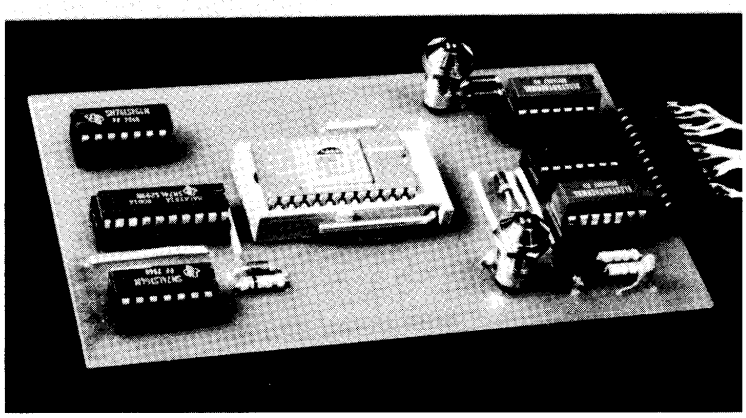
EPROM-PRO-GRAMMEER-APPARAAT



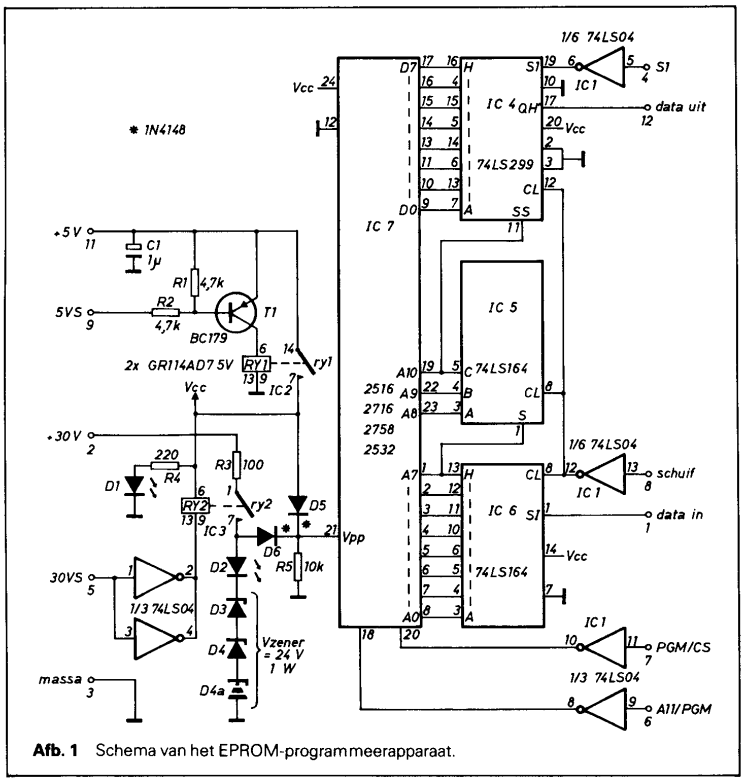
J.M.V.D.PEIJL

Dit programmeer-apparaat is ontworpen om aangesloten te kunnen worden op een groot aantal microcomputers. Het bestaat uit slechts één enkelzijdig printje. Deze print maakt gebruik van 8 in/uit-lijnen, één aardaansluiting, één 5 V aansluiting (300 mA) en één 30 V aansluiting (40 mA). Aan onderdelen kost de print ongeveer *f* 30,00.

We kunnen met dit apparaat de volgende EPROM's programmeren: de 2758 (1K), de 2716 (2K), de 2516 (2K Texas Instr.) en de 2532 (4K Texas Instr.). We maken hierbij geen onderscheid in de verschillende versies van deze IC's. Hoewel de software geschikt is om ook de 2732 te programmeren is hiervan afgezien omdat de 2732 iets andere pen-aansluitingen heeft. Het gaat hierbij vooral om de 25 V programmeerspanning. Als we de print geschikt willen maken voor dit type IC kunnen we het printontwerp gemakkelijk aanpassen. We moeten dan echter afzien van de mogelijkheid om de andere IC's erop te programmeren. De hier beschreven software is ontwikkeld voor de PET. Hoewel we ons EPROM-programmeerapparaat niet universeel kunnen noemen is het predikaat „veelzijdig" wel op zijn plaats.

Hardware

Voor de data- en adresaansluitingen van de EPROM is gebruik gemaakt van schuifregisters, zodat maar weinig aansluitingen naar de microprocessor nodig zijn (zie afb. 2). Met IC5 en IC6 worden de adreslijnen op de EPROM aangesloten. Het derde schuifregister is van een zeer universeel type. We maken hier slechts gebruik van twee mogelijkheden (modes) van het schuifregister. De ene mogelijkheid is „parallel load", voor het inlezen van de data vanuit de EPROM. De tweede mogelijkheid is het naar rechts schuiven van gegevens. Hierbij zijn de uitgangen actief. Deze uitgangen kennen namelijk drie standen, 0, 1 en ingang. We noemen



Afb. 1 Schema van het EPROM-programmeerapparaat.

dat „Tri-State" (3 toestanden). De klokingangen (schuif) van de drie schuifregisters zijn gebufferd om de microcomputer niet te zwaar te belasten. De pennen 18 en 20 van de EPROM (IC7) zijn om een andere reden gebufferd. We kunnen namelijk vanuit de microcomputer de voedingsspanningsaansluitingen van de EPROM besturen. Dit zijn de 5 V en de 25 V. De 25 V wordt op de print gerealiseerd door middel van enige zenerdioden (samen 24 V) en een LED in serie. We kunnen zo dan ook zien of de 25 V aanwezig is. Als de 5 V wordt uitgeschakeld krijgt ook de buffer, IC1 (74LS04), geen spanning meer. Alle aansluitingen van de EPROM zijn dan spanningloos. We kunnen nu zonder bezwaar de EPROM plaatsen of wegnemen. Als de 5 V is uitgeschakeld is automatisch de 25 V ook niet meer aanwezig omdat het reedrelais RY2 van de 25 V eveneens door de dubbele buffer wordt gestuurd. Na een reset zijn alle microcomputeraansluitingen hoog of zwevend. De 5 V is dan uitgeschakeld zoals uit afb. 1 blijkt. Het eigenlijke programmeren van een EPROM is zeer eenvoudig. We moeten eerst alle adres- en data-aansluitingen in de gewenste toestand brengen.
Tevens moet de CS (Chip Select) of OE (Output Enable) in de juiste stand worden gezet. De 25 V programmeerspanning moet zijn aangesloten. Vervolgens geven we een programmeerpuls (gewoon TTL-niveau). Deze puls moet ongeveer 50 ms duren. Dan is het desbetreffende adres geprogrammeerd. We kunnen zoveel adressen programmeren als we zelf willen. Uit de specificaties van de EPROM's kunnen we zien hoe de verschillende pennen moeten worden gestuurd. Deze waarden zijn verzameld in tabel 1. Hierbij is aangegeven welke spanningen door de microcomputer moeten worden geleverd. Data uit is hier niet aangegeven want dit is de uitgang van het schuifregister IC4 en dus een ingang voor de microcomputer. Zoals in tabel 1 is te zien moet bij het lezen van de EPROM's een klokpuls worden gegeven. Een eigenschap van de 74LS299 is namelijk dat deze bij een „parallel load" ook nog een klokpuls moet hebben.

Bouw

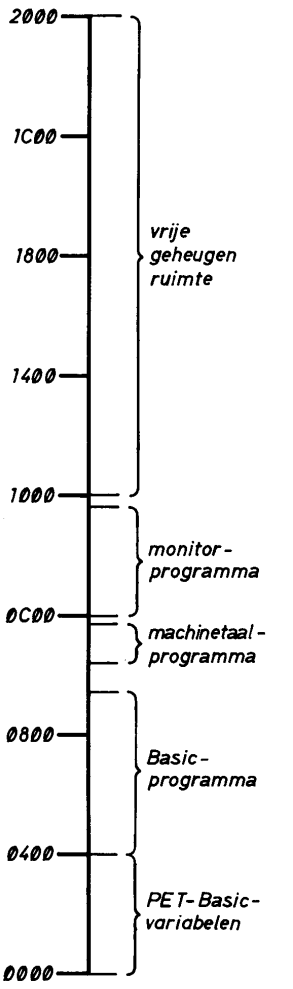
Hierover is weinig te zeggen. Het printontwerp (zie afb. 4 en 5) spreekt voor zichzelf. Voor de EPROM-voet kunnen we eventueel een gewoon voetje nemen. De IC's kunnen dan met een schroevendraaier en wat handigheid worden losgemaakt. Een IC-voet met een „pookje" is natuurlijk „het einde". In de handel is een dure van ongeveer *f* 50, maar ook een eenvoudige uitvoering van ongeveer *f* 15 bij Inelco verkrijgbaar (zie afb. 6).
Bij het plaatsen van de EPROM moet wel goed worden opgelet waar pen 1 is. Dit moet men op de print aangeven om vergissingen te voorkomen.

Software

De software is zo ontworpen dat deze goed past in het geheugen van de PET. Daarvoor moeten we eerst de geheugenruimte van de PET bekijken. De beschikbare RAM-ruimte begint bij adres Ø4ØØ en eindigt bij 1FFF Hex. De ruimte van 1ØØØ tot en met 1FFF willen we vrij houden om er de inhoud van de te programmeren of uit te lezen EPROM's te schrijven. Dit is een

Tabel 1 Overzicht van de stuursignalen voor diverse typen EPROM's.

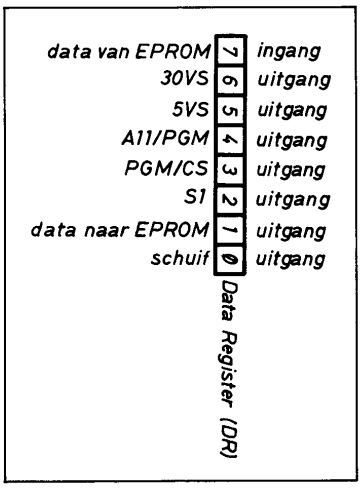
bit nr. / mode	7 data uit	6 30VS	5 5VS	4 A11 PGM	3 PGM CS	2 S1	1 data in	Ø schuif	
schuiven voor lezen		Ø	Ø	Ø	Ø	1	Ø	⎍‾‾‾⎍	2532
lezen		Ø	Ø	Ø	1	Ø	Ø	⎍	
schuiven voor programm.		1	Ø	1/Ø	1/Ø	1	Ø	⎍‾‾‾⎍	
programmeren		1	Ø	1/Ø	⎍ 50 ms	1	Ø	Ø	
schuiven voor lezen		Ø	Ø	Ø	Ø	1	Ø	⎍‾‾‾⎍	2758 2516 2716
lezen		Ø	Ø	1	1	Ø	Ø	⎍	
schuiven voor programm.		1	Ø	1	Ø	1	Ø	⎍‾‾‾⎍	
programmeren		1	Ø	⎍ 50 ms	Ø	1	Ø	Ø	



Afb. 2 Overzicht van de geheugenindeling voor de diverse programma's.

ruimte van 4K en daarin kan precies de inhoud van een 2532 staan. Het Basic-programma begint altijd automatisch op Ø4ØØ. Ons programma bestaat voor een deel uit Basic en voor een deel uit machinetaal. Verder maken we gebruik van het monitorprogramma, dat in het PET Users Manual staat. Dit programma is onmisbaar want daardoor kunnen we zien wat in de RAM staat van 1ØØØ tot en met 1FFF, en dus ook wat in de EPROM staat. We kunnen daar ook data in schrijven of veranderen. Het Basic-programma loopt tot circa Ø9ØØ. De pointers voor „Start of Variables" en „End of Variables" worden geïnitieerd op Ø98Ø. De pointers voor „Start of Strings" en „End of Memory" worden geïnitieerd op ØADF. Dit initiëren is nodig, omdat anders de programma's die boven het Basic-programma liggen worden overgeschreven of verschoven. Van ØAEØ tot ØBFD liggen de verschillende machinetaal-routines en van ØCØF tot ØF6D ligt het naar boven geschoven monitorprogramma (zie afb. 2). In lijst 1 vindt u een overzicht van het totale programma. We beginnen met het initiëren van de pointers en stellen daarmee de grenzen van ons Basic-programma vast. Wanneer we ons Basic-programma nog wat langer willen maken moeten we alle pointers die op Ø98Ø (hex) staan naar een hogere waarde brengen, bijvoorbeeld ØAØØ. Het is echter mogelijk dat we dan met een „Out of memory error" te maken krijgen. Vervolgens worden het Data Direction en het Data register ingeschreven.

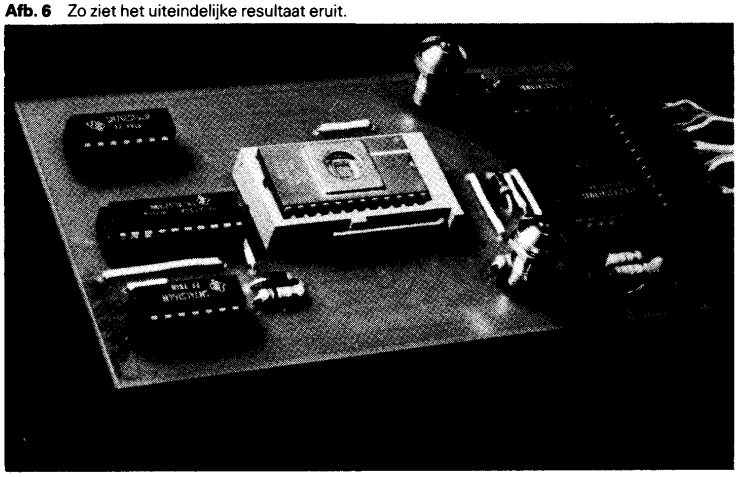
Het Data Direction register wordt op 7F gezet, omdat het belangrijkste bit (bit 7) een ingang is en de rest uitgangen. Zie hiervoor ook afb. 3. Vervolgens vraagt het programma of we naar het monitorprogramma willen. Als we daar naar toe gaan, kunnen we er weer uit komen door een „X" te typen. De rest van de monitor-instructies kunnen we in het PET Users Manual vinden. Als we niet naar de monitor gaan, moeten we aangeven of we willen uitlezen of programmeren en welk type EPROM we gebruiken. Vervolgens moeten we opgeven waar het begin- en eindadres ligt in de ROM en het beginadres in de RAM. Als de adressen minder dan 4 cijfers hebben worden ze automatisch aan de voorkant aangevuld met nullen (regel 1006). Het programma controleert vervolgens of we meer ruimte gebruiken dan in de EPROM aanwezig is. We worden dan gewaarschuwd met de tekst „zoveel ruimte heeft uw EPROM niet". Intussen is de 5 V ingeschakeld, zoals we aan de LED kunnen zien. Voor het programmeren en



Afb. 3 Indeling van de in- en uitgangsfuncties van het Data Register (DR) van de VIA in de PET.

uitlezen zijn de 2758, 2716 en 2732 aan elkaar gelijk. Er zijn dan ook vier verschillende programma's, lezen 2532, schrijven (programmeren) 2532, lezen 2658, 2716 en 2732 en schrijven van deze drie EPROM's. De regels 1ØØØ tot en met 1Ø5Ø vormen een routine, die de hexadecimale adressen vertaalt in decimale adressen. Deze subroutine wordt ook gebruikt door de subroutine die ligt op de regels 15Ø tot en met 19Ø. Deze laatste subroutine vertaalt de hexadecimale beginadressen van ROM en RAM, in groepen van 2 digits, naar decimaal en plaatst deze via een POKE-instructie op de adressen FB en FC voor het ROM-beginadres en op FD en FE voor het RAM-beginadres. Deze Zero Page-adressen worden niet door Basic gebruikt. Op regel 23Ø begint het programma dat dient om de 2532 te programmeren. Na het plaatsen van de beginadressen naar de Zero Page ko-

Afb. 6 Zo ziet het uiteindelijke resultaat eruit.



Lijst 1

```
  2 POKE 124, 128: POKE 125,9:
    POKE 126, 128: POKE 127,9:        zet pointers voor tekst, arrays en
    POKE 128, 128: POKE 129,9         variabelen voor Basic-programma
  3 POKE 130, 223: POKE 131, 1Ø:
    POKE 132, 223: POKE 133,1Ø:
    POKE 134, 223: POKE 135,1Ø
  5 POKE 59459, 127: POKE 59457,36    zet DDR en initialiseer DR
  6 PRINT "PLAATS EPROM"
  7 INPUT "TERUG NAAR MONITOR
    (J/N)"; Z$
  8 IF Z$ = "J" THEN SYS (3Ø87)       naar monitorprogramma
 1Ø INPUT "WELK TYPE EPROM"; A:       schakel 5V in en zet schuif mode
    POKE 59457,Ø4
 2Ø IF A< >2758 AND A< >2716 AND
    A< >2516 AND A< >2532
    GOTO 1Ø
 4Ø INPUT "LEZEN OF SCHRIJVEN
    (L/S)"; B$
 5Ø IF B$< >"L" AND B$< >"S" THEN
    4Ø
 7Ø INPUT "ROM-BEGINADRES"; C$
 75 INPUT "ROM-EINDADRES"; D$:       haal begin- en eindadres van het te
    INPUT "RAM-BEGINADRES"; E$       gebruiken ROM-geheugen en converteer
 8Ø F$ = C$: GOSUB 1ØØØ: C = H       dit naar hexadecimaal; hetzelfde voor het
 9Ø F$ = D$: GOSUB 1ØØØ: D = H:      beginadres van het RAM-geheugen.
    B = D – C                        bepaal of de ruimte overeenkomt met het
1 ØØ F$ = E$: GOSUB 1ØØØ: E = H      type EPROM
 11Ø IF B< 1Ø24 GOTO 19Ø
 12Ø IF B< 2Ø48 AND A< >2758 GOTO
    19Ø
 13Ø IF B< 4Ø96 AND A = 2532 GOTO 19Ø
 14Ø PRINT "ZOVEEL RUIMTE HEEFT
    UW EPROM NIET": GOTO 5
 15Ø F$ = C$: J = 2: K = 1: GOSUB 1ØØ5:  converteer het lage byte van het ROM-
    POKE 251, H                      beginadres naar decimaal; cônverteer het
 16Ø J = 4: K = 3: GOSUB 1ØØ5:       hoge byte en plaats deze in de pointer.
    POKE 252, H                      converteer het lage byte van het RAM-
 17Ø F$ = E$: J = 2: K = 1: GOSUB 1ØØ5:  beginadres naar decimaal; converteer het
    POKE 253, H                      hoge byte en plaats deze in de pointer.
 18Ø J = 4; K = 3: GOSUB 1ØØ5:
    POKE 254, H: RETURN
 19Ø IF A = 2532 AND B$ = "L" GOTO 4ØØ   bepaal wat gedaan moet worden en spring
 2ØØ IF A< >2532 AND B$ = "L" GOTO       naar de desbetreffende routine
    42Ø
 21Ø IF A< >2532 AND B$ = "S" GOTO
    32Ø
 22Ø GOSUB 15Ø: FOR L = ØTO B:       lees de 2532
    SYS(2882)                        ØB42 hex
 23Ø IF PEEK (1)< >255 GOTO 25Ø      en bepaal of deze laag is (FF),
 24Ø NEXT: GOTO 26Ø                  zo niet; print de tekst en terug
 25Ø PRINT "EPROM IS NIET            zo ja: ga verder
    SCHOON": GOTO 5
 26Ø POKE 59457, 68: GOSUB 15Ø       zet DR en programmeer de 2532; ØB9F.
 27Ø FOR L = ØTO B: SYS(2975); NEXT  hex
 28Ø POKE 59457,Ø4: GOSUB 15Ø       zet DR en test of het programma juist is
 285 FOR L = ØTO B: SYS (2882)       geschreven
 29Ø IF PEEK (1)< > PEEK (E + I) goto  ØB42 hex
    31Ø                              zo ja: ga terug
 3ØØ NEXT: GOTO 5                    zo nee: print tekst en ga terug
 31Ø PRINT "ADRES"; C + L; "IS FOUT":
    GOTO 5
 32Ø GOSUB 15Ø FOR L = Oto B:        controleer of de 2716, 2516, of 2758 schoon
    SYS (3Ø29)                       is; ØBD 5 hex
 33Ø IF PEEK(1)< >255 GOTO 25Ø:      zo nee: print tekst
    NEXT
 34Ø POKE 59457,84: GOSUB 15Ø        programmeer de 2516, 2716, 2758; ØAEØ
 35Ø FOR L = ØTO B: SYS(2784): NEXT  hex
 36Ø POKE 59457,Ø4: GOSUB 15Ø       en controleer of alles goed is geschreven;
 37Ø FOR L = ØTO B: SYS(3Ø29)        ØBD5 hex
 38Ø IF PEEK(1)< >PEEK(E + L) GOTO
    31Ø
 39Ø NEXT: GOTO 5
 4ØØ GOSUB 15Ø: FOR L = ØTO B:       lees de 2532 en ga terug
    SYS (2882)                       ØB42 hex
 41Ø Z = PEEK(1): POKE E + L,Z: NEXT:
    GOTO 5
 42Ø GOSUB 15Ø: FOR L = ØTO B:       lees de 2516, 2716, 2758 en ga terug;
    SYS (3Ø29)                       ØBD5 hex
 43Ø Z = PEEK(1): POKE E + L,Z: NEXT:
    GOTO 5

Subroutine hex naar decimaal.

1ØØØ J = 4: K = 1                    zet eerste en laatste karakter en initieer
1ØØ5 H = Ø                           H
1ØØ6 IF LEN(F$)< 4 THEN F$ = "Ø" + F$:  vul op met nullen
    GOTO 1ØØ6
1Ø1Ø FOR I = J TO K STEP – 1         haal het rechter karakter en converteer
1Ø2Ø G = ASC(RIGHT$(F$,I))           dit naar decimaal
1Ø3Ø IF G< 6Ø THEN G = G – 7
1Ø4Ø G = G – 48; H = H*16 + G: NEXT  ga één karakter naar links
1Ø5Ø RETURN                          keer terug, wanneer klaar
```

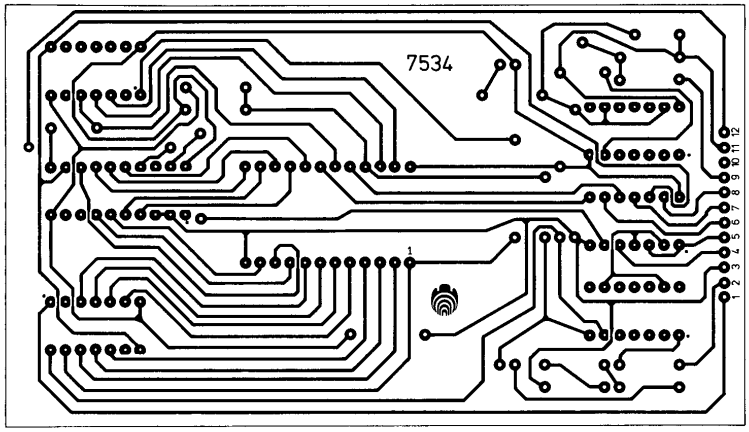
men we bij een lus die vergelijkt of het stuk ROM, dat we willen programmeren, wel schoon is (FF). We gebruiken hiervoor een machinetaalprogramma dat staat op adres ØB42, 2882 decimaal. Dit machinetaalprogramma begint met alle uitgangen op de juiste waarde te brengen. Zie ook lijst 2. Vervolgens komt een routine die begint op adres ØB8Ø. Deze routine plaatst de waarde van adreslijn A11 op de A11-aansluiting van de 2532 (pen 18). Daarna wordt het adres uit FB en FC via de schuifregisters naar binnen geschoven. Vervolgens komt een routine die de bits 3 en 2 van de in/uit-poort inverteert. Zie ook tabel 1. Er komt een klokpuls voor Parallel Load, waarna een routine volgt om de data uit de 74LS299 naar buiten te schuiven via ,,data uit''. Deze routine begint op adres ØB61. Omdat ,,data uit'' op het belangrijkste bit van de in/uit-poort staat kunnen we met een BPL-instructie controleren of dit 1 of Ø is. Omdat we de accumulator gebruiken om de ingelezen bits te roteren wordt het dataregister door middel van het Y-register ingelezen. Als laatste wordt de accumulator naar adres Ø1 hex geschreven. Dit adres is ook vrij, omdat het voor de instructie USR wordt gebruikt, terwijl we hier via SYS naar de machinetaalprogramma's springen. De laatste routine van ,,Read 2532'' is het ophogen van ROM- en RAM-adres. Als er in deze controleloop in het Basic-programma wordt ontdekt dat de inhoud van een geheugenplaats, die moet worden geprogrammeerd, niet op FF staat, springen we uit de FOR NEXT loop. Volgens het PET-boek is dat niet toegestaan maar ik heb tevergeefs gewacht op een strafmaatregel van mijn PET. Het eigenlijke programmeren kan nu beginnen (regel 26Ø). Met een POKE-instructie wordt de 25 V programmeerspanning ingeschakeld. De subroutine van regel 15Ø geeft het reedrelais RY2 de tijd om even na te ,,klapperen''. In de daarna komende FOR NEXT loop maken we gebruik van de machinetaalroutine op regel ØB9F. Deze routine lijkt veel op het begin van de leesroutine maar we schuiven hier eerst de data naar binnen en vervolgens zetten we bit A11 en schuiven het adres in. Bit 2 van het dataregister wordt nu geïnverteerd (zie afb. 2). We wachten vervolgens 50 ms. Bit 2 wordt weer geïnverteerd en ROM- en RAM-adres worden opgehoogd. Na het programmeren wordt de 25 V uitgeschakeld. We gaan de ROM nog een keer lezen en vergelijken met de RAM-inhoud (regel 28Ø tot en met 31Ø). Als er ongelijk wordt gevonden wordt het adres waar de fout is opgetreden, aangegeven (in decimale notatie). Op regel 32Ø tot 39Ø vinden we

Lijst 2

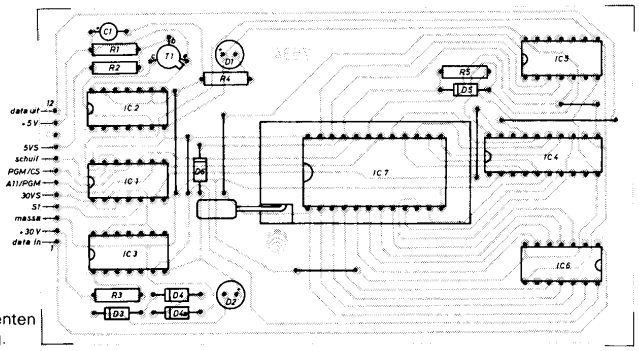
```
                        MICRO-WARE ASSEMBLER 65XX-1.0 PAGE 01
0010:                   **********************************
0020:                   *                                *
0030:                   *  EPROM PROGRAMMER              *
0040:                   *                                *
0050:                   *  COMPUTER BULLETIN HO80        *
0060:                   *                                *
0070:                   **********************************
0080:
0090:                   AUTEUR : J.M.V.D.PEIJL
0100:
0110: 0AE0                      ORG   $0AE0
0120:
0130:                   GEBRUIKTE VARIABELEN
0140:
0150: 0AE0              TEMPB  *      $0001
0160: 0AE0              ROMLOB *      $00FB
0170: 0AE0              ROMHIB *      $00FC
0180: 0AE0              RAMLOB *      $00FD
0190: 0AE0              RAMHIB *      $00FE
0200:
0210:                   VIA POORT PET
0220:
0230: 0AE0              DR     *      $E841
0240:
0250:                   SUBROUTINE VOOR HET PROGRAM-
0260:                   MEREN VAN DE 2516, 2716 EN 2758
0270:
0280: 0AE0 A9 54        PRZEST LDAIM $54     SCHAKEL 5 V IN ZET PRO-
0290: 0AE2 8D 41 E8            STA   DR      GRAMMA MODE 25 V AAN
0300: 0AE5 A2 00               LDXIM $00     MAAL INHOUD VAN RAM BEGIN
0310: 0AE7 A1 FD               LDAIX RAMLOB ADRES
0320: 0AE9 A2 08               LDXIM $08     ZET SCHUIFTELLER
0330: 0AEB A8                  TAY           EN
0340: 0AEC 20 1C 0B            JSR   SHFADR SCHUIF DATABITS D0-D7
0350: 0AEF 20 CC 0B            JSR   SHFZET SCHUIF ADRESBITS A0-A10
0360: 0AF2 A9 44               LDAIM $44     SCHAKEL PROGRAMMEER-
0370: 0AF4 8D 41 E8            STA   DR      PULS IN EN PROGRAMMEER
0380: 0AF7 20 C0 0B            JSR   WAITU  WACHT 50 MSEC
0390: 0AFA A9 54               LDAIM $54     SCHAKELPROGRAMMEER-
0400: 0AFC 8D 41 E8            STA   DR      PULS UIT
0410: 0AFF 20 00 0B            JSR   INCR    HOOG ROM EN RAM ADRES OP
0420: 0B02 60                  RTS           EN KEER TERUG
0430: 0B03 EA                  NOP
0440: 0B04 EA                  NOP
0450: 0B05 EA                  NOP
0460: 0B06 EA                  NOP
0470: 0B07 EA                  NOP
0480:
0490:                   SUBROUTINE OM ROM EN RAM
0500:                   ADRES OP TE HOGEN
0510:
0520: 0B08 E6 FB        INCR   INC   ROMLOB VERHOOG ROM ADRES
0530: 0B0A D0 02               BNE   LOOPA  PAGINA VOL?
0540: 0B0C E6 FC               INC   ROMHIB ZO JA; VOLGENDE PAGINA
0550: 0B0E E6 FD        LOOPA  INC   RAMLOB VERHOOG RAM ADRES
0560: 0B10 D0 02               BNE   RTSA    PAGINA VOL?
0570: 0B12 E6 FE               INC   RAMHIB ZO JA; VOLGENDE PAGINA
0580: 0B14 60           RTSA   RTS           EN TERUG
0590:
0600:                   SUBROUTINE OM VIER OF VIJF MAAL
0610:                   TE SCHUIVEN
0620:
0630: 0B15 0A           SHFV   ASLA          SCHUIF TOTAAL VIJF MAAL
0640: 0B16 0A           SHFF   ASLA          SCHUIF TOTAAL VIER MAAL
0650: 0B17 0A                  ASLA
0660: 0B18 0A                  ASLA
0670: 0B19 0A                  ASLA
0680: 0B1A A8                  TAY           BEWAAR ACCU IN Y
0690: 0B1B 60                  RTS           EN KEER TERUG
0700:
0710:                   SUBROUTINE VOOR HET SCHUIVEN
0720:                   VAN HET ADRES NAAR DE UITGANG
0730:
0740: 0B1C 98           SHFADR TYA           ADRES NAAR ACCU
0750: 0B1D 0A                  ASLA          SCHUIF LINKS
0760: 0B1E A8                  TAY           ACCU NAAR Y
0770: 0B1F 90 08               BCC   LOOPB  WAS BIT 3 NUL?
0780: 0B21 AD 41 E8            LDA   DR      ZO NEE; ZET DATA OP 1
0790: 0B24 09 02               ORAIM $02
0800: 0B26 8D 41 E8            STA   DR        EN
0810: 0B29 20 5A 0B     LOOPB  JSR   CLOCK   GEEF KLOKPULS
0820: 0B2C AD 41 E8            LDA   DR      CLEAR DATA
0830: 0B2F 29 FD               ANDIM $FD
0840: 0B31 8D 41 E8            STA   DR
0850: 0B34 CA                  DEX
0860: 0B35 D0 E5               BNE   SHFADR ALLE BITS GEHAD?
0870: 0B37 60                  RTS           DAN TERUG
0020:
0030:                   SUBROUTINE OM DE 2532 TE LEZEN
0040:
0050: 0B42 A9 04        RETWED LDAIM $04     ZET 5 V AAN EN
0060: 0B44 8D 41 E8            STA   DR      MAAK LINKS SCHUIVEN GEREED
0070: 0B47 20 80 0B            JSR   BITADR SCHUIF A0-A10 EN ZET A11
0080: 0B4A 20 76 0B            JSR   EXORC  ENABLE PARALLEL LOAD
0090: 0B4D 20 5A 0B            JSR   CLOCK   GEEF KLOKPULS
0100: 0B50 20 76 0B            JSR   EXORC  DISABLE PRAL. LOAD
0110: 0B53 20 61 0B            JSR   SHFDAT SCHUIF DATA UIT
0120: 0B56 20 00 0B            JSR   INCR    HOOG ROM EN RAM ADRES OP
0130: 0B59 60                  RTS
0140:
0150:                   SUBROUTINE OM KLOKPULS TE MAKEN
0160:                   VOOR HET SCHUIVEN EN PARALLEL LADEN
0170:
0180: 0B5A EE 41 E8     CLOCK  INC   DR
0190: 0B5D CE 41 E8            DEC   DR
0200: 0B60 60                  RTS
0210:
0220:                   SUBROUTINE OM ACHT DATABITS NAAR
0230:                   BUITEN TE SCHUIVEN
0240:
0250: 0B61 A2 08        SHFDAT LDXIM $08     TOTAAL 8 BITS
0260: 0B63 AC 41 E8     LOOPD  LDY   DR
0270: 0B66 18                  CLC
0280: 0B67 10 01               BPL   LOOPC  IS BIT 7 EEN "0"
0290: 0B69 38                  SEC           NEEN
0300: 0B6A 2A           LOOPC  ROLA          JA; ROTEER ACCU
0310: 0B6B 20 5A 0B            JSR   CLOCK   GEEF KLOKPULS
0320: 0B6E CA                  DEX           8 MAAL GEWEEST?
0330: 0B6F D0 F2               BNE   LOOPD  NEEN; GA TERUG
0340: 0B71 85 01               STA   TEMPB  JA; DATA IN ADRES 01
0350: 0B73 60                  RTS
0360: 0B74 EA                  NOP
0370: 0B75 EA                  NOP
0380:
0390:                   SUBROUTINE OM HET SCHUIFREGISTER
0400:                   TE BESTUREN
0410:
0420: 0B76 AD 41 E8     EXORC  LDA   DR
0430: 0B79 49 0C               EORIM $0C     INVERTEER BIT 2 EN 3
0440: 0B7B 8D 41 E8            STA   DR
0450: 0B7E 60                  RTS
0460: 0B7F EA                  NOP
0470:
0480:                   SUBROUTINE OM ADRESBITS A0-A10
0490:                   TE SCHUIVEN EN A11 TE ZETTEN
0500:
0510: 0B80 A5 FC        BITADR LDA   ROMHIB
0520: 0B82 20 16 0B            JSR   SHFF    SCHUIF 4 MAAL LINKS
0530: 0B85 B0 08               BCS   ADRES   IS BIT 4 (A11) HOOG?
0540: 0B87 AD 41 E8            LDA   DR      NEEN;
0550: 0B8A 09 10               ORAIM $10     MAAK A11 = "0"
0560: 0B8C 8D 41 E8            STA   DR
0570: 0B8F A2 03        ADRES  LDXIM $03     JA; A11 = "1"
0580: 0B91 20 1C 0B            JSR   SHFADR SCHUIF ADRESBITS A8-A10
0590: 0B94 A4 FB               LDY   ROMLOB
0600: 0B96 A2 08               LDXIM $08
0610: 0B98 20 1C 0B            JSR   SHFADR SCHUIF ADRESBITS A0-A7
0620: 0B9B 60                  RTS
0630: 0B9C EA                  NOP
0640: 0B9D EA                  NOP
0650: 0B9E EA                  NOP
0660:
0670:                   SUBROUTINE OM DE 2532 TE
0680:                   PROGRAMMEREN
0690:
0700: 0B9F A9 44        PRTWED LDAIM $44     PRESET SCHUIFREGISTER
0710: 0BA1 8D 41 E8            STA   DR
0720: 0BA4 A2 00               LDXIM $00
0730: 0BA6 A1 FD               LDAIX RAMLOB HAAL INHOUD RAM
0740: 0BA8 A2 08               LDXIM $08     8 BITS TE SCHUIVEN
0750: 0BAA A8                  TAY           BEWAAR ACCU IN Y REGISTER
0760: 0BAB 20 1C 0B            JSR   SHFADR SCHUIF D0-D7 BINNEN
0770: 0BAE 20 80 0B            JSR   BITADR SCHUIF A0-A10 EN ZET A11
0780: 0BB1 20 F7 0B            JSR   EXORA  PROGRAMMEERPULS AAN
0790: 0BB4 20 C0 0B            JSR   WAITU  WACHT 50 MSEC
0800: 0BB7 20 F7 0B            JSR   EXORA  PROGRAMMEERPULS UIT
0810: 0BBA 20 00 0B            JSR   INCR    HOOG ROM EN RAM ADRES OP
0820: 0BBD 60                  RTS
0830: 0BBE EA                  NOP
0840: 0BBF EA                  NOP
0020:
0030:                   SUBROUTINE OM 50 MSEC TE WACHTEN
0040:
0050: 0BC0 A0 28        WAITU  LDYIM $28
0060: 0BC2 A2 FA        LOOPE  LDXIM $FA
0070: 0BC4 CA           LOOPF  DEX
0080: 0BC5 D0 FD               BNE   LOOPF  WACHT 1,25 MSEC
0090: 0BC7 88                  DEY
0100: 0BC8 D0 F8               BNE   LOOPE  WACHT 40 X 1,25 = 50 MSEC
0110: 0BCA 60                  RTS
0120: 0BCB EA                  NOP
0130:
0140:                   ROUTINE OM VIJF ADRESBITS
0150:                   TE SCHUIVEN
0160:
0170: 0BCC A5 FC        SHFZET LDA   ROMHIB
0180: 0BCE 20 15 0B            JSR   SHFV
0190: 0BD1 4C 8F 0B            JMP   ADRES
0200: 0BD4 EA                  NOP
0210:
0220:                   SUBROUTINE OM DE 2516, 2716 EN
0230:                   2758 TE LEZEN
0240:
0250: 0BD5 A9 04        REZEST LDAIM $04
0260: 0BD7 8D 41 E8            STA   DR
0270: 0BDA 20 CC 0B            JSR   SHFZET SCHUIF ADRESBITS IN
0280: 0BDD 20 EE 0B            JSR   EXORIC ENABLE
0290: 0BE0 20 5A 0B            JSR   CLOCK   GEEF KLOKPULS
0300: 0BE3 20 EE 0B            JSR   EXORIC DISABLE
0310: 0BE6 20 61 0B            JSR   SHFDAT SCHUIF DATABITS UIT
0320: 0BE9 20 00 0B            JSR   INCR    HOOG ROM EN RAM ADRES OP
0330: 0BEC 60                  RTS
0340: 0BED EA                  NOP
0350:
0360:
0360:                   SUBROUTINE TER VOORBEREIDING
0370:                   OP LEZEN
0380:
0390: 0BEE AD 41 E8     EXORIC LDA   DR
0400: 0BF1 49 1C               EORIM $1C     INVERTEER BIT 2, 3 EN 4
0410: 0BF3 8D 41 E8            STA   DR
0420: 0BF6 60                  RTS
0430:
0440:                   SUBROUTINE TER VOORBEREIDING
0450:                   OP PROGRAMMEREN
0460:
0470: 0BF7 AD 41 E8     EXORA  LDA   DR
0480: 0BFA 49 08               EORIM $08     INVERTEER BIT 3
0490: 0BFC 8D 41 E8            STA   DR
0500: 0BFF 60                  RTS

SYMBOL TABLE 3000 30B4
ADRES  0B8F    BITADR 0B80    CLOCK  0B5A    DR     E841
EXORA  0BF7    EXORC  0B76    EXORIC 0BEE    INCR   0B08
LOOPA  0B0E    LOOPB  0B29    LOOPC  0B6A    LOOPD  0B63
LOOPE  0BC2    LOOPF  0BC4    PRTWED 0B9F    PRZEST 0AE0
RAMHIB 00FE    RAMLOB 00FD    RETWED 0B42    REZEST 0BD5
ROMHIB 00FC    ROMLOB 00FB    RTSA   0B14    SHFADR 0B1C
SHFDAT 0B61    SHFF   0B16    SHFV   0B15    SHFZET 0BCC
TEMPB  0001    WAITU  0BC0
```

het programmeerprogramma voor de andere EPROM's. Op regel 400 staat een programma om van EPROM 2532 naar RAM te schrijven (pas op dat u uw programmeerprogramma niet overschrijft!). Op regel 420 staat een leesprogramma voor de andere EPROM's ■



Afb. 4 Printontwerp van het EPROM-programmeerapparaat.



Afb. 5
Componenten opstelling.

Hier volgt een vereenvoudigde versie van het voorgaande programma, afkomstig van de heer Van de Peijl, dat was bedoeld voor de CBM/PET. Dit is meer geschikt voor kleine systemen met een 6502, zoals de KIM, Junior, SYM, AIM enz. De machinetaalroutines zijn vrijwel ongewijzigd overgenomen, doch het Basic-gedeelte is vervangen door eveneens een stukje machinetaal. Hiermee kunnen de 2516 en 2716 (2 Kbytes EPROM's) worden gelezen, geprogrammeerd en geverifieerd. Het programma beslaat zelf nog geen half Kbytes, maar toch bezit een standaard uitvoering van een enkelkaartcomputer veelal niet voldoende geheugen om alle taken te kunnen verrichten. Er is minstens 3 Kbytes RAM nodig, waaronder 2 Kbytes voor de buffer. Na wat handwerk bij het laden van een aantal registers kan het geheel op een EPROM worden losgelaten.

EPROM-PRO-GRAMMEER-PROGRAMMA VOOR KLEINE 6502-SYSTEMEN

P.G.J.DE BEER

Hardware

Het EPROM-programmeerapparaat is geheel dezelfde gebleven. De aansluitingen zijn op dezelfde wijze gemaakt, maar moeten nu worden aangesloten op de A-poort van uw eigen PIA, zie de tabel. Het enige probleem zal meestal de voedingsspanning van 30 V zijn. Wanneer u uw computer voedt met behulp van een ringkerntrafo is de oplossing gemakkelijk. Een aantal wikkelingen extra tot een openklemspanning van ongeveer 24 V met daarachter een brugcel en een afvlakelco leveren het gewenste resultaat. Anders zal er een trafo bij moeten komen met een secundaire spanning van 24 V. De voedingsspanning van 5 V kan meestal van de computer zelf worden betrokken. De massa-aansluiting dient deugdelijk te zijn om problemen met lussen en dergelijke te voorkomen.

Routines

Gekozen is voor het programmeren en lezen van de 2516 van Texas Instruments en de 2716, geleverd door diverse fabrikanten. Deze EPROM's zijn op het moment zeer gangbaar vanwege hun prijs-prestatie. Er kan 2 Kbytes in worden opgeslagen en dat is niet mis. Ze hebben als volgende voordeel, dat ze slechts één voedingsspanning van 5 V nodig hebben, in plaats van drie. Van de machinetaalroutines zijn dan ook alleen diegenen overgenomen, die voor deze EPROM's van belang zijn, zie de lijst. Het programma start op adres Ø2ØØ hex en begint met een initialiseringsroutine. Deze zet de registers van de betreffende PIA goed. Ingebouwd is ,,SAVTMP'', dat de beginwaarden voor start- en eindadres van het RAM- en EPROM-geheugenblok copieert naar andere locaties op de zero-page, waar ze kunnen worden gebruikt voor ophoging. Het startadres van het geheugenblok in EPROM wordt gehaald van de locaties ØØEØ en ØØE1, het startadres van het blok in RAM van ØØE2 en ØØE3. Deze worden geplaatst op respectievelijk de locaties ØØE6 en ØØE7, en ØØE8 en ØØE9. Het eindadres van het blok in EPROM moet staan op de locaties ØØE4 en ØØE5. Hiermee wordt, tijdens de loop van het programma, vergeleken of alles klaar is; niet met het RAM-eindadres dus. Telkens wanneer het programma een bepaalde functie heeft verricht, worden de startadressen hersteld door middel van de routine ,,SAVTMP''

Na de initialisatie komen de routines voor het programmeren, schuiven, 50 ms wachten, lezen en ophogen van het EPROM- en RAM-adres. Er is één verschil met de oorspronkelijke publicatie. Daar wordt het aantal af te werken adressen bijgehouden door het Basic-programma. Het adres, waarin op een bepaald moment wordt gelezen of geschreven, wordt onafhankelijk bepaald in het machinetaalgedeelte. In deze vereenvoudigde versie zijn beide functies gecombineerd. Dat houdt in dat de incrementeerfunctie is verdwenen uit de lees- en schrijfroutines en een plaats heeft gevonden in het hoofdprogramma. Er is een extra subroutine, ,,EADR'', toegevoegd, die controleert of het laatste adres van de EPROM al is bereikt. Verder is er een uitleesroutine, die een aantal foutmeldingen mogelijk maakt. Het maakt gebruik van de zescijferige uitlezing op, in dit geval, de KIM. Wat wordt weergegeven, wordt bepaald door de pointer op locatie ØØEB van de zero-page. Eerst worden de registers van de betreffende PIA goed gezet en wordt ØØ in een teller geladen. Dan wordt het eerste display

Tabel 1 Aansluitgegevens voor zowel de KIM als de Junior.

EPROM-programmeer-apparaat		COMPUTER		
		KIM	Junior	
signaal	**pen**	**pen**	**pen**	**signaal**
schuif	8	A-14	3	PAØ
data naar EPROM	1	A-4	4	PA1
S1	4	A-3	5	PA2
PGM/CS	7	A-2	6	PA3
A11/PGM	6	A-5	7	PA4
5 V S	9	A-6	8	PA5
30 V S	5	A-7	9	PA6
data van EPROM	12	A-8	10	PA7
massa	3	A-1	1	massa
+ 5 V	11	A-A	2, 17	+ 5 V
+ 30 V	2			+ 30 V

geselecteerd en wordt er door middel van de pointer een code uit de tabel gehaald. Deze licht op, waarna er even wordt gewacht. Het volgende display komt aan de beurt, waarin de volgende code wordt geplaatst. Dit gebeurt totaal 6 maal. Na een extra wachttijd wordt deze cyclus opnieuw herhaald, totdat de in het begin genoemde teller weer nul is geworden.

Hoofdprogramma

Het hoofdprogramma bestaat uit drie delen. Eerst is er het programmeergedeelte met daaraan gekoppeld een verifieergedeelte. Dit laatste kan onafhankelijk worden uitgevoerd door op het betreffende adres te starten. Als laatste is er een leesgedeelte.

Het programmeergedeelte begint met de initialisatie, haalt dan de pointer voor het uitlezen van een bericht en plaatst deze in een locatie op de zeropage. De voedingsspanning van 5 V wordt ingeschakeld en de LED gaat branden. Er wordt even gewacht om het relais uit te laten denderen. Vervolgens wordt een leesactie gestart om te kijken of de EPROM, voor wat betreft de gevraagde geheugenlocaties, schoon is. Wordt op een bepaalde plaats geen FF gelezen, dan springt het programma naar de uitleesroutine en laat aldaar „GEEN FF" op het display verschijnen. Is alles in orde dan worden de startadressen hersteld, de programmeerspanning van 30 V wordt ingeschakeld en de tweede LED gaat branden. Het programmeren kan beginnen. Wanneer het eindadres van het geheugenblok in EPROM nog niet is bereikt, worden EPROM- en RAM-adres opgehoogd en kan de volgende byte worden geschreven. Zijn alle bytes geweest, dan springt het programma naar het verifieergedeelte.

De startadressen worden hersteld en de volgende pointer voor een bericht wordt opgehaald. Er wordt een byte uit de EPROM gelezen en deze wordt vergeleken met de data uit de overeenkomende RAM-locatie. Zijn deze gegevens aan elkaar gelijk dan worden de adressen opgehoogd en wordt er opnieuw vergeleken. Wordt ergens een fout geconstateerd dan springt het programma naar de uitleesroutine en laat „FOUTJE" zien. Hierna wordt naar de monitor teruggesprongen. Zijn RAM en EPROM elkaars spiegelbeeld, dan wordt een nieuwe pointer gehaald en zal het bericht „EINDE" verschijnen. Automatisch springt het programma ook hier terug naar de monitor.

Het leesgedeelte begint weer met een initialisatie, waarna, uitgaande van de beide startadressen, de bytes vanuit de EPROM worden overgebracht naar RAM. Als alles klaar is, verschijnt eveneens „EINDE" op het display en springt het programma na een bepaalde tijd naar de monitor terug.

Gebruik

Wanneer het programma en de print voor de eerste maal worden gebruikt, is het aan te raden wat testen uit te voeren.

Eerst worden op de print zelf de voedingsspanningen van 5 V en 30 V gecontroleerd. Dan wordt op de adressen ØØEØ en ØØE1 met behulp van de monitor tweemaal achtereen ØØ ingebracht. Op de adressen ØØE2 en ØØE3 worden respectievelijk de lage en de hoge byte van het adres gezet, van waaraf men het RAM-geheugen wil gebruiken. Vervolgens worden de lage en hoge byte van het eindadres van het EPROM-geheugen geladen in de locaties ØØE4 en ØØE5. Nu wordt het programma vanaf het label „LEES" gestart, zonder dat er zich een EPROM in de houder bevindt. Alleen de 5V-LED gaat branden. Het resultaat is, dat het betreffende RAM-geheugenbereik wordt gevuld met FF. Dit kan, na het verschijnen van „EINDE" op het display, door middel van de monitor worden gecontroleerd. Vervolgens wordt bij het label „VERF" gestart voor een verificatie. Dit moet foutloos eindigen. Wanneer we nu echter in RAM een byte veranderen door een andere waarde en het programma „VERF" opnieuw starten, moet het bericht „FOUTJE" verschijnen. Door middel van de monitor kan op de adressen ØØE6 en ØØE7 worden gevonden op welk adres in de EPROM de fout is opgetreden. Als dit allemaal in orde is bevonden, kan een liefst voorgeprogrammeerde EPROM in de houder worden geplaatst. Denk erom dat u dit altijd pas dan doet, als de voedingsspanningen van de computer zijn ingeschakeld en er een reset is gegeven. Het programma wordt wederom vanaf het label „LEES" gestart. Na „EINDE" wordt met de monitor gekeken of de data overeenkomt met de verwachtingen. Door „VERF" te starten kunt u nagaan of er niets is fout gegaan. Vervolgens starten we vanaf label „PROG" om te gaan programmeren. Dit zal direct een foutmelding opleveren, daar de EPROM niet schoon was: „GEEN FF". We nemen nu een schone EPROM, plaatsen deze in de houder, vullen adres ØØEØ en ØØE1 beide met ØØ, adres ØØE4 met bijvoorbeeld Ø7 en ØØE5 met ØØ. Op de adressen ØØE2 en ØØE3 wordt het door u gewenste RAM-startadres gezet. Na starten bij „PROG' worden deze acht bytes geprogrammeerd. De 5V-LED gaat aan. Na even wachten gaat de 30V-LED even aan, dan weer uit en na nog even wachten gaat ook de 5V-LED weer uit. In de laatste periode vond de verificatie plaats. Als het bericht „EINDE" verschijnt is alles goed verlopen. Komt het bericht „FOUTJE", dan kan op de adressen ØØE6 en ØØE7 worden gevonden, waar de fout optrad.

Is echter alles goed, dan kunt u een willekeurig programma in EPROM zetten. Voor de volledige 2 Kbytes duurt dat wel een minuutje, maar dan bezit u ook zelfgemaakte firmware. ■

Lijst Programma om EPROM's te programmeren op kleine 6502-systemen.

```
        ASS.6502 COMPUTER BULLETIN DB PAGE 01


0010:                   ****************************************
0020:                   *                                      *
0030:                   *       EPROM PROGRAMMEERAPPARAAT      *
0040:                   *       VOOR DE KIM MET 12K RAM        *
0050:                   *       COMPUTER BULLETIN PDB*81       *
0060:                   *                                      *
0070:                   ****************************************
0080:
0090:                   AUTEUR : J.M. V.D.PEIJL
0100:                       EN : P.G.J. DE BEER
0110:
0120: 0200                      ORG   $0200
0130:
0140:                   GEBRUIKTE VARIABELEN
0150:
0160: 0200              ROMLO   *     $00E0
0170: 0200              ROMHI   *     $00E1
0180: 0200              RAMLO   *     $00E2
0190: 0200              RAMHI   *     $00E3
0200: 0200              EROMLO  *     $00E4
0210: 0200              EROMHI  *     $00E5
0220: 0200              TMPROL  *     $00E6
0230: 0200              TMPROH  *     $00E7
0240: 0200              TMPRAL  *     $00E8
0250: 0200              TMPRAH  *     $00E9
0260: 0200              TEMPA   *     $00EA
0270: 0200              TEMPB   *     $00EB
0280: 0200              TEMP    *     $00FC
0290: 0200              TMPX    *     $00FD
0300:
0310:                   PIA POORT KIM
0320:
0330: 0200              DR      *     $1700
0340: 0200              DDR     *     $1701
0350:
0360:                   KIM ROUTINE
0370:
0380: 0200              MON     *     $1C4F
0390:
0400: 0200 A9 7F        INIT    LDAIM $7F     ZET DATARICHTINGS-
0410: 0202 8D 01 17             STA   DDR     REGISTER
0420: 0205 A9 24                LDAIM $24     REZET DATAREGISTER
0430: 0207 8D 00 17             STA   DR
0440: 020A A2 03        SAVTMP  LDXIM $03
0450: 020C B5 E0        SATP    LDAZX ROMLO   BEWAAR START-
0460: 020E 95 E6                STAZX TMPROL  ADRESSEN
0470: 0210 CA                   DEX
0480: 0211 10 F9                BPL   SATP    ALLES GEHAD?
0490: 0213 60                   RTS           JA,TERUG
0500:
0510:                   SUBROUTINE VOOR HET PROGRAM-
0520:                   MEREN VAN DE 2516, 2716
0530:
0540: 0214 A9 54        PRZEST LDAIM $54      SCHAKEL 5 V IN ZET PRO-
0550: 0216 8D 00 17            STA   DR       GRAMMA MODE 25 V AAN
0560: 0219 A2 00               LDXIM $00      HAAL INHOUD VAN RAM BEGIN
0570: 021B A1 E8               LDAIX TMPRAL   ADRES
0580: 021D A2 08               LDXIM $08      ZET SCHUIFTELLER
0590: 021F A8                  TAY            EN
0600: 0220 20 48 02            JSR   SHFADR   SCHUIF DATABITS D0-D7
0610: 0223 20 89 02            JSR   SHFZET   SCHUIF ADRESBITS A0-A10
0620: 0226 A9 44               LDAIM $44      SCHAKEL PROGRAMMEER-
0630: 0228 8D 00 17            STA   DR       PULS IN EN PROGRAMMEER
0640: 022B 20 7E 02            JSR   WAITV    WACHT 50 MSEC
0650: 022E A9 54               LDAIM $54      SCHAKELPROGRAMMEER-
0660: 0230 8D 00 17            STA   DR       PULS UIT
0670: 0233 60                  RTS            EN KEER TERUG
0680:
0690:                   SUBROUTINE OM ROM EN RAM
0700:                   ADRES OP TE HOGEN
0710:
0720: 0234 E6 E6        INCR    INC   TMPROL  VERHOOG ROM ADRES
0730: 0236 D0 02                BNE   LOOPA   PAGINA VOL?
0740: 0238 E6 E7                INC   TMPROH  JA, VOLGENDE PAGINA
0750: 023A E6 E8        LOOPA   INC   TMPRAL  VERHOOG RAMADRES
0760: 023C D0 02                BNE   RTSA    PAGINA VOL?
0770: 023E E6 E9                INC   TMPRAH  JA, VOLGENDE PAGINA
0780: 0240 60           RTSA    RTS           EN TERUG
0790:
0800:                   SUBROUTINE OM VIER OF VIJF MAAL
0810:                   TE SCHUIVEN
0820:
0830: 0241 0A           SHFV    ASLA          SCHUIF TOTAAL VIJF MAAL
0840: 0242 0A           SHFF    ASLA          SCHUIF TOTAAL VIER MAAL
0850: 0243 0A                   ASLA
0860: 0244 0A                   ASLA
0870: 0245 0A                   ASLA
0880: 0246 A8                   TAY           BEWAAR ACCU IN Y
0890: 0247 60                   RTS           EN KEER TERUG
0900:
0910:                   SUBROUTINE VOOR HET SCHUIVEN
0920:                   VAN HET ADRES NAAR DE UITGANG
0930:
0940: 0248 98           SHFADR  TYA           ADRES NAAR ACCU
0950: 0249 0A                   ASLA          SCHUIF LINKS
0960: 024A A8                   TAY           ACCU NAAR Y
0970: 024B 90 08                BCC   LOOPB   WAS BIT 3 NUL?
0980: 024D AD 00 17             LDA   DR        ZO NEE; ZET DATA OP 1
0990: 0250 09 02                ORAIM $02
1000: 0252 8D 00 17             STA   DR        EN
1010: 0255 20 64 02     LOOPB   JSR   CLOCK   GEEF KLOKPULS
1020: 0258 AD 00 17             LDA   DR      CLEAR DATA
1030: 025B 29 FD                ANDIM $FD
1040: 025D 8D 00 17             STA   DR
1050: 0260 CA                   DEX
1060: 0261 D0 E5                BNE   SHFADR  ALLE BITS GEHAD?
1070: 0263 60                   RTS           DAN TERUG
1080:
1090:                   SUBROUTINE OM KLOKPULS TE MAKEN
1100:                   VOOR HET SCHUIVEN EN PARALLEL LADEN
1110:
1120: 0264 EE 00 17     CLOCK   INC   DR
1130: 0267 CE 00 17             DEC   DR
1140: 026A 60                   RTS
1150:
1160:                   SUBROUTINE OM ACHT DATABITS NAAR
1170:                   BUITEN TE SCHUIVEN
1180:
1190: 026B A2 08        SHFDAT  LDXIM $08     TOTAAL 8 BITS
1200: 026D AC 00 17     LOOPD   LDY   DR
1210: 0270 18                   CLC
1220: 0271 10 01                BPL   LOOPC   IS BIT 7 EEN "0"
1230: 0273 38                   SEC           NEEN
1240: 0274 2A           LOOPC   ROLA          JA; ROTEER ACCU
1250: 0275 20 64 02             JSR   CLOCK   GEEF KLOKPULS
1260: 0278 CA                   DEX           8 MAAL GEWEEST?
1270: 0279 D0 F2                BNE   LOOPD   NEEN; GA TERUG
1280: 027B 85 EA                STA   TEMPA   JA; DATA IN ADRES 01
1290: 027D 60                   RTS
1300:
1310:                   SUBROUTINE OM 50 MS TE WACHTEN
1320:
1330: 027E A0 28        WAITV   LDYIM $28
1340: 0280 A2 FA        LOOPE   LDXIM $FA
1350: 0282 CA           LOOPF   DEX
1360: 0283 D0 FD                BNE   LOOPF   WACHT 1,25 MS
1370: 0285 88                   DEY
1380: 0286 D0 F8                BNE   LOOPE   WACHT 40X1,25=50 MS
1390: 0288 60                   RTS
1400:
1410:                   ROUTINE OM VIJF ADRESBITS
1420:                   TE SCHUIVEN
1430:
1440: 0289 A5 E7        SHFZET  LDA   TMPROH
1450: 028B 20 41 02             JSR   SHFV
1460: 028E 4C AF 02             JMP   ADRES
1470:
1480:                   SUBROUTINE OM DE 2516,2716
1490:                   TE LEZEN
1500:
1510: 0291 A9 04        REZEST  LDAIM $04
1520: 0293 8D 00 17             STA   DR
1530: 0296 20 89 02             JSR   SHFZET  SCHUIF ADRESBITS
1540: 0299 20 A6 02             JSR   EXORIC  ENABLE
1550: 029C 20 64 02             JSR   CLOCK   KLOKPULS
1560: 029F 20 A6 02             JSR   EXORIC  DISABLE
1570: 02A2 20 6B 02             JSR   SHFDAT  SCHUIF DATABITS
1580: 02A5 60                   RTS
1590:
1600:                   SUBROUTINE TER VOORBEREIDING
1610:                   VAN HET LEZEN
1620:
1630: 02A6 AD 00 17     EXORIC  LDA   DR
1640: 02A9 49 1C                EORIM $1C     INVERTEER BIT 2,3 EN 4
1650: 02AB 8D 00 17             STA   DR
1660: 02AE 60                   RTS
1670:
1680:                   SUBROUTINE OM ADRES TE LADEN
1690:
1700: 02AF A2 03        ADRES   LDXIM $03
1710: 02B1 20 48 02             JSR   SHFADR  SCHUIF ADRESBITS 8-10
1720: 02B4 A4 E6                LDY   TMPROL
1730: 02B6 A2 08                LDXIM $08
1740: 02B8 20 48 02             JSR   SHFADR  SCHUIF ADRESBITS 0-7
1750: 02BB 60                   RTS
1760:
1770:                   SUBROUTINE BEPAALT OF EINDADRES
1780:                   IS BEREIKT
1790:
1800: 02BC A5 E4        EADR    LDA   EROMLO  HAAL EINDADRES LAAG
1810: 02BE C5 E6                CMP   TMPROL  VERGELIJK MET ADRES LAAG
1820: 02C0 D0 04                BNE   ARTS    ONGELIJK,TERUG
1830: 02C2 A5 E5                LDA   EROMHI  HAAL EINDADRES HOOG
1840: 02C4 C5 E7                CMP   TMPROH  VERGELIJK MET ADRES HOOG
1850: 02C6 60           ARTS    RTS           Z=0,GELIJK
1860:
1870:
1880:                   PROGRAMMA SCHRIJF 2516 EN 2716
1890:
1900: 02C7 20 00 02     PROG    JSR   INIT    INITIEER PIA EN KOPIEER
1910: 02CA A9 90                LDAIM TABA    HAAL ADRES VAN POINTER
1920: 02CC 85 EB                STA   TEMPB   BEWAAR
1930: 02CE A9 03                LDAIM TABA    / HOGE BYTE
1940: 02D0 85 EC                STA   TEMPB   +01
1950: 02D2 A9 04                LDAIM $04     SCHAKEL IN
1960: 02D4 8D 00 17             STA   DR
1970: 02D7 20 7E 02             JSR   WAITV   WACHT OP RELAIS
1980: 02DA 20 91 02     CHFF    JSR   REZEST  HAAL BYTE
1990: 02DD A5 EA                LDA   TEMPA
2000: 02DF C9 FF                CMPIM $FF     GELIJK $FF?
2010: 02E1 D0 3E                BNE   TERR    NIET GELIJK, FOUT
2020: 02E3 20 BC 02             JSR   EADR    KLAAR?
2030: 02E6 F0 05                BEQ   FUR
2040: 02E8 20 34 02             JSR   INCR    NEEN, HOOG OP
2050: 02EB D0 ED                BNE   CHFF    VOLGENDE BYTE
2060: 02ED 20 0A 02     FUR     JSR   SAVTMP  JA, KOPIEER OPNIEUW
2070: 02F0 A9 54                LDAIM $54     SCHAKEL IN
2080: 02F2 8D 00 17             STA   DR
2090: 02F5 20 7E 02             JSR   WAITV   WACHT OP RELAIS
2100: 02F8 20 14 02     BYT     JSR   PRZEST  PROGRAMMEER EEN BYTE
2110: 02FB 20 BC 02             JSR   EADR    ALLES GEHAD?
2120: 02FE F0 05                BEQ   VERF
2130: 0300 20 34 02             JSR   INCR    NEEN, HOOG OP
2140: 0303 D0 F3                BNE   BYT     VOLGENDE BYTE
2150:
2160:                   PROGRAMMA VERIFIEER 2516 EN 2716
2170:
```

Lijst *Vervolg*

```
0320: 0305 20 00 02  VERF   JSR   INIT   JA, KOPIEER
0330: 0308 A9 96            LDAIM TABB   HAAL ADRES VAN POINTER
0340: 030A 85 EB            STA   TEMPB  BEWAAR
0350: 030C A9 03            LDAIM TABB   / HOGE BYTE
0360: 030E 85 EC            STA   TEMPB  +01
0370: 0310 A9 04            LDAIM $04
0380: 0312 8D 00 17         STA   DR     SCHAKEL IN
0390: 0315 20 7E 02         JSR   WAITU  WACHT OP RELAIS
0400: 0318 20 91 02  CHK    JSR   REZEST HAAL BYTE
0410: 031B A2 00            LDXIM $00
0420: 031D A1 E8            LDAIX TMPRAL HAAL RAM BYTE
0430: 031F C5 EA            CMP   TEMPA  VERGELIJK MET EPROM-BYTE
0440: 0321 D0 12     TERR   BNE   ERR    NIET GELIJK, FOUT
0450: 0323 20 BC 02         JSR   EADR   ALLES GEHAD?
0460: 0326 F0 05            BEQ   NEX
0470: 0328 20 34 02         JSR   INCR   NEEN, HOOG OP
0480: 032B D0 EB            BNE   CHK    VOLGENDE BYTE
0490: 032D A9 9C     NEX    LDAIM TABC   HAAL ADRES VAN POINTER
0500: 032F 85 EB            STA   TEMPB  BEWAAR
0510: 0331 A9 03            LDAIM TABC   / HOGE BYTE
0520: 0333 85 EC            STA   TEMPB  +01
0530: 0335 A9 24     ERR    LDAIM $24    RESET
0540: 0337 8D 00 17         STA   DR
0550: 033A 20 5E 03         JSR   DISP   LEES UIT
0560: 033D 4C 4F 1C         JMP   MON    TERUG NAAR MONITOR
0570:
0580:                PROGRAMMA LEES 2516 EN 2716
0590:
0600: 0340 20 00 02  LEES   JSR   INIT
0610: 0343 A9 04            LDAIM $04
0620: 0345 8D 00 17         STA   DR     SCHAKEL IN
0630: 0348 20 7E 02         JSR   WAITU  WACHT OP RELAIS
0640: 034B 20 91 02  RED    JSR   REZEST HAAL BYTE
0650: 034E A2 00            LDXIM $00
0660: 0350 A5 EA            LDA   TEMPA
0670: 0352 81 E8            STAIX TMPRAL ZET BYTE IN GEHEUGEN
0680: 0354 20 BC 02         JSR   EADR   ALLES GEHAD?
0690: 0357 F0 D4            BEQ   NEX
0700: 0359 20 34 02         JSR   INCR   NEEN, HOOG OP
0710: 035C D0 ED            BNE   RED
0720: 035E A2 00     DISP   LDXIM $00    TELLER
0730: 0360 86 FD     START  STX   TMPX   IN TMPX
0740: 0362 A9 7F            LDAIM $7F    SET DDR
0750: 0364 8D 41 17         STA   $1741
0760: 0367 A2 09            LDXIM $09    KIES DISPLAY
0770: 0369 A0 06            LDYIM $06    ZES DISPLAYS
0780: 036B 8E 42 17  LOOP   STX   $1742  LICHT DISPLAY OP
0790: 036E B1 EB            LDAIY TEMPB  HAAL KARAKTER
0800: 0370 8D 40 17         STA   $1740  DISPLAY
0810: 0373 E8               INX
0820: 0374 E8               INX          VOLGENDE DISPLAY
0830: 0375 88               DEY          VOLGENDE KARAKTER
0840: 0376 20 87 03         JSR   DELAY  WACHT
0850: 0379 D0 F0            BNE   LOOP   ALLES GEHAD?
0860: 037B 8C 40 17         STY   $1740  DISPLAY UIT
0870: 037E 20 87 03         JSR   DELAY  WACHT
0880: 0381 A6 FD            LDX   TMPX
0890: 0383 CA               DEX          WACHT EXTRA
0900: 0384 D0 DA            BNE   START
0910: 0386 60               RTS
0920: 0387 84 FC     DELAY  STY   TEMP   BEWAAR Y
0930: 0389 A0 00            LDYIM $00
0940: 038B 88        ALOOP  DEY
0950: 038C D0 FD            BNE   ALOOP
0960: 038E A4 FC            LDY   TEMP   HAAL Y
0970: 0390 60        TABA   RTS          DUMMY
0980: 0391 F1               =     $F1
0990: 0392 F1               =     $F1
1000: 0393 D4               =     $D4
1010: 0394 F9               =     $F9
1020: 0395 F9               =     $F9
1030: 0396 BD        TABB   =     $BD
1040: 0397 F9               =     $F9
1050: 0398 9E               =     $9E
1060: 0399 F8               =     $F8
1070: 039A BE               =     $BE
1080: 039B BF               =     $BF
1090: 039C F1        TABC   =     $F1
1100: 039D F9               =     $F9
1110: 039E BF               =     $BF
1120: 039F D4               =     $D4
1130: 03A0 86               =     $86
1140: 03A1 F9               =     $F9
1150: 03A2 00               =     $00
```