	8D 86 17		STA,abs	BOUDHI	
0030	78	START7	SEI,impl		
0031	D8		CLD,impl		
0032	A2 00		LDX,imm	\$00	
0034	38		SEC,impl		
0035	AD 83 17		LDA,abs	BOUDLO	
0038	ED 87 17		SBC,abs	BNIEMI	
003B	AD 84 17		LDA,abs	BOUDHI	
003E	ED 88 17		SBC,abs	BNIEMI	
0041	90 3C		BCC,rel	SHUP	
0043	AD 83 17	SHDOWN	LDA,abs	BOUDLO	
0046	85 FA		STA,Zpage	POINTL	
0048	AD 84 17		LDA,abs	BOUDHI	
004B	85 FB		STA,Zpage	POINTH	
004D	AD 87 17		LDA,abs	BNIEMI	
0050	85 FC		STA,Zpage	TEMP	
0052	AD 88 17		LDA,abs	BNIEMI	
0055	85 FD		STA,Zpage	TEMP	
0057	A1 FA		LDA,ind,X	POINTL	
0059	81 FC		STA,ind,X	TEMP	
005B	EE 87 17		INC,abs	BNIEMI	
005E	D0 03		BNE,rel	GREST5	
0060	EE 88 17		INC,abs	BNIEMI	
0063	EE 83 17	GREST5	INC,abs	BOUDLO	
0066	D0 03		BNE,rel	GREST6	
0068	EE 84 17		INC,abs	BOUDHI	
006B	F0 0F	GREST6	BEQ,rel	MON	
006D	38		SEC,impl		
006E	AD 85 17		LDA,abs	BOUDLO	
0071	ED 83 17		SBC,abs	BOUDLO	
0074	AD 86 17		LDA,abs	BOUDHI	
0077	ED 84 17		SBC,impl	BOUDHI	
007A	B0 C7		BCS,rel	SHDOWN	

Zet het paginanummer waar het programma naar toe moet op 0.

Zet het paginanummer waar het programma naar toe moet zoals ingevoerd. Vul voor de laatste cijfers van het adres '00' in, en voor de laatste cijfers van het eindadres 'FF'.

Zet het oude beginadres zoals ingevoerd.

Bereken het oude eindadres, uitgaande van een lengte van 1k.

Trek het oude en nieuwe beginadres van elkaar af om te bepalen of omhoog dan wel naar beneden moet worden verplaatst.

Copieer de oude en nieuwe beginadressen.

Verplaats de inhoud van oud naar nieuw.

Hoog het nieuwe adres op.

Hoog het oude adres op.

Buiten geheugencapaciteit.

Indien het beginadres hoger is dan het eindadres, zijn we klaar.

Zo niet, terug.

007C	4C 4F 1C	MON	JMP,abs	START	Klaar.
007F	AD 85 17	SHUP	LDA,abs	BOUDLO	
0082	85 FA		STA,Zpage	POINTL	Copieer het oude eindadres.
0084	AD 86 17		LDA,abs	BOUDHI	
0087	85 FB		STA,Zpage	POINTH	
0089	38		SEC,impl		
008A	AD 85 17		LDA,abs	BOUDLO	
008D	ED 83 17		SBC,impl	BOUDLO	
0090	85 FC		STA,Zpage	TEMP	
0092	AD 86 17		LDA,abs	BOUDHI	
0095	ED 84 17		SBC,abs	BOUDHI	
0098	85 FD		STA,Zpage	TEMP	
009A	18		CLC,impl		Bereken het nieuwe eindadres.
009B	A5 FC		LDA,Zpage	TEMP	
009D	6D 87 17		ADC,abs	BNIEMI	
00A0	85 FC		STA,Zpage	TEMP	
00A2	A5 FD		LDA,Zpage	TEMP	
00A4	6D 88 17		ADC,abs	BNIEMI	
00A7	85 FD		STA,Zpage	TEMP	
00A9	A1 FA		LDA,ind,X	POINTL	Verplaats de inhoud van oud naar nieuw.
00AB	81 FC		STA,ind,X	TEMP	
00AD	AD 85 17		LDA,abs	BOUDLO	
00B0	D0 03		BNE,rel	GBOR	
00B2	CE 86 17		DEC,abs	BOUDHI	Verlaag het oude adres.
00B5	CE 85 17	GBOR	DEC,abs	BOUDLO	
00B8	38		SEC,impl		
00B9	AD 85 17		LDA,abs	BOUDLO	
00BC	ED 83 17		SBC,abs	BOUDLO	
00BF	AD 86 17		LDA,abs	BOUDHI	
00C2	ED 84 17		SBC,abs	BOUDHI	
00C5	90 0C		BCC,rel	MONI	
00C7	A9 FF		LDA,imm	\$FF	Naar het monitorprogramma.
00C9	CD 85 17		CMP,abs	BOUDLO	
00CC	D0 B1	SHUP1	BNE,rel	SHUP	
00CE	CD 86 17		CMP,abs	BOUDHI	Zijn we niet lager dan adres \$0000?
00D1	D0 F9		BNE,rel	SHUP1	
00D3	4C 4F 1C	MONI	JMP,abs	START	Zo ja, stop.

EPROM PROGRAMMEER-APPARAAT MET DE KIM (slot)

J. M. v.d. PEIJL

In het vorige deel werd de (zeer eenvoudige) interface besproken die nodig was om de programmeerpulsen aan de 2708 toe te voeren. Ook het programmeerprogramma, dat er voor zorgt dat de juiste pulsen op de juiste tijd komen, kwam in dat deel aan de orde.

In dit laatste deel gaan we nog even in op de extra benodigde voedingsspanningen, alsmede op enige uitvoeringsmogelijkheden.

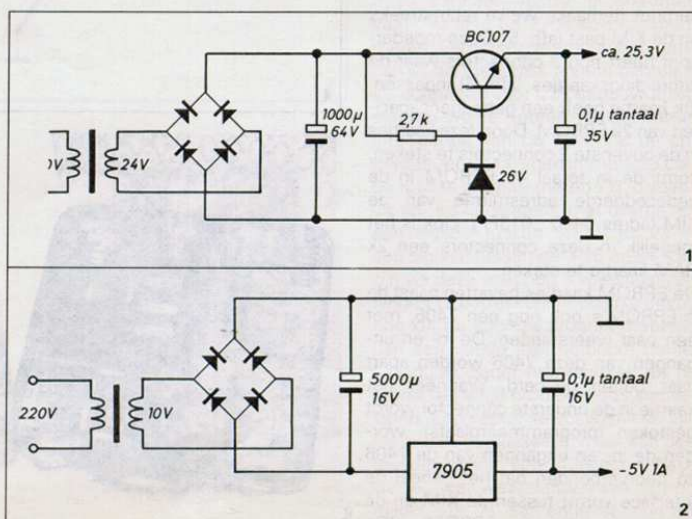
De voeding

De KIM maakt zelf gebruik van +12 V en +5 V. Voor normaal gebruik van de 2708 is alleen nog -5 V nodig. Wanneer we gaan programmeren moet bovendien een voedingsspanning van +26 V worden gemaakt. Veel stroom trekt de 2708 niet. De 26 V voeding hoeft maximaal slechts 20 mA te leveren. Als we echter toekomstige ontwikkelingen in het oog houden (de 2716, waar ze zeker op terug komen), moeten we de voeding geschikt maken voor minimaal 30 mA. In afb. 1 is een mogelijk schema van deze voeding gegeven. De extra -5 V voedingsspanning moet per 2708 ca. 45 mA kunnen leveren. In verband met toekomstige uitbreidingen is het echter verstandig de -5 V geschikt te maken voor minimaal 1 A. Dit kan het makkelijkst op de wel bekende manier volgens afb. 2. De +5 V en de +12 V voeding van de KIM zullen de extra

stroom voor de 2708 wel kunnen leveren (resp. 10 mA en 65 mA per 2708).

De trafo en de brugcel

Wanneer de voeding verzwaard moet worden, doordat u uw systeem gaat uitbreiden is de trafo meestal een probleem. Door de relatief hoge prijs van een trafo wordt zijn capaciteit maar al te vaak zo gekozen dat nét de verlangde stroom kan worden geleverd. Het gevolg is dat voor iedere uitbreiding opnieuw een trafo moet worden gekocht om de extra stroom/spanning te leveren. Bovendien moet voor elke trafo ook weer een plaatsje worden gevonden. In dit geval is opeens weer een spanning van -5 V en +26 V nodig, dus inderdaad wéér een trafo. Om voor eens en altijd met dit probleem af te rekenen moet u zorgen voor een trafo van zo'n 80 tot 120 VA. Het best kunt u dan een ringkern-trafo nemen (afb. 3). Deze trafo's zijn wel iets duur-



1. Het schema van de 26 V voeding. De extra benodigde wisselspanning kan ook verkregen worden door een aantal extra windingen om een ringkerntrafo te leggen. De uitgangsspanning is iets lager gekozen, zodat deze voeding ook voor de 2716 gebruikt kan worden.
2. Het schema van de -5 V voeding. Ook voor deze extra wisselspanning kan draad bijgewikkeld worden.

3. De ringkerntrafo, buiten de bekende pluspunten heeft hij het grote voordeel dat hij makkelijk van extra windingen kan worden voorzien.
4. Wanneer de 'ruwe' gelijkspanning een voldoende stroom kan leveren is een uitbreiding makkelijk te realiseren.
5. De KIM met moederprint vormt de basis van het systeem zoals de auteur die gemaakt heeft.
6. Het EPROM kaartje dat in de moederprint (afb. 5) gestoken kan worden.



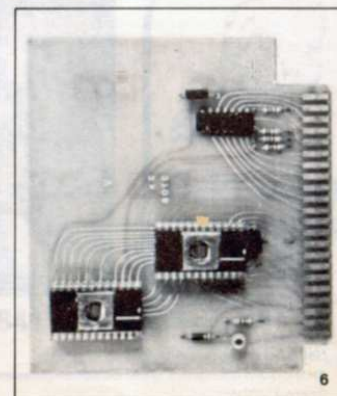
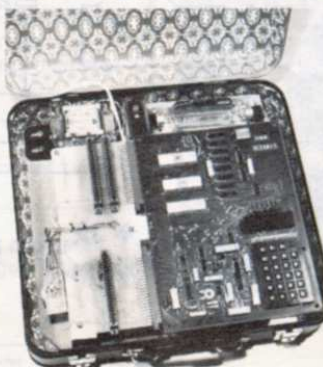
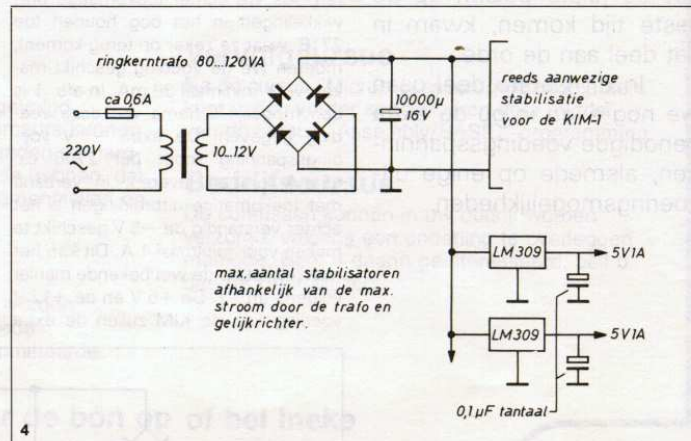
der, maar hebben buiten de bekende pluspunten (klein, lage verliezen, geen strooienveld) het grote voordeel dat ze makkelijk van extra windingen kunnen worden voorzien. Een onverwachte spanning (zoals nu +26 V) kost dan alleen een bosje wikkeldraad en wat tijd (ongeveer 240 windingen). De wikkeldruimte is haast onbeperkt, en u hoeft geen blik te demonteren. Wel moet u er op letten dat u niet boven het max. schijnbare vermogen uitkomt, maar met 80 of 120 VA kunt u voorlopig voort. Ook is het verstandig de brugcel voldoende zwaar te maken (voor de +5 V voeding ca. 5...10 A). Voor elke uitbreiding kan dan gewoon een 5 V stabilisator worden toegevoegd (b.v. LM309 of 7805). Een en ander is nog eens getekend in afb. 4.

Uitvoeringsmogelijkheden

De auteur heeft voor deze PROM-programmer een heel eigen systeem ontwikkeld. Als basis heeft hij een moederprint gemaakt, welke rechtstreeks op de KIM past (afb. 5). Deze moederprint heeft nog 3 connectors waar de uitbreidingskaartjes (afb. 6) inpassen. Elk kaartje heeft een geheugencapaciteit van 2k EPROM. Door deze kaartjes in de bovenste 2 connectors te steken, komt de in totaal 4k EPROM in de gedecodeerde adresruimte van de KIM (adres 0400...013FF). Ook is het mogelijk in deze connectors een 2k RAM kaartje te steken. De EPROM kaartjes bevatten naast de 2 EPROM's ook nog een 7406, met een paar weerstanden. De in- en uitgangen van deze 7406 worden apart naar buiten gevoerd. Wanneer het kaartje in de onderste connector wordt gestoken (programmeerplaats), worden de in- en uitgangen van de 7406 zó doorverbonden dat het geheel de interface vormt tussen de KIM en de te programmeren EPROM. (De

EPROM moet dan in de onderste voet op het kaartje zitten.) De adres- en data lijnen zijn nu met I/O van de KIM verbonden, en het programmeren kan beginnen. Wanneer het EPROM-kaartje normaal wordt gebruikt (in een van de twee bovenste connectors) blijft de 7406 niet ongebruikt, in dit geval worden de inverters zo geschakeld dat er een mo-

gelijkheid tot vector-selectie ontstaat (afb. 7). Hierdoor hebben we de mogelijkheid beide interruptvectors en de resetvector boven in een willekeurig 1k blok te leggen. Afb. 8 geeft tenslotte het eindresultaat: een compact kofertje, met daarin een compleet ontwikkelingssysteem.



7. De realisering van een 'vector select'.
8. Het systeem van de auteur. Boven in het koffertje kunnen EPROM's weer gewist worden.
9. De print op euro-card formaat, zoals RB hem maakte.

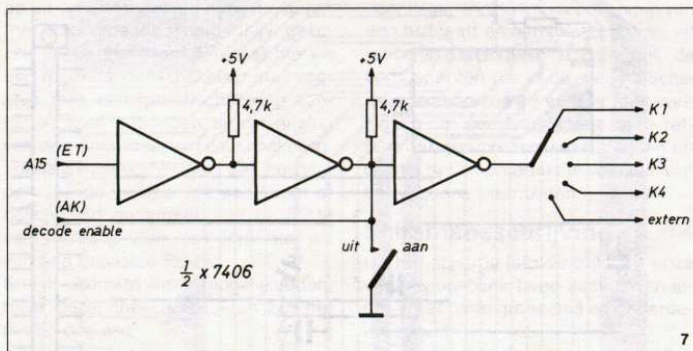
R.B. uitvoering

Hoewel bovenstaande beschrijving tot een vrij compleet geheel leidt, zijn er toch enkele bezwaren. We denken aan de ijverige hobbyist, die een mooie behuizing voor zijn KIM heeft gemaakt, en de moederprint hier onmogelijk in kwijt kan. Of de trouwe RB-lezer, die in de gedecodeerde 4k ruimte al 2 connectors heeft geplaatst voor de BEM-1 kaart (RB nov.'77). Daarbij komt dan nog dat de auteur door gebruik van dubbelzijdige contacten op de print, verplicht was ook dubbelzijdige print te

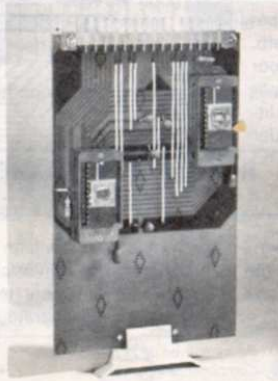
gebruiken. Nu is het zelf vervaardigen van dubbelzijdige print voor de meeste hobbyisten niet weggelegd. Ook wanneer een dergelijke print door ons verkocht zou worden, zou hij onnodig duur worden. Daarom ontwierp de RB redactie een enkelzijdige eurocard, met dezelfde connector, en dezelfde aansluitingen als de BEM-1 (afb. 9). Door het gebruik van enkelzijdige print werden enkele doorverbindingen noodzakelijk, voor de prijsbewuste hobbyist zal dit nauwelijks een bezwaar zijn. Overigens kan deze enkel-

zijdige print makkelijk dubbelzijdig uitgevoerd worden door de doorverbindingen op film te tekenen. De print layout kunt u in afb. 10 vinden, de bijbehorende componentenopstelling in afb. 11. De Muiderkring heeft een beperkt aantal van deze printen in voorraad, welke besteld kunnen worden door storting van f 17,50 per print op giro nr. 83214 onder vermelding van: expl. print nr. 7455.

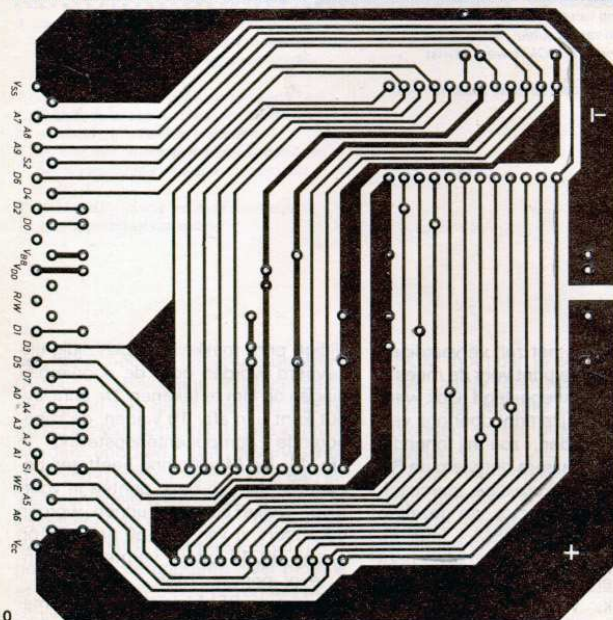
Doordat de BEM-kaarten gebruik maken van een 31-polige connector, terwijl de auteur een 40-polige connector gebruikte, was het niet mogelijk om ook de 7406 op deze print te zetten. Strikt genomen is dat ook helemaal niet nodig. De 7406 wordt gebruikt in de interface tussen de KIM en de te programmeren EPROM. Deze interface kan echter ook makkelijk op een stukje Veroboard worden gemaakt. Ergens in of op de behuizing van uw KIM is dan altijd wel een plek te vinden waar u een 24-polige I.C. voet kwijt kunt. U heeft dan een aparte 'programmeervoet', waar u de 2708 kunt programmeren. Deze programmeervoet kunt u dan in de toekomst tevens gebruiken als connector voor (zelf gebouwde) apparaten welke gebruik maken van de intelligentie van de KIM. Op de programmeervoet komen immers 18 door de KIM te programmeren TTL niveau's uit (in- en uitgangen), met daarbij nog één signaal tussen +12 V en 0V en een signaal tussen de +26 en 0V. Ook zijn via deze voet nog 3 voedingsspanningen te betrekken. In de luxe uitvoering kunt u voor die programmeervoet dan een voetje met hefboom nemen (vrij duur), zodat het insteken en het uittrekken altijd probleemloos kan verlopen. Een vector select is met dit systeem niet gerealiseerd, maar kan door toevoeging van één 7406, een schakelaar en wat weerstanden worden gemaakt (afb. 7). Samengevat is de RB-



8

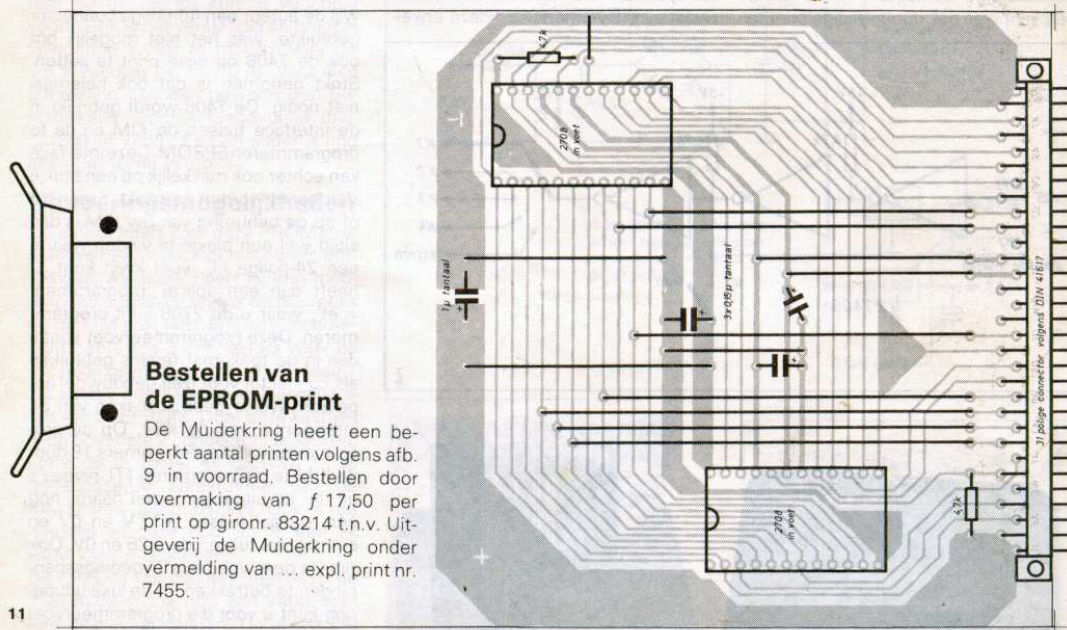


9



RB7L55
2k EPROM
(2x 2708)

10



11

Bestellen van de EPROM-print

De Muiderkring heeft een beperkt aantal printen volgens afb. 9 in voorraad. Bestellen door overmaking van f 17,50 per print op giro nr. 83214 t.n.v. Uitgeverij de Muiderkring onder vermelding van ... expl. print nr. 7455.

uitvoering van deze EPROM-programmer:

In de behuizing van de KIM bevinden zich 2 connectors. In elke connector kan of een BEM1 (2K RAM) of een EPROM kaart (2K EPROM) gestoken worden. Op de behuizing bevindt zich

een programmeervoet, waarin een 2708 geprogrammeerd kan worden. Deze programmeervoet dient tevens als connector voor randapparatuur.

10. De lay-out van de 2k EPROM print.

11. Componentenopstelling en doorverbindingen.