D. M. DE BOER

Grafisch TV-display

Stelt u zich eens voor, een gewone TV waarop uzelf tekeningen kunt maken. Dit kan variëren van gewone tekst, grafische voorstellingen van wiskundige functies tot een tekening gemaakt door een kleuter. Voor dit doel werd het TV-scherm verdeeld in een puntenraster van 256 × 256 punten!! Een dergelijk raster geeft ongeveer de maximale resolutie die op een gewone TV (zonder interliniëring) kan worden weergegeven. De toepassingsmogelijkheden zijn legio. De computer-hobbyist heeft er een bijzonder mooi uitvoerorgaan mee, een soort combinatie van plotter en teletype. Voor de niet-computerbezitter is het mogelijk dit toestel uit te bouwen tot een hobby-computer.
Dit TV-display is de eerste schakel. Aan de ingang treft u 8 adreslijnen en wat stuursignalen aan. Met deze aansluitingen is het mogelijk *elke* willekeurige stip (van de totaal 65536) op het scherm aan of uit te zetten. Natuurlijk ligt het sturen met een microcomputer het meest voor de hand. In een later stadium zal dit display een eigen microprocessor krijgen, waardoor een grafische/alphanumerieke terminal ontstaat.

Een ontwerp naar maat

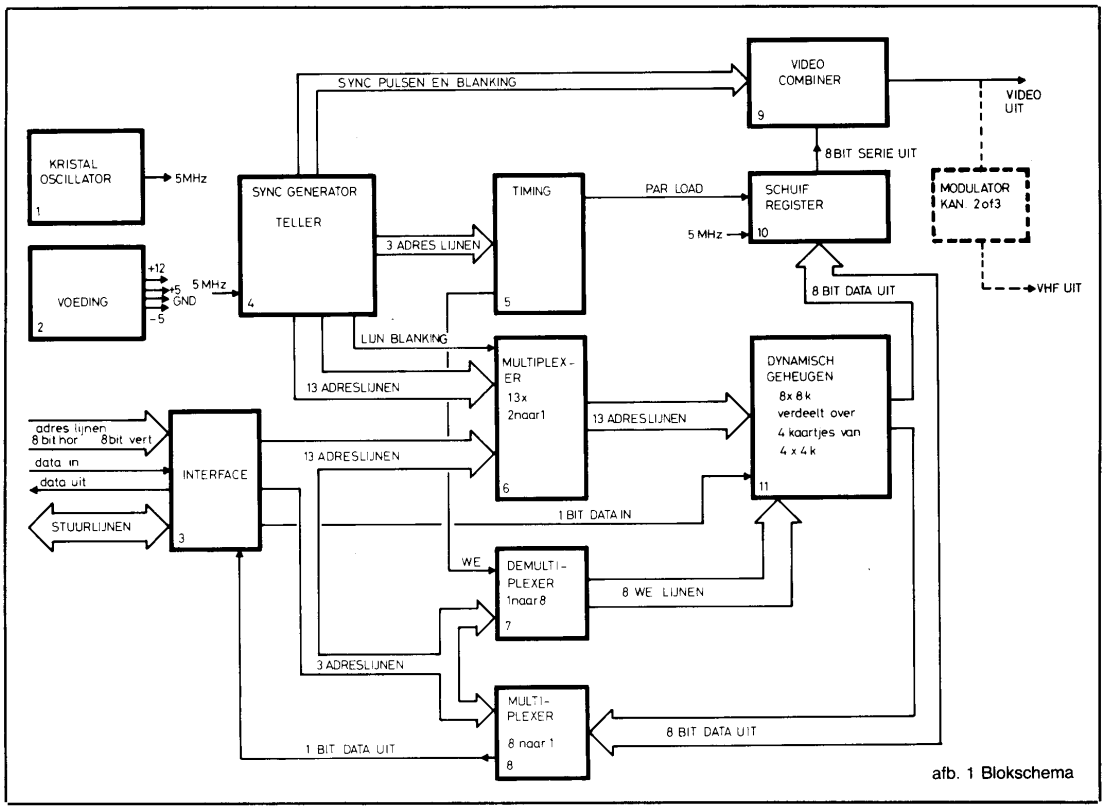
Dit ontwerp is zo gemaakt dat u zowel een goedkopere, als een duurdere versie kunt maken. De goedkopere versie werkt met een kwart van de normale hoeveelheid geheugen. De consequentie is een grover raster (128 × 128 puntjes). Overigens is het **resultaat niet slecht. De duurdere versie heeft naast zwart en wit ook nog de mogelijkheid om twee grijstinten weer te geven. Om dit te bereiken moet de geheugencapaciteit worden verdubbeld. Bij dit ontwerp is het steeds mogelijk om na verloop van tijd op een duurdere versie over te stappen. Uitbreidingen zoals een kleurenversie, toevoeging van een lichtpen (echt tekenen op het scherm!) behoren tot de mogelijkheden, en worden op dit moment onderzocht.**

Ontwerpfilosofie

Bij het ontwerp zijn we ervan uitgegaan dat het geheel zo flexibel mogelijk moet worden. De eerste keus die gemaakt moet worden is: gaan we een grafisch display maken of een alphanumeriek display? Een alphanumeriek display kan uitsluitend letters en cijfers weergeven. Een grafisch display daarentegen kan ook lijnen van elke lengte zichtbaar maken. Door lijnen te combineren kunnen alle mogelijke figuren weergegeven worden, dus ook alle letters! Het zal duidelijk zijn dat het grafisch display verreweg de voorkeur verdient. Buiten de mogelijkheid om grafieken, figuren enz. weer te geven bestaat met het grafische display ook de mogelijkheid om bijv. de letter I wat minder en de letter M wat meer ruimte te geven. Bij het alphanumeriek display zijn alle letterbreedten aangepast aan de breedste letter. Verder is het grafische display in staat om ook grotere letters en bewegende figuren te laten zien. Het grafische display heeft één nadeel; het heeft veel meer geheugen nodig. De laatste tijd zijn de geheugenchips echter sterk in prijs gedaald (4K **dynamisch kost ca. *f* 20.-) zodat de extra kosten opwegen tegen de vele extra mogelijkheden. De keus is dus** gevallen op het grafische display. Bij de keus van de gebruikte onderdelen gold als criterium: Welke mogelijkheid is het goedkoopst? Uiteraard werd erop gelet dat de goedkoopste oplossing wél een goede oplossing was.

Het blokschema (afb. 1)

In afb. 1 vindt u het blokschema van de totale schakeling. Het TV-display werkt met een puntenraster van 256 punten horizontaal en 256 punten verticaal. Totaal levert dit 65536 punten op, zodat we in principe een geheugen nodig hebben van 64 K × 1 bit (1 K = 1024). Op één lijn kunnen maximaal 256 punten worden geschreven. Dit moet in 51,2 µs gebeuren (de sync duurt 12,8 µs), zodat elke punt in 200 ns wordt geschreven. Het 64 K éénbits geheugen zou dus elke 200 ns nieuwe data moeten leveren. Dit is voor de meeste geheugenchips (nog?) te snel, zodat we gebruik moeten maken van een schuifregister. In dit geval een 8-bits schuifregister (74166). Het 64 K geheugen wordt daarom uitgevoerd als 8 × 8 K. Eéns in de 1,6 µs (8 × 0,2) moet het geheugen nu 8 bits tegelijk uitlezen. Het schuifregister zorgt ervoor dat deze 8 bits op het scherm terecht komen, terwijl het geheugen ruimschoots de tijd heeft om de nieuwe 8-bits-data in orde te maken. De syncgenerator (blok 4 afb. 1) zorgt niet alleen voor de benodigde syncpulsen, maar ook voor het genereren van de adressen. De teller begint links boven op het TV-scherm met 0000 (HEX) Aan het eind van de eerste lijn staat de teller op 00FF. Tijdens de syncpuls wordt de teller gebruikt voor de timing van interne signalen. Gedurende deze tijd schakelt de multiplexer (blok 6) de adreslijnen om, zodat de computer met het geheugen is verbonden en de data geschreven of gelezen kan worden. Aan het begin van de volgende lijn staat de teller keurig op 0100 (HEX). Het eerste byte stelt dus het verticale adres voor. De lijnen zijn genummerd van 00 (boven) tot FF (beneden). Het tweede byte stelt het horizontale adres voor. De 256 punten op één lijn zijn genummerd van 00 (links) tot FF (rechts).
Op deze manier heeft dus elk punt op het beeld een eigen adres. Omdat het geheugen 8 naast elkaar gelegen bits tegelijk uitleest hebben we horizontaal nog maar 5 adreslijnen over. (5 adreslijnen geven $2^5 = 32$ verschillende adressen. Elk adres heeft 8 bits, zodat er totaal weer 8 × 32 = 256 verschillende punten zijn). De hele schakeling werkt met 8 verticale adreslijnen, en 5 horizontale adreslijnen. Totaal zijn er dus 13 adreslijnen. De 3 niet gebruikte adreslijnen, welke in de syncgenerator wél worden opgewekt, zorgen voor de timing in de rest van de schakeling. Deze drie adreslijnen doorlopen immers alle 8 mogelijke adressen binnen één cyclus van 1,6 µs. Door deze adressen aan een binair/BCD omzetter toe te voeren (timing, blok 5) hebben we dit tijdvak van 1,6 µs onderverdeeld in 8 tijden van 200 ns. Uit deze BCD omzetter betrekken we de timing voor signalen als $\overline{WE}$, $\overline{CE}$, Parallel load (schuifregister) enz.
We hebben nu bijna alle blokken die



afb. 1 Blokschema

strikt noodzakelijk zijn kort besproken (4, 5, 6, 10 en 11, afb. 1). De voeding (2) en de kristaloscillator (1) behoeven hier nog geen toelichting. De videocombiner (9) zorgt dat de syncpulsen, de blankingpulsen en de video-informatie gecombineerd worden tot het complete videosignaal. De mooiste oplossing is om dit videosignaal rechtstreeks aan de TV toe te voeren. Eventueel kan een VHF modulator toegevoegd worden, maar dit komt de kwaliteit van het beeld niet ten goede.

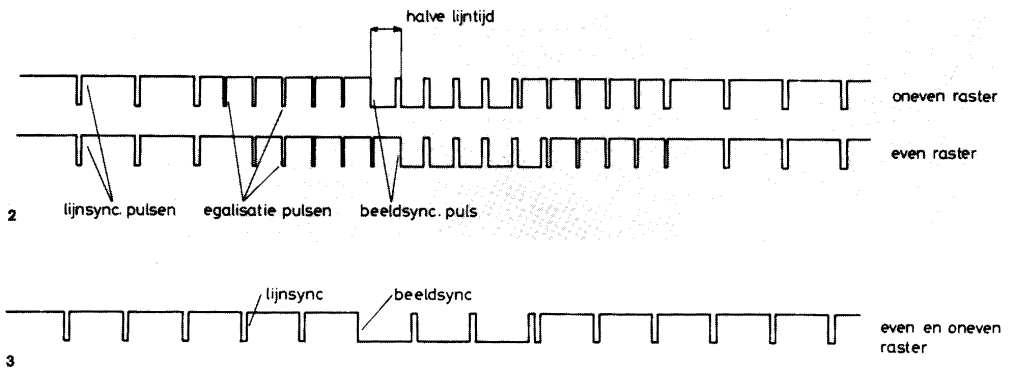
De blokken 7 en 8 (multiplexer en demultiplexer) zorgen ervoor dat het 8 × 8 k geheugen er voor de computer uitziet als 1 × 64 k. Dit is niet strikt noodzakelijk, maar het vereist slechts 2 extra IC's en biedt veel voordelen. Allereerst wordt de software voor de computer eenvoudiger, doordat we maar 1 data-bit hebben. Ten tweede wordt het aantal verbindingen tussen de computer en het display beperkt. We hebben nu immers maar 1 data lijn in, en 1 data lijn uit. Wel zijn er 3 adreslijnen extra, maar we hebben toch maar liefst 11 lijnen uitgespaard. Dit aantal wordt nog groter als het geheugen dubbel wordt uitgevoerd, zodat ook grijstinten kunnen worden weergegeven.

De computer heeft dus gewoon 8 horizontale en 8 verticale adreslijnen. Van de horizontale lijnen gaan er zoals gezegd 5 naar het geheugen. De overige 3 worden gebruikt om op slechts 1 van de 8 plaatsen te schrijven, of om slechts 1 van de 8 bits te lezen.

Dynamisch geheugen

Omdat dynamische RAM's nu eenmaal veel goedkoper zijn dan de statische, zijn hier dynamische RAM's toegepast. Het nadeel dat deze cellen om de 2 ms gerefreshed moeten worden, valt hier weg. Het geheugen wordt immers continu uitgelezen. Voor het refreshen is het voldoende om adreslijnen A0 ... A5 uit te lezen. De toestand van $\overline{CS}$ doet hierbij niet terzake. De adressen A0 ... A4 worden tijdens elke lijn doorlopen. Adres A5 behoort tot de lijnadressen. Dit adres is zo geschakeld dat in 4 lijnen van het TV-scherm het totale geheugen gerefreshed is. Deze 4 lijnen worden in 4 × 64 µs = 256 µs geschreven, zodat we ons over het refreshen niet druk hoeven te maken. Het schrijven van de data in het geheugen kan tijdens de lijn en/of beeldterugslag gebeuren. In deze schakeling is gekozen voor schrijven alléén tijdens de lijnterugslag. Tijdens deze terugslag hebben we precies tijd genoeg om 8 punten te schrijven. Schrijven tijdens de beeldsync heeft weinig zin, omdat de microprocessor toch niet snel genoeg kan werken om meer dan 8 punten per 64 µs te geven. Daarbij geeft het schrijven in het geheugen tijdens de beeldsync problemen met het refreshen. De totale onderdrukkingspuls bij beeldterugslag duurt immers 56 × 64 µs = 3584 µs. Wanneer we deze tijd gebruiken om in het geheugen te schrijven, hebben we geen enkele garantie dat na 2 ms alle geheugenplaatsen gerefreshed zijn. Al met al is het dus veel praktischer om alléén tijdens lijnterugslag te schrijven.

Om prijstechnische redenen hebben we een beperking ingevoerd. Het TV-display is snel genoeg om elke 64 µs 8 punten te schrijven. De computer moet steeds de adressen voor deze punten leveren. Dit houdt in, dat we een buffergeheugen nodig hebben om de 8 verticale adressen, en 8 horizontale adressen te onthouden. Door in de



afb. 2. *De syncpulsen volgens de CCIR-norm, met als resultaat een geinterlinieerd beeld.*

afb. 3. *De syncpulsen zoals ze in ons ontwerp gemaakt moeten worden. Met deze pulsen worden het even en het oneven raster precies over elkaar heen geschreven.*

schakeling maar één buffergeheugen op te nemen voor de 16 adresbits, is de schrijfsnelheid beperkt tot 1 punt per 64 µs. Aan de ene kant levert dit een belangrijke besparing van de hardware, aan de andere kant is de combinatie computer/TV-display toch niet erg veel langzamer geworden, omdat 64 µs in de praktijk erg weinig is om een nieuw punt te bepalen en aan het scherm door te geven. Wanneer er ooit gewerkt zou worden met een microcomputer die veel sneller is, kan de schakeling altijd worden aangepast.

Opbouw van het TV-beeld

Het TV-beeld is opgebouwd uit 625 lijnen, verdeeld over 2 in elkaar grijpende rasters van elk 312½ lijn. Door dit in elkaar grijpen van de rasters (interliniëren) krijgen we de indruk dat de beeldfrequentie 50 Hz is i.p.v. 25 Hz. Voor bewegende beelden is dit een mooie oplossing. Wanneer we echter maar één oplichtende lijn hebben, krijgen we de indruk dat de lijn heen en weer springt. Daarom is bij dit ontwerp gezorgd dat beide rasters precies over elkaar heen vallen, zodat een rustig beeld wordt verkregen. Hierdoor is het aantal lijnen op het scherm teruggebracht tot 312. Hiervan gaan er nog eens ca. 25 verloren tijdens de beeldterugslag. In principe kunnen we dus 287 lijnen gebruiken. Om praktische redenen is dit aantal terug gebracht tot de eerder genoemde 256, zodat het verticale adres precies 1 byte in beslag neemt.

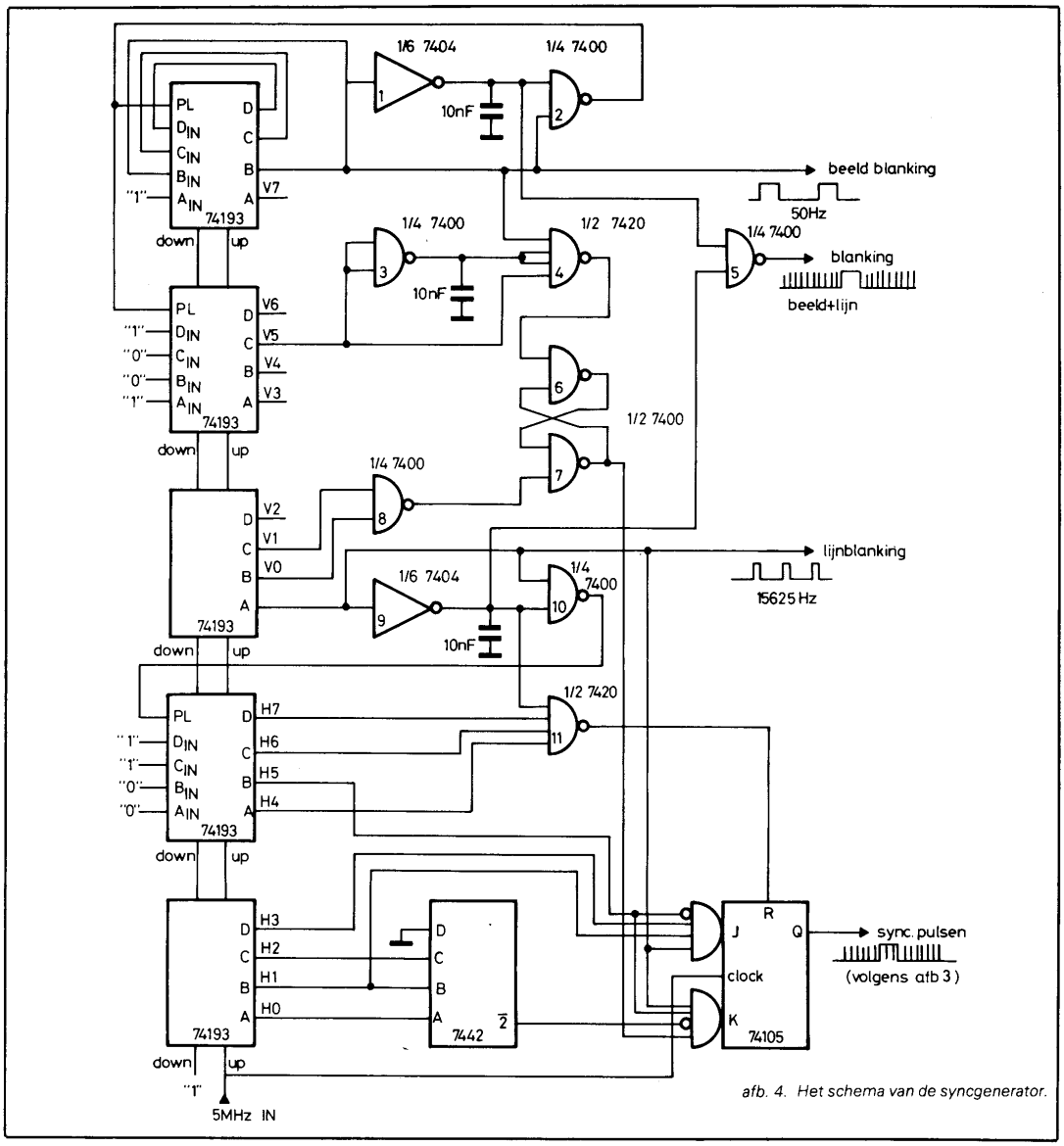
Om te zorgen dat de 2 rasters precies over elkaar heen vallen moeten we bewust afwijken van de CCIR-norm voor wat betreft de sync-pulsen. In afb. 2 ziet u de sync-pulsen overeenkomstig deze norm. Doordat de beeldsync bij het even raster een halve lijn tijd is verschoven t.o.v. de beeldsync bij het oneven raster, wordt de interliniëring (het in elkaar grijpen van de rasters) verkregen. De beeldsync wordt in de TV weer gescheiden van de lijnsyncpulsen d.m.v. integratie. Om te zorgen dat de integratiecondensator in de ontvanger bij aanvang van de beeldsyncpuls zowel bij het even als bij het oneven raster dezelfde beginlading heeft, worden op de helft van de lijn zogenaamde egalisatiepulsen toegevoegd. De TV heeft nu, populair gezegd, niet in de gaten dat de beeldsyncpuls soms op de helft van de lijn begint. De lijnsyncpulsen moeten gedurende de beeldsync doorlopen om de lijnoscillator in de pas te houden. De egalisatiepulsen hebben geen effect op de lijnoscillator in de TV, omdat deze oscillator de 2× zo hoge frequentie van de egalisatiepulsen toch niet kan bijhouden.

Bij ons ontwerp willen we de beide rasters juist precies op elkaar laten vallen. Om dit te bereiken moeten we de beeldsync bij het even en het oneven raster relatief precies op hetzelfde moment laten beginnen. Hierdoor vervalt ook de noodzaak van de egalisatieimpulsen. De syncpulsen zoals wij ze dus nodig hebben staan getekend in afb. 3. Het leek ons weinig zinvol om een schakelaar 'wel/niet interliniëren' toe te voegen, omdat goede interliniëring alleen ontstaat als ook de egalisatiepulsen worden opgewekt. Dit vereist weer een groter aantal componenten in de syncgenerator, zodat de totale schakeling duurder wordt. Gezien het feit dat interliniëring toch niet wenselijk is (alle letters lijken te 'dansen') hebben we er in dit ontwerp maar helemaal van af gezien.

De syncgenerator

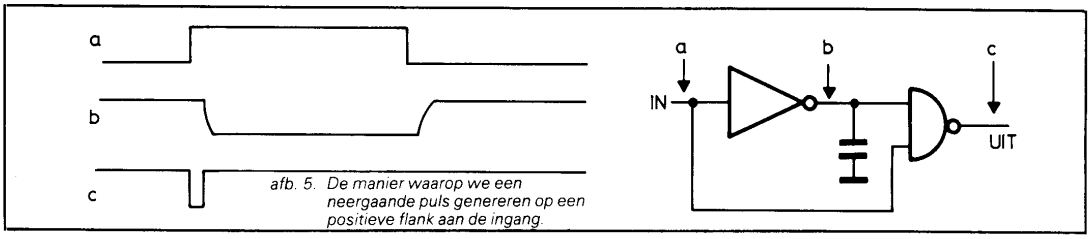
Voor de syncgenerator hebben we bewust geen compleet sync-IC genomen. Allereerst zijn deze IC's vrij duur, en ten tweede zijn ze niet altijd even gemakkelijk leverbaar. Daarbij komt dat we de 5 delen IC's (5 × 74193, afb. 4) toch nodig hadden voor het genereren van de adressen. Door toevoeging van een paar poorten is nu vrij eenvoudig een syncgenerator te maken.

Bij de uitleg van de werking beginnen we links onderaan in het schema uit afb. 4. Hier zorgen 2 tellers voor het opwekken van de horizontale adressen (H0 ... H7). De 'DOWN' ingang leggen we aan een logische '1'. Hierdoor zal in de hele delerketen de 'DOWN'-ingang op een logisch '1' niveau blijven. In de schakeling kunnen we nu eenvoudig alle ongebruikte ingangen aan dit log. '1' niveau leggen. De 'UP'-ingang voeden we met een 5 MHz blokgolf. Wanneer de eerste 2 tellers een volledige cyclus hebben doorlopen (na 256 clockpulsen) zal er een tijd verstreken zijn van 256 × 200 ns = 51,2 µs. Dit is precies de 'zichtbare' tijd van 1 lijn op het TV-scherm. Hierna moet de schakeling een lijnsyncpuls en een lijn blankingpuls genereren. De syncpuls zorgt dat de lijnoscillator in de TV met een



afb. 4. Het schema van de syncgenerator.

nieuwe zaagtand begint. De blankingpuls zorgt dat de lijn tijdens de terugslag niet zichtbaar is. De blankingpuls moet een lengte hebben van 12,8 µs, zodat de totale lijntijd uitkomt op 64 µs. Wanneer de eerste 2 tellers een volledige cyclus hebben gemaakt (na 51,2 µs) zal uitgang A van de 3e teller van '0' naar '1' gaan. De schakeling rond de poorten 9 en 10 genereert op deze flank een neergaande puls. In afb. 5 is dit stukje ter verduidelijking uit de schakeling gelicht. Aan de NAND poort voeren we signalen a en b toe. Doordat signaal b vertraagd is t.o.v. signaal a, zullen gedurende een korte tijd de signalen a én b '1' zijn. Dit zal een korte neergaande puls op de uitgang tot gevolg hebben. Bij een neergaande flank op de ingang zullen de signalen gedurende korte tijd beide '0' zijn. De uitgang van de NAND zal nu gewoon '1' blijven. Terug naar het schema uit afb. 4. We zien dat de opgaande flank van uitgang A van de derde teller via poorten 9 en 10 een korte neergaande puls levert aan de 'parallel load' ingang van de tweede teller. Hierdoor zal deze teller geladen worden met de vast ingestelde waarde van '1100'. Het zal nu 64 klokpulsen duren voordat de onderste 2 tellers weer opnieuw op 0 staan. Deze klokpulsen duren precies 64 × 200 ns = 12,8 µs. Uitgang A van de derde teller zal nu weer van '1' naar '0' zijn gegaan. Tevens is het lijnadres (V0 ... V7) opgehoogd. Uitgang A (3e teller) levert dus de blankingpuls voor de lijnterugslag. De syncpuls wordt gemaakt door de JK flip flop 74105 (rechts onder). De clockingang wordt continu getriggerd, zodat de toestand



afb. 5. De manier waarop we een neergaande puls genereren op een positieve flank aan de ingang.

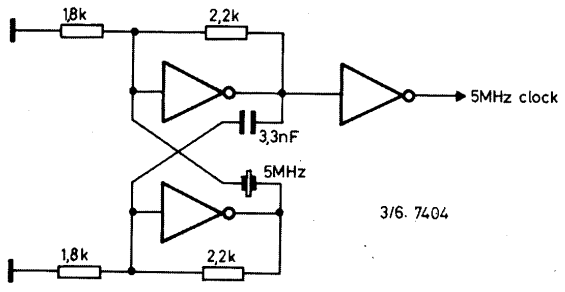
van de Q uitgang bepaald wordt door de signalen op de J ingangen, de K ingangen en de reset. De gecombineerde JK ingang van de 74105 wordt met 'lijnblanking' verbonden. Hierdoor kan tijdens het zichtbare deel van de beeldlijn de toestand van de JK flip flop alleen veranderen door 'reset'. Tijdens de beeldlijn kan dus **nooit** een syncpuls ontstaan. Tijdens de lijnblanking zijn de J- en de K-ingangen vrij gegeven. Op de J-ingangen komt de voorwaarde voor 'sync aan', op de K-ingang komt de voorwaarde voor 'sync uit'. De J- en K-ingangen zijn precies zo met de tellers verbonden, dat een goede syncpuls ontstaat. Doordat een K-ingang verbonden is met de $\overline{J}$-ingang kunnen de voorwaarden voor sync aan of uit nooit gelijktijdig optreden. Eén van de K-ingangen is verbonden met de uitgang van poort 7. Normaal is deze uitgang '1', en heeft op de voorwaarde voor 'sync uit' geen invloed. Op het doel van deze verbinding komen we nog terug.

We weten nu dus hoe de lijnsyncpuls en de lijnblankingpuls ontstaan. Na elke lijn (mét blanking) wordt het verticale adres opgehoogd (V0 ... V7). Wanneer we een volledig beeld hebben doorlopen (256 lijnen) wordt uitgang B van de bovenste teller '1'. De overgang van '0' naar '1' van deze uitgang zal weer een neergaande puls aan de uitgang van poort 2 opleveren. Hierdoor zullen de bovenste 2 tellers weer geladen worden met een vast ingestelde waarde. Pas na 56 lijnen zal de verticale adresteller weer op 0 staan. We hebben dan precies de 312 lijnen van één beeld doorlopen, en uitgang B van de bovenste teller is weer '0' geworden. Deze uitgang is dus '1' tijdens de 56 lijnen bij de beeldterugslag. Aldus wordt de beeldblanking gevormd. De lijn- en beeldblanking komen samen geïnverteerd op poort 5, waar een totaal blankingsignaal wordt gemaakt. We hebben nu alle benodigde signalen, op de beeldsync na. Om het beeld op de TV ongeveer in het midden te laten uitkomen moet de beeldsync ongeveer op de helft van de beeldblanking komen. De beeldsync ontstaat als V5 van 0 naar 1 gaat. De bekende schakeling rond poort 3 en 4 vormt op deze overgang een korte neergaande puls. Doordat poort 4 als extra voorwaarde is verbonden met de beeldblanking, kan deze puls alleen ontstaan tijdens de beeldterugslag. Wanneer dus V5 tijdens de beeldterugslag van 0 naar 1 gaat, zal de korte puls op de uitgang van poort 4 de flip flop (poorten 6 en 7) omzetten. De uitgang van poort 7 zal hierdoor gedurende 3 lijnen '0' worden. Poort 8 zal immers zorgen dat de flip flop (poorten 6 en 7) op het juiste moment terug gezet wordt. Doordat de uitgang van poort 7 gedurende 3 lijnen '0' wordt, zal ook de met deze uitgang verbonden K-ingang van de 74105 '0' blijven. De voorwaarde voor 'sync uit' kan dan ook niet ontstaan, en de syncpuls zal hoog blijven. (Beeldsync volgens afb. 3) blijven. Juist voordat de nieuwe lijnsync begint wordt de 74105 door poort 11 gereset. De neergaande flanken van de lijnsync én de beeldsync liggen dus altijd op hetzelfde moment (relatief), omdat de voorwaarde voor 'sync aan' **nooit** veranderd. Alleen de voorwaarde 'sync uit' wordt elk raster gedurende 3 lijnen gewijzigd. Op deze wijze hebben we meteen de gecombineerde syncpulsen te pakken.

De drie condensatoren in deze syncgenerator zorgen voor de juiste pulsbreedte aan de uitgangen van poorten 2, 4 en 10. Deze pulsbreedten zijn *niet* critisch, ze mogen echter niet breder zijn dan ca. 3 µs omdat anders de PL-puls van de 2e teller de puls op de 'UP'-ingang blokkeert. De minimum breedte is ongeveer 20 ns. Door deze ruime grenzen kunnen de drie condensatoren in deze schakeling nooit een instabiliteit veroorzaken. De overige pulsen worden opgewekt aan de hand van de tellerstanden, en liggen hierdoor 'keihard' vast. Voor de tellers is bewust een synchroon type gekozen, omdat de grote delertrein anders nooit op tijd 'omgeklapt' is. De uitgangen C en D van de bovenste teller kunnen eventueel als rasterteller worden gebruikt. De 7442 midden onderaan (afb. 4) is voor de syncgenerator eigenlijk overbodig, maar zoals uit het blokschema blijkt gebruiken we de uitgangen van deze binair/BCD omzetter voor de timing van signalen als $\overline{CE}$, $\overline{WE}$, par. load (schuifregister) enz.

De 5 MHz oscillator

In afb. 6 vindt u het schema van de kristaloscillator. De oscillator wordt gevormd door 2 inverters. De derde inverter dient als buffer en zorgt er tegelijkertijd voor dat het uitgangssignaal een mooie blokgolf is. Het kristal zorgt vooral voor de stabiliteit van de opgewekte frequentie. De hoogte van de frequentie is minder belangrijk, en moet liggen tussen de 5 MHz en 5,05 MHz.



afb. 6. De kristaloscillator.

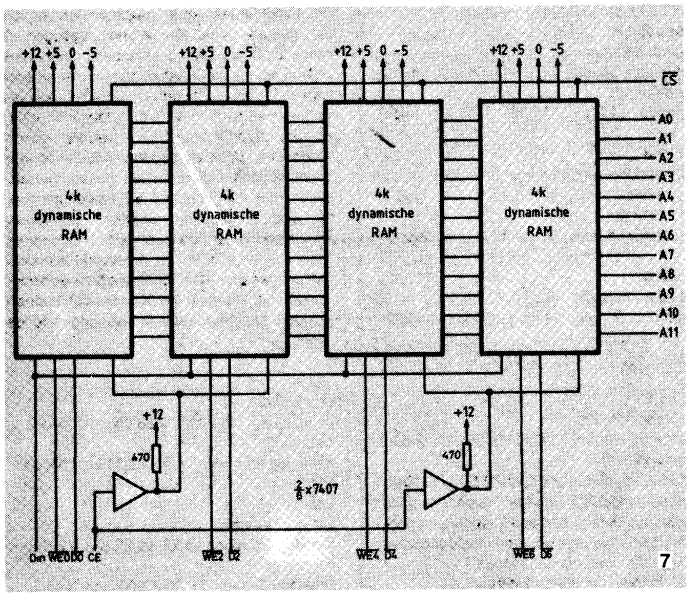
Het geheugen

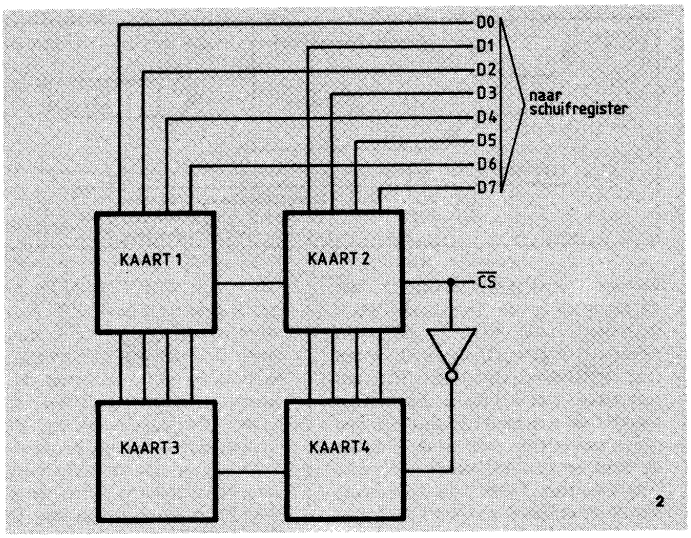
Zoals reeds aangekondigd, maken we in dit ontwerp gebruik van dynamische 4 k RAM's. Dynamisch omdat de dynamische RAM's veel goedkoper zijn dan hun statische broers. Het nadeel, dat dynamische RAM's om de 2 ms gerefreshed moeten worden speelt hier geen rol, omdat het geheugen cyclisch wordt uitgelezen. Hierdoor wordt automatisch elke gerefreshed.

Verder is in dit ontwerp gekozen voor de 22 pins 4 k RAM (industrie standaard). Ook dit is bewust gedaan, omdat de 16- en 18 pins uitvoeringen gebruik maken van resp. gemultiplexte adressen of gecombineerde in- en uitgangen. De 22-pins uitvoering heeft zowel alle adressen als de afzonderlijke in- en uitgang beschikbaar, zodat hier de aansturing het eenvoudigst kan geschieden.

Het geheugen is verdeeld over 4 kaartjes van elk 4 x 4 k. Op elke geheugenkaart bevinden zich vier geheugenchips. Het schema van één geheugenkaartje vindt u in afb 7. Zoals u ziet zijn alle adressen met elkaar doorverbonden. Dit is ook het geval met de $\overline{CS}$ aansluitingen, zodat we nu de mogelijkheid hebben het totale kaartje wel of niet te selecteren. Onderaan in het schema vindt u de aansluitingen D0, D2, D4 en D6. Dit zijn de 4 data uitgangen, die samen met vier uitgangen van een ander geheugenkaartje op het 8-bits schuifregister worden aangesloten. Bij dit tweede kaartje spreken we niet van D0, D2, D4 en D6, maar van D1, D3, D5 en D7. Waarom we deze nummering zo hebben gekozen, zal nog blijken. De data in lijnen zijn ook allemaal aan elkaar gekoppeld. Dit kan, omdat we toch maar in één bitje tegelijk willen schrijven. Door de $\overline{WE}$-puls op het juiste IC te zetten, wordt slechts in één van de vier bitjes geschreven. Dit verklaart tevens waarom de $\overline{WE}$-lijnen los naar buiten worden gevoerd. De klokpuls voor de dynamische chips (CE) moet op MOS niveau liggen, dus op 12 V. Het is niet mogelijk om de TTL klok één keer om te zetten naar MOS niveau en dan naar de vier geheugenkaartjes te gaan. Deze methode zou veel te veel overspraak geven op de zeer hoogohmige klokleiding. Daarom komt er op elk kaartje een eigen levelshifter (7407). Jammer genoeg blijven nu 4 van de 6 buffers ongebruikt. Het gebruikte IC is echter niet duur, zodat dit geen groot bezwaar is.

afb. 7 Het schema van de 4 x 4 k RAM kaart.





afb. 8 Zo zijn de kaartjes onderling gekoppeld.

De goedkope versie, met 1 geheugenkaartje

Zoals dus gezegd, is het geheugen verdeeld over 4 kaartjes van 4 x 4 k. Hierdoor kunnen we op eenvoudige wijze het puntenraster van 256 x 256 (4 geheugenkaartjes) terugbrengen tot 128 x 128 (1 geheugenkaartje). In afb. 8 ziet u hoe de 4 geheugenkaartjes onderling zijn doorverbonden.

Voor de duidelijkheid zijn alleen de datalijnen en de $\overline{CS}$-lijnen getekend. De adreslijnen, de Dinlijnen en de $\overline{CE}$-lijn zijn gewoon doorgelust. De $\overline{WE}$-lijnen worden op dezelfde manier behandeld als de data-uitgangen. (De $\overline{WE}$-lijnen gaan natuurlijk niet naar het schuifregister, maar naar een demultiplexer, afb. 16). Terug naar afb. 8. We zien dat de data-uitgangen van kaartje 1 en 3 (2 en 4) met elkaar zijn doorverbonden. De $\overline{CS}$-lijn zorgt ervoor dat op de even lijnen van het TV-scherm kaartje 1 (2) geselecteerd is, en op de oneven lijnen kaartje 3 (4).

Wanneer we nu uit kostenoverwegingen maar 1 geheugenkaartje gebruiken, laten we zeggen alléén kaartje 2, dan zal het schuifre-

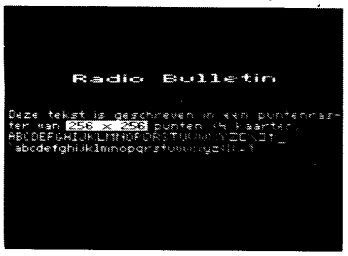
gister in ieder geval tijdens de oneven lijnen geen informatie ontvangen. Op dit moment zouden immers kaartjes 3 en 4 geselecteerd zijn. De printvoet is echter leeg, waardoor de datalijnen 'open hangen'. In het schuifregister worden nu 8 logische enen ingeklokt. Later zal dit worden geïnterpreteerd als 'zwart'. We zien dus dat alle oneven lijnen uit het raster verdwijnen, zodat we van de 256 lijnen nog maar 128 lijnen over hebben.
Horizontaal gebeurt hetzelfde. Doordat kaartje 1 niet aanwezig is, zullen de bits D0, D2, D4 en D6 '1' worden, waardoor er altijd onvoorwaardelijk 128 stippen op één lijn zwart blijven. Op de even lijnen blijven dus ook maar 128 stippen over. Het resultaat is een puntenraster van 128 x 128 punten. Met puntenraster bedoelen we hier echt 'punten'. Een vol wit vlak is niet mogelijk. In afb. 9 staat een tekst geschreven in dit puntenraster, en zoals u ziet is dit raster voor letters zeer acceptabel. Met een kleine ingreep kunnen we de punten groter maken, zodat vol wit wel mogelijk is (afb. 10). Na verloop van tijd kan het geheugen worden uitgebreid tot 4 kaartjes, waardoor eën raster van 256 x 256 puntjes ontstaat (afb. 11 en 12).

De adressen

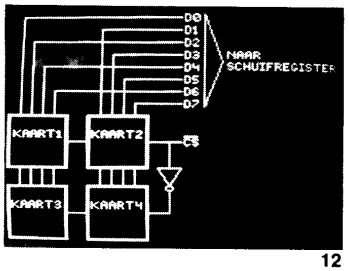
In afb. 13 ziet u de adresaansturing van de geheugenkaartjes. Tijdens het zichtbare deel van het TV-beeld is 'lijnblanking' een logische '0'. Hierdoor zullen de multiplexers (3 x 74157 en 1 x 7400) de door de syncgenerator gegenereerde adressen naar de geheugenkaartjes doorkoppelen. Omdat het geheugen 8 bits gelijktijdig genereert hoeven de adressen H0 . . . H2 niet aan het geheugen te worden toegevoerd. Tijdens het zichtbare deel van een lijn worden alle H-adressen doorlopen, tijdens het zichtbare deel van het beeld worden alle V-adressen doorlopen. Op deze manier heeft ieder puntje op het scherm z'n eigen adres.
Wanneer we de inhoud van het geheugen willen veranderen, moeten we de mogelijkheid hebben om het adres van het te veranderen bitje aan het geheugen toe te voeren. Dit mag nooit tijdens het zichtbare deel van het TV-beeld, omdat dan opeens midden in het beeld de inhoud van een verkeerd adres verschijnt. Daarom is de lijnblanking met de stuuringang van de multiplexers verbonden. Hierdoor zal gedurende de lijnterugslag altijd het adres dat in de 4 tellers (4 x 74193) staat aan het geheugen worden toegevoerd. Dit gebeurt ook als we helemaal niets op het TV-scherm willen veranderen, om de eenvoudige reden dat de hardware hierdoor eenvoudiger en dus goedkoper is. De 4 tellers worden hier gebruikt als buffergeheugen tussen de computer en het TV display. Bij normaal bedrijf wordt de telfunctie van de 74193 niet gebruikt. De enige reden

9

10



11



12

afb. 9. Wanneer we maar één geheugenkaartje gebruiken, ziet de tekst er zo uit.
afb. 10. Met een kleine wijziging worden de stippen 4 × zo groot.
afb. 11. Een tekst in een raster van 256 × 256 punten. Alle letters nemen net zoveel ruimte in als zij nodig hebben.
afb. 12. Een voorbeeld van een tekening op het scherm.

dat er tellers gekozen zijn, is om de mogelijkheid van handbediening open te houden. Hierop komen we nog terug. De adressen h 0...h 2 worden niet naar de geheugenkaartjes gevoerd, maar naar de multiplexer en demultiplexer uit afb. 10. Met deze drie adreslijnen bepalen we welke van de 8 geheugenbitjes veranderd moet worden. Aan de 'buitenkant' van het TV display lijkt het nu net of het display een geheugen heeft van 64 k x 1 bit i.p.v. 8 k x 8 bit. Hierdoor wordt de software voor het display een stuk eenvoudiger.

Verder zien we in afb. 13 dat de CS van de geheugenkaartjes via de multiplexer verbonden is met V0. Dit houdt in dat het ene stel geheugenkaartjes geselecteerd is gedurende de even lijnen van het TV-scherm en het andere stel geheugenkaartjes gedurende de oneven lijnen van het scherm. Hierdoor kunnen we gemakkelijk overschakelen naar 128 lijnen (goedkope versie). Een en ander is reeds duidelijk gemaakt.

Om het aantal aansluitingen tussen computer en display te beperken, worden de horizontale en verticale adressen over dezelfde 8 lijnen doorgegeven. Hierdoor kan het hele display op één PIA (2 x 8 bits) worden aangesloten.

Stuursignalen

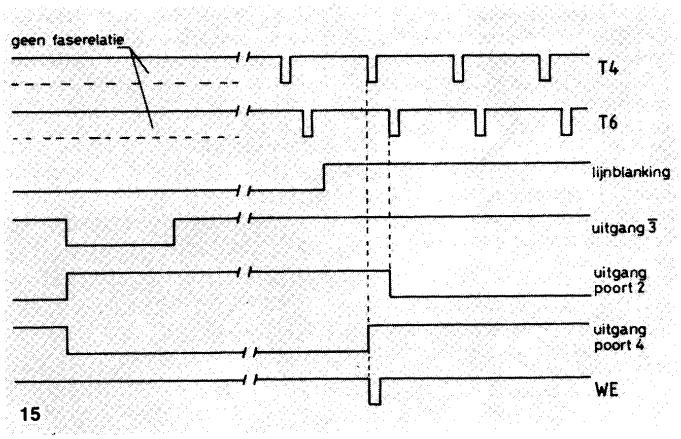
Natuurlijk moet de computer stuursignalen geven, die aangeven of het aangeboden adres een horizontaal of verticaal adres is. Verder moet de mogelijkheid bestaan om het hele scherm in één keer wit of zwart te maken (clear).
In afb. 14 ziet u hoe een en ander gerealiseerd is. Een BCD-decimaal decoder (7442, links boven) decodeert 3 stuurlijnen van de computer. Afhankelijk van het binaire getal dat de computer geeft hebben we de volgende functies:

C B A	
0 0 0	geen actie
0 0 1	zet adres in X buffer
0 1 0	zet adres in Y buffer
0 1 1	wijzig het geadresseerde bitje
1 0 0	clear het hele display
1 0 1	reserve voor toekomstige uitbreidingen
1 1 0	
1 1 1	

Toestand '0 0 0' is de ruststand, er worden geen adressen aan het display doorgegeven, en er wordt niet geschreven. Uitgang $\overline{0}$ van de decoder is dan ook nergens op aangesloten. Zo-

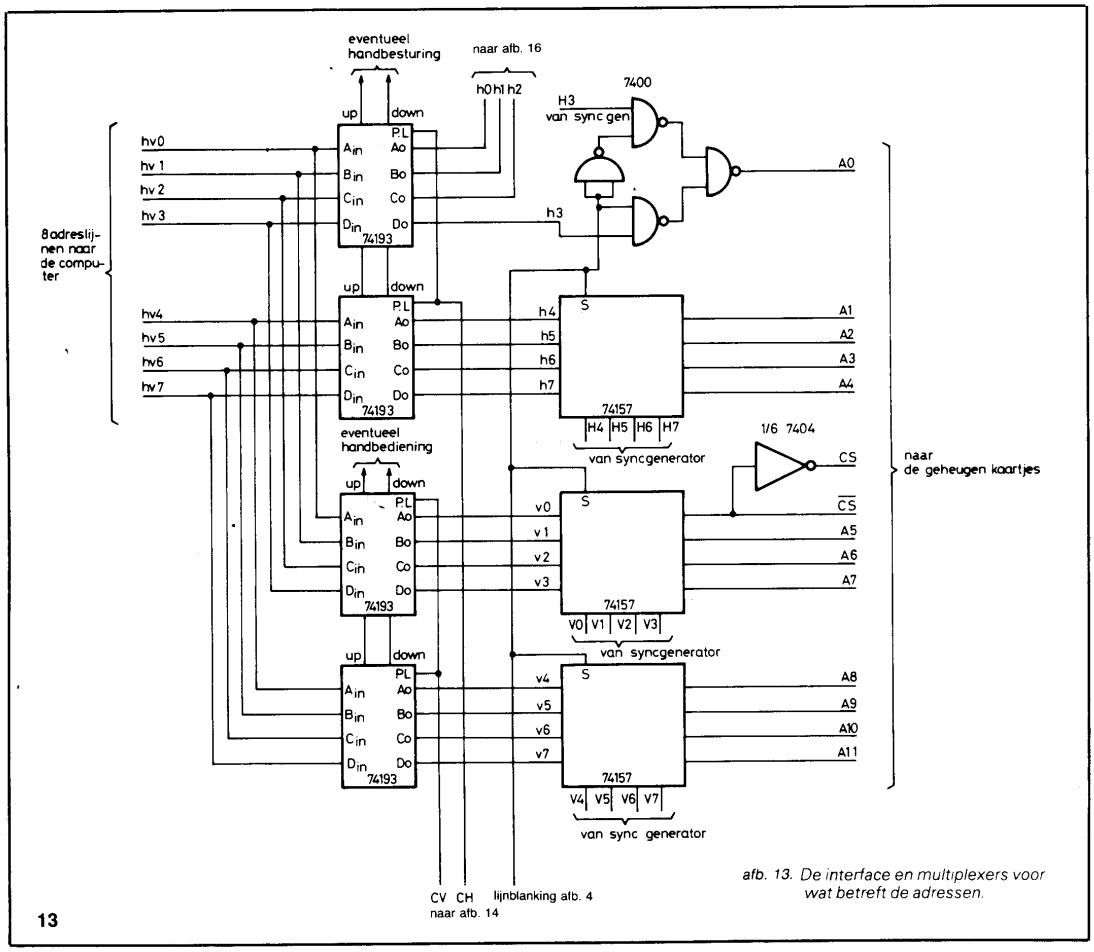
dra de computer het getal '0 0 1' door-, geeft, zal uitgang $\overline{1}$ van de decoder laag worden. Deze uitgang is verbonden met de clock van de X buffer (afb. 13), zodat het horizontale adres geklokt wordt. Bij toestand '0 1 0' gebeurt hetzelfde voor het verticale adres. Natuurlijk moet steeds het gewenste adres op het juiste moment op de ingangsadressen hv0 . . . hv7 (afb. 13) worden gezet. Wanneer in de X- en Y-buffer het juiste adres is opgeslagen kunnen we een 'actie' commando geven door op de stuurlijnen (afb. 14) het getal '0 1 1' te zetten. Het gevolg van dit commando moet zijn dat er één WE-puls gegenereerd wordt tijdens de lijnblanking. In afb. 14 zien we de schakeling die dit verzorgt.

De neergaande flank op uitgang $\overline{3}$ van de bovenste 7442 zorgt dat er 2 flip-flops worden geset. De eerste flip-flop, gemaakt met poorten 1 en 2, zal aan de uitgang van poort 2 '1' worden.

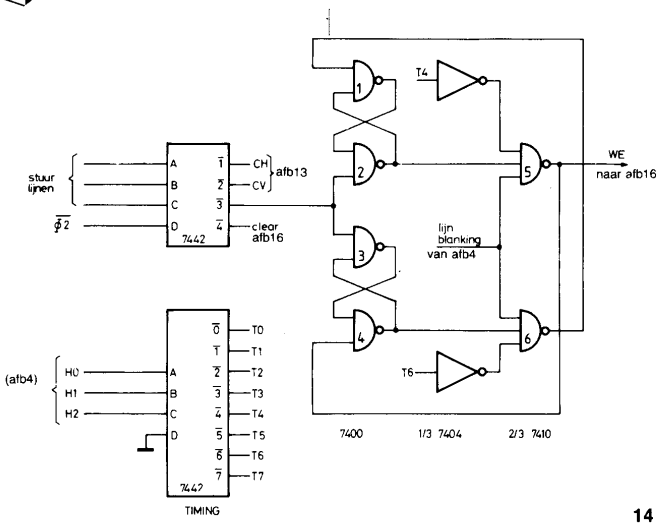


afb. 15. Het opwekken van één WE-puls.

Zodra de lijnblanking hoog wordt (pas dan wordt het adres in de X- en Y-buffer naar het geheugen geschakeld) zal de puls T4 aan de uitgang van poort 5 verschijnen. Deze puls wordt gebruikt als WE-puls. Met het verschij-



afb. 13. De interface en multiplexers voor wat betreft de adressen.



14

afb. 14. De interface voor de stuursignalen.

nen van de WE-puls wordt de FF rond poorten 3 en 4 weer gereset. Hierdoor zal de puls T6 via poort 6 de eerste fliptiop (poorten 1 en 2) resetten. De timing van deze signalen is nog eens duidelijk getekend in afb. 15.

Het veranderen van een willekeurige stip op het scherm vereist dus 3 handelingen (clock X, clock Y en 'actie'). Om de sturing eenvoudig te houden hebben we hier expres de opeenvolgende codes 0 0 0... 0 1 1 voor genomen, zodat we steeds met één 'increment-instructie' (ophoog instructie) van de ene toestand in de andere toestand kunnen komen.

In de praktijk komt het wel eens voor dat wanneer de stuurcode van '0 0 1' naar '0 1 0' gaat, een zeer korte tijd de toestand '0 1 1' ontstaat. Dit zou een valse 'actie'-puls opleveren. Dat is de reden dat we klokpuls $\overline{\emptyset 2}$ uit het computersysteem op de D-ingang van de decoder zetten. Zolang de D-ingang hoog is, zullen de uitgangen 0... 7 ook altijd hoog zijn. De $\overline{\emptyset 2}$ puls wordt pas weer laag als de omschakeling van 001 naar 010 heeft plaatsgevonden. Op deze manier kan dus nooit een valse puls ontstaan.

Clear

Wanneer we op de stuurlijnen de code '100' zetten, zal uitgang 4 van de BCD-omzetter laag worden. Deze uitgang is doorverbonden met 'clear' uit afb. 16. We zien dat hierdoor (via 8 EN-poorten) alle WE-lijnen naar de IC's naar '0' worden getrokken. Indien we zorgen dat deze toestand minstens 20 ms gehandhaafd blijft, zal het hele display worden volgeschreven met een '0' of een '1'.

Hierdoor zal het hele scherm wit of zwart worden. De 20 ms puls moet met software worden gemaakt. Hierdoor hoeven we geen one shot van 20 ms in de schakeling op te nemen.

De kritische lezer zal het opgevallen zijn dat uitgang 4 van de BCD-omzetter (afb. 14) nooit 20 ms laag kan blijven omdat de 02 clockpuls continu de D-ingang van deze omzetter hoog en laag maakt. Zolang de clockfrequentie van het computersysteem maar groter is dan ca. 1 MHz, en kleiner dan 10 MHz werkt dit systeem goed, omdat er altijd wel een puls binnen elke geheugencyclus van 1.6 us valt. De meeste hobbycomputers werken met frequenties tussen deze grenzen.

De multiplexer (7442) in afb. 16 zorgt ervoor dat de WE-puls bij het juiste geheugen-IC terechtkomt. Op deze manier schrijven we maar in 1 van de 8 geadresseerde bits. (Een geheugenadres bevat 8 bits). In welke bits uiteindelijk geschreven wordt, wordt bepaald door de laagste 3 horizontale adressen (ho . . . h2, afb. 13).

De datalijnen

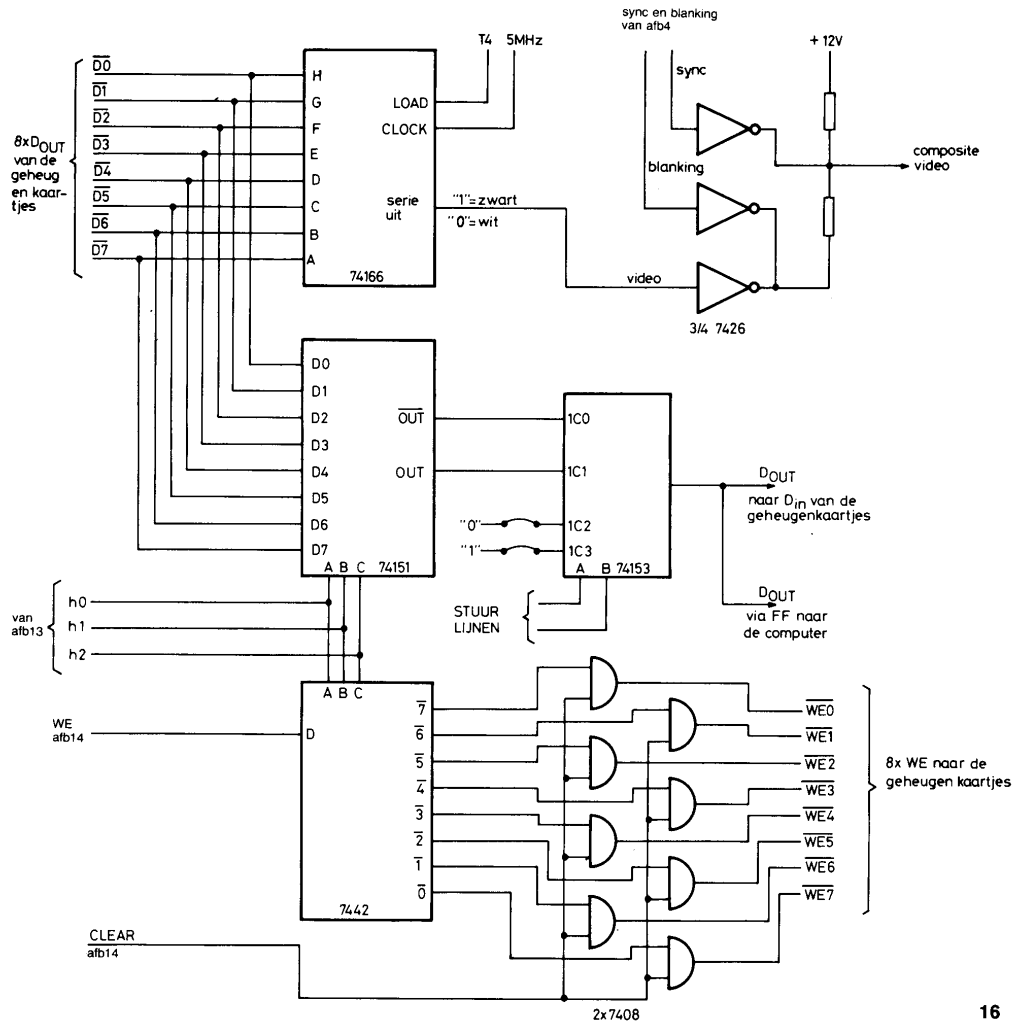
De 8-bits-data, welke het geheugen eens in de 1,6 us levert, wordt aan een schuifregister toegevoerd (74166 in afb. 16). De schuiffrequentie van het schuifregister is 5 MHz, zodat er elke 200 ns een 'bitje' aan de uitgang verschijnt. De computer moet ook de mogelijkheid hebben om een bitje van het scherm terug te lezen. Daarom wordt de 8-bits-data ook op de multiplexer (74151) gezet. Met de drie laagste horizontale adresbits wordt nu weer bepaald welke van de 8 bitjes doorgegeven moet worden. Zoals al eerder opgemerkt, merken we hierdoor niets van de inwendige 8-bits-structuur. Op de uitgang van de 74151 hebben we nu zowel $\overline{\text{Dout}}$ als Din tot onze beschikking. Wanneer we $\overline{\text{Dout}}$ doorkoppelen aan Din, ontstaat de interessante

Voor de duidelijkheid

Deze bouwbeschrijving leidt tot een display waarvan 65536 stippen afzonderlijk aan- en uitgezet kunnen worden. Met behulp van een geschikt programma kunnen alle symbolen van elke breedte op elke willekeurige plaats op het schema worden gezet. Lettervormen en -formaten kunnen zelf worden bepaald. Natuurlijk kunnen ook figuren worden getekend. De benodigde programma's zullen zeker voor de 6502 en waarschijnlijk voor de 6800 en 8080 gepubliceerd worden. De prints welke bij dit onderwerp zijn gebruikt zullen in onze printservice worden opgenomen. In het ontwerp is gebruik gemaakt van de gewone TTL (ongeveer voor *f* 100.-). Voor de geheugenchips hebben we de standaard 22-pins 4 K dynamische RAM genomen (NEC 411D of 2107 van Intel). De goedkope versie heeft ongeveer 80.- geheugen, de duurdere versie (256x256) heeft ongeveer 320.- geheugen.

Printen

Het basispakket (moederprint, syncprint, interfaceprint en één geheugenprint) kan worden besteld door overmaking van *f* 68,70 op gironummer 83214 ten name van Uitgeverij de Muiderkring, onder vermelding van „RB7484/87". Losse geheugenkaartjes kunnen worden besteld door overmaking van *f* 8,55 per print onder vermelding van „RB7486". Wanneer u het display wilt uitvoeren met 4 geheugenkaartjes, bestelt u dus: 1 x RB7484/87 en 3 x RB7486. Goedkope 5MHz kristallen worden geleverd door Holland Electronics, Leiden.



afb. 16. Zo wordt in het geheugen geschreven en gelezen.

mogelijkheid om één stip te inverteren. De stip wordt dus wit als hij zwart was, en zwart als hij wit was. Met behulp van nog een multiplexer (74153) kunnen we nu kiezen uit 4 mogelijkheden:
B A

0 0: schrijf dezelfde data
0 1: schrijf geïnverteerde data
1 0: schrijf '0'
1 1: schrijf '1'

We zien dat er nog 2 stuurlijnen voor de computer bijkomen. De oorspronkelijke bedoeling was om data naar de computer terug te lezen. In dit geval moet de $\overline{WE}$-puls eigenlijk niet naar het geheugen gevoerd worden (er mag immers niets geschreven worden), maar naar een uitgangsflip-flop. Deze flip-flop moet dan de toestand van het uitgelezen bitje onthouden. Alweer om de hardware zo eenvoudig mogelijk te houden, voeren we **altijd** de $\overline{WE}$-puls zowel aan het geheugen als aan de uitgangsflip-flop. Wanneer we niet willen schrijven maar lezen, moeten we op de bovengenoemde stuurlijnen '00' zetten. Hierdoor wordt de oude data in het geheugen geschreven, zodat de inhoud (ondanks de $\overline{WE}$-puls) niet verandert. Op een later tijdstip kan de computer dan de toestand van de flip-flop bepalen. Met de 3 stuurlijnen die we eerder in het verhaal genoemd hebben kunnen we dus bepalen waar het adres terecht komt (X- of Y-buffer), en wanneer het display moet gaan schrijven.

Met de 2 bovenstaande stuurlijnen kunnen we bepalen of de geadresseerde punt wit, zwart of geïnverteerd wordt of dat de geadresseerde stip van het scherm wordt gelezen.

Opbouw

Het complete display bestaat uit een moederprint, waarin m.b.v. printconnectoren een interfaceprint, een syncprint en één, twee of vier geheugenkaartjes kunnen worden gestoken. Elke print wordt recht gehouden m.b.v. een printgeleider. Op de foto (afb. 17) kijken we tegen de koperzijde van de interfaceprint. Net achter deze print is de syncprint te zien. De vier geheugenkaartjes staan met de componenten naar voren opgesteld. Om het geheel zo goedkoop mogelijk te houden zijn alle printen enkelzijdig uitgevoerd. Hierdoor waren de doorverbindingen op de print niet te vermijden, maar gezien de besparing die dit oplevert, kan dat nauwelijks een bezwaar zijn. Bovendien is het voordeel van doorverbindingen, dat uitbreidingen en wijzigingen vaak gemakkelijker zijn te realiseren, zodat de opbouw flexibel blijft. De printconnectoren (DIN 41617) kunnen eventueel worden vervangen door vaste draadverbindingen; bedenk echter wel dat reparaties, uitbreidingen e.d. dan wel lastiger zijn uit te voeren.

Een TV met modulator, of een TV als monitor?

De eerste stap bij het bouwen van het grafisch TV-display is het geschikt maken van de TV voor videosignalen. Verreweg de eenvoudigste manier om dit te bereiken, is het toepassen van een modulator, zodat het videosignaal op een draaggolf wordt gemoduleerd. Aan de TV hoeft dan niets te worden gewijzigd. De redactie heeft een aantal in de handel zijnde modulatoren getest. Zeer redelijke resultaten werden bereikt met de TV-1 (bouwpakket) van UHF- associatés, in Nederland geleverd door Ingenieursbureau Koopmans in Papendrecht. Echter géén van de modulatoren voldeed aan de gestelde normen betreffende modulatiediepte wit-niveau (10%), onderdrukking van harmonischen e.d. Het hoeft dan ook geen betoog, dat de mooiste oplossing wordt verkregen met een 'echte' monitor-ingang. De moeilijkheid die zich hier voordoet is, dat eigenlijk géén algemene regel te geven is voor het ombouwen van een TV tot monitor. Deze ingreep is dan ook beslist af te raden, wanneer u niet voldoende op de hoogte bent met de manier waarop een TV werkt. De enige richtlijn die we kunnen geven is dat de monitoraansluiting moet komen ná de laatste demodulator, dus het punt waar in de TV voor het eerst een videosignaal verschijnt.
Bij buizentoestellen verloopt de ingreep het gemakkelijkst. In dit geval is het meestal voldoende om het stuurrooster van de video-eindbuis los te koppelen van de voorgaande elektronica, en de monitoraansluiting op dit punt aan te brengen. Bij transistortoestellen moeten we vaak nog gelijkspanningsniveaus herstellen. Wanneer u uw huiskamer-TV gebruikt verdient het aanbeveling een schakelaar in te bouwen om te kiezen tussen TV en monitor.
Vaak is het mogelijk om voor bedragen van *f* 10,– tot *f* 50,– goedwerkende buizen-TV's te kopen, zodat u naar hartelust kunt knippen en solderen, zonder het risico te lopen dat u ruzie krijgt met uw vrouw en/of kinderen, omdat zij 's avonds geen TV kunnen kijken.

Waarschuwing: De meeste TV's hebben geen voedingstrafo, zodat het chassis spanningvoerend kan zijn. Het omdraaien van de steker is de goedkoopste oplossing, een scheidingstrafo is verreweg de veiligste oplossing. Het spreekt vanzelf dat wij de laatste methode aanbevelen. Het video-signaal is vrij fors gemaakt: 12 Vtt. Wanneer deze spanning te hoog is (transistor-TV's) moet een spanningsdelertje zorgen voor de juiste uitgangsspanning.

De voedingen

De voeding valt een beetje buiten dit ontwerp, de voedingsspanningen (+12, +5 en −5 V) zijn de 'gewone' spanningen, die haast elke computer hobbyist ter beschikking heeft. Wanneer u het display een eigen voeding wilt geven, moet u voor deze schakeling rekening houden met 0,5 A voor de 12 V; 0,7 A voor de 5 V en slechts 1 mA voor de −5 V.

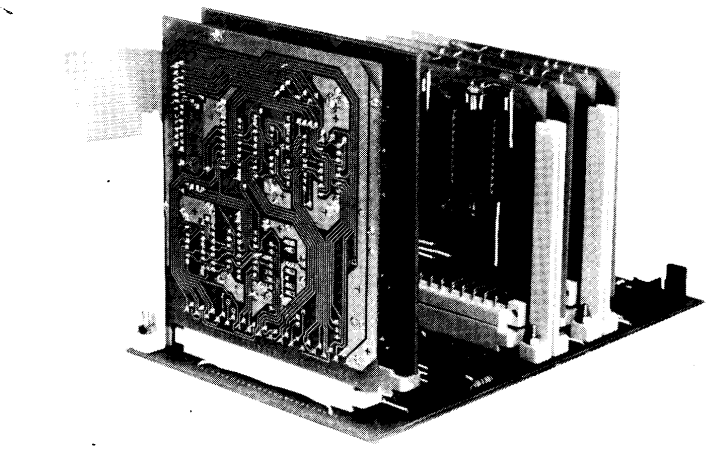
Wanneer u het display wilt uitbreiden met een eigen microcomputersysteem, is het raadzaam de +12V geschikt te maken voor 1 A en de +5 V voor 2 A.

De syncpulsenprint

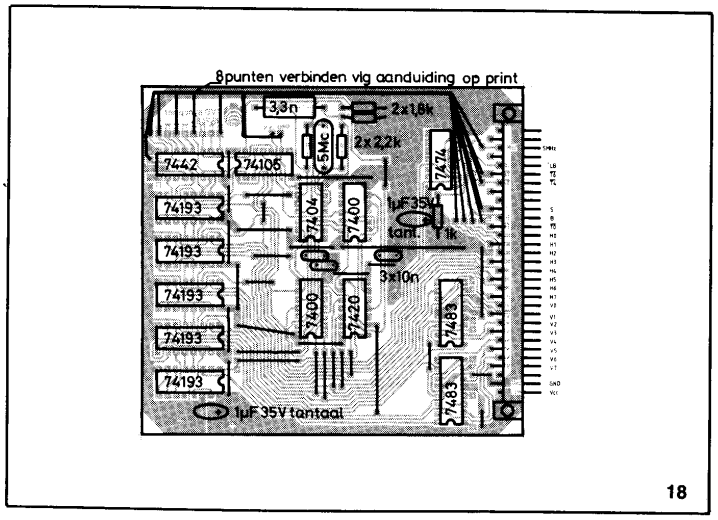
Wanneer u de TV zover klaar heeft dat u met video in kunt gaan (al dan niet via een modulator) kunnen we beginnen met het in elkaar zetten van de syncpulsenprint. Deze print genereerd de benodigde sync- en blankingpulsen voor de TV en verzorgt bovendien de adressen voor de geheugenkaartjes. Op de print is de mogelijkheid om twee 4-bits full adders extra te plaatsen. Hierdoor ontstaat de mogelijkheid voor een hardware scroll-up (de verticale adressen worden met een 8-bits getal verhoogd, waardoor het hele beeld in verticale zin 'doordraait').
Voorlopig kunnen we de adreslijnen gewoon doorlussen, of (naar wens) de 2 IC's monteren maar verder ongebruikt laten. In de toekomst kan een ingebouwde microprocessor deze scroll-up verzorgen. Het blankingsignaal, dat zorgt dat de elektronenstraal op de TV wordt onderdrukt tijdens de terugslag, liep in ons prototype niet geheel op tijd.

Op het moment dat de horizontale adressenteller op 0000 0000 staat (helemaal links op het scherm) geeft het blankingsignaal de elektronenstraal vrij. Het duurt dan echter nog ca. 800 ns vóórdat het schuifregister wordt ingeklokt, en bit 0 echt op het scherm verschijnt.
Het 'zichtbare' adres is dus ca. 800 ns verschoven t.o.v. het adres in de tel-

afb. 17. Zo ziet het display er na montage uit.



afb. 18. De componentenopstelling van de syncprint.
afb. 19. De moederprint.

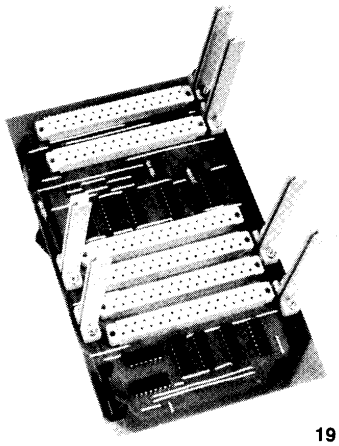


18

lers. Hierdoor moet ook het blankingsignaal worden verschoven. Daarom werd aan deze, definitieve versie 'n flipflop toegevoegd, welke het blankingsignaal op T4 inklokt. Hierdoor wordt de gewenste verschuiving verkregen.
In afb. 18 vindt u de componentenopstelling van syncprint. Om IC 7442 (links boven) vindt u de aansluitingen T0 . . . T7. Normaal gebruiken we hiervan alleen T0 (CE), T4 (WE) en T6. Wanneer we echter met langzamere of snellere RAM's werken kunnen we op zeer eenvoudige wijze de timing verschuiven.

De moederprint (afb. 19)

Voordat de syncprint kan worden getest, moeten we eerst de moederprint in elkaar zetten. Om kosten te besparen zijn de gaten voor de printgeleiders op dezelfde diameter geboord als de gaten van de connectors.
Vóór met de montage wordt begonnen moeten deze gaten eerst worden geboord met een 3 mm boor. Vervolgens alle doorverbindingen aanbrengen, daarna de IC's monteren en als laatste de connectors (DIN 41617) solderen. Denk er aan dat bij het monteren van de bovenste printgeleiders isolatieringetjes moeten worden gebruikt, om sluiting tussen de onderliggende voedingssporen te voorkomen. In afb. 20 vindt u de componentenopstelling van de moederprint. Wanneer u moederprint en syncprint af heeft, komt de eerste testfase aan de beurt. De TV moet op de uitgang van het display



19

worden geschakeld en de voedingsspanningen moeten worden aangesloten. Denk erom dat het 12 V videosignaal via een voltmeter van ca. 1kohm teruggebracht moet worden tot ca. 1 V. Later kan dan experimenteel een goede waarde worden bepaald. Na het inschakelen moeten eerst de voedingsspanningen worden gemeten (sluiting in een van de printen?). Wanneer de spanningen goed zijn moet op de TV een egaal raster verschijnen met *stilstaande* lijnen. Wanneer u twijfelt kunt u de video-aansluiting even los nemen, en het verschil bekijken. Wanneer het raster niet goed is zit de fout waarschijnlijk in de syncprint. In dit geval moet de syncprint worden gecontroleerd op sluiting, defecte of verkeerde IC's, verkeerde doorverbindingen, enz. Vanaf het moment dat het raster goed is, kan het display worden nagekeken met het display als meter!

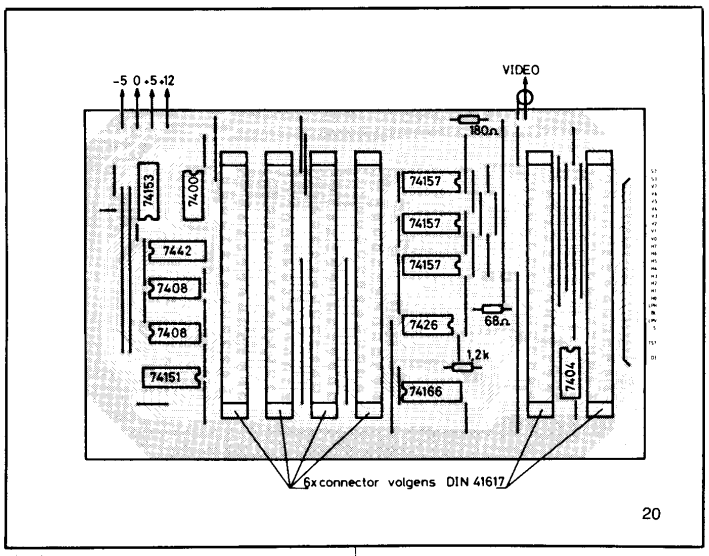
Het via de TV controleren van de connectoren

Het display is zo ingericht dat met een extra video-ingang het display kan worden gecontroleerd. Een logische '1' geeft een zwart beeld, een logische '0' geeft een wit beeld. Blokgolven geven een balkenpatroon, afhankelijk van de frequentie horizontaal of verticaal. De extra video ingang kan *alleen* digitale signalen aan. We mogen deze ingang dus nooit aan de +12 of de −5 V leggen. Helemaal rechts op de print (afb. 20) bevinden zich de aansluitingen van de moederprint. Een van deze aansluitingen is gemerkt met 'TST', de afkorting van 'Test'. We monteren nu tijdelijk een meetdraad aan dit punt. Aan het uiteinde komt een dikker stukje draad, dat in de connector past. We beginnen met de connector voor de interfaceprint (uiterst rechts) op punt 31 (bovenaan):

PUNT	SIGNAAL	BEELD
31	Vcc	zwart
30	GND	wit
29 t/m 8	ingangen	zwart
7	$\overline{T4}$	dunne zwarte verticale strepen
6	$\overline{T6}$	idem, niet op dezelfde plaats
5	LB	wit
4	d4	wit
3...1	ingangen	zwart

Wanneer er bij één van de pennen 29...8 en 1...3 een balkenpatroon op het scherm verschijnt, zal het spoortje vanaf dit connectorpunt onderweg sluiting maken met een naast liggend spoortje. Een dergelijke fout kan gemakkelijk optreden bij het solderen van de connectoren. Zodra de signalen voor de interfaceprint in orde zijn, gaan we verder met de connectoren voor de geheugenkaartjes. De test moet voor elk van de 4 geheugenconnectoren worden uitgevoerd. We beginnen weer bovenaan, bij punt 6. Punt 1 en 5 mogen *niet* worden gemeten (+12 en −5 V!!). Om vergissing te voorkomen beginnen we bij punt 6:

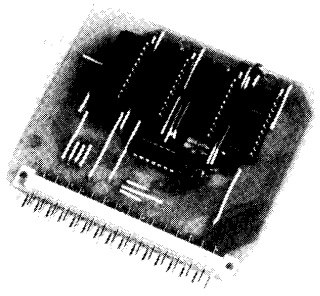
PUNT	SIGNAAL	BEELD
6	A11	boven wit, onder zwart
7	A10	twee horizontale balken
8	A9	4 horizontale balken
9	A8	8 horizontale balken



afb. 20. De componentenopstelling van de moederprint.
afb. 21. Een geheugenkaartje.

10	A7	16 horizontale balken
11	A6	32 horizontale balken (elke balk 4 lijnen dik)
12	A5	balken van 2 lijnen dik
13	$\overline{CS}$	'balken' van 1 lijn dik
14	A4	1 witte verticale balk
15	A3	2 witte verticale balken
16	A2	4 witte verticale balken
17	A1	8 witte verticale balken
18	A0	16 witte verticale balken
19,20	$\overline{WE2}$, $\overline{WE1}$	zwart
21	NC	zwart
22	CE	dunne verticale witte lijnen
23, 24	$\overline{WE3}$, $\overline{WE4}$	zwart
25, 26	$\overline{Dout\ 1}$, $\overline{Dout\ 2}$	zwart
27	NC	zwart
28, 29	$\overline{Dout\ 4}$, $\overline{Dout\ 3}$	zwart
30	DIN	wit

21



Het is mogelijk dat het aantal balken niet exact klopt, doordat er één of meer balken buiten beeld vallen. Wanneer alle connectoren de goede signalen afgeven moeten we de ingangen nog even controleren. Het meetdraadje moet nu van 'TST' worden losgemaakt en aan massa worden gelegd. Vervolgens moet van elke geheugenconnector achtereenvolgens de punten 25, 26 en 28, 29 aan massa worden gelegd. Bij elk van deze punten moeten dunne witte verticale strepen op het scherm verschijnen. Een fout bij deze laatste test wijst op een fout bij het schuifregister (74166). Wanneer alles tot zover in orde is, beginnen we met de geheugenkaartjes.

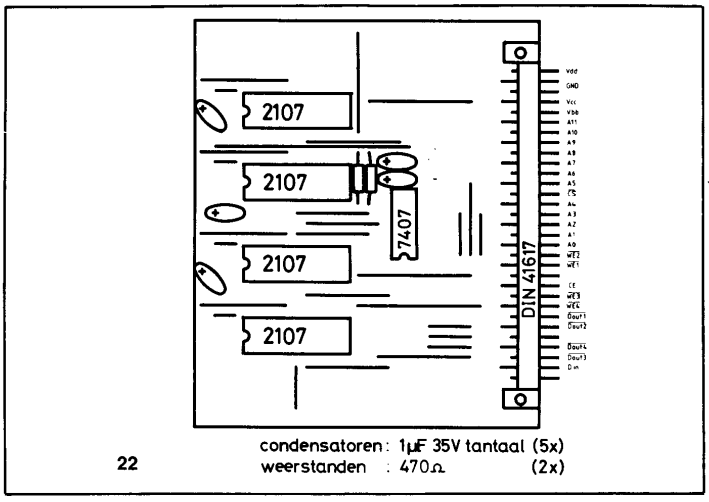
De geheugenkaartjes (afb. 21)

Voorlopig maken we één geheugenkaartje. De componentenopstelling vindt u in afb. 22. Op de foto is te zien dat de 4 k RAM's in voetjes zijn gemonteerd. Nodig is dat natuurlijk niet. Het redactieexemplaar werd getest met de intel 2107C-4 en met de NEC D411D-1. Bij de TMS 4060 NL was inverteren niet mogelijk. Ook bij de geheugenkaartjes is een kleine modificatie toegepast. Het IC 7407 (6 buffers, open collector) had nog 4 buffers over. Deze buffers werden achter de 4 data-uitgangen geschakeld. Echt nodig is dit niet, het voordeel is dat de RAM's iets minder zwaar worden belast. Bovendien zal bij een fout in de chip-select worden voorkomen dat de RAM's defect raken. We hebben nu immers open collectoruitgangen i.p.v. de three state-uitgangen van de RAM's. Wanneer het kaartje klaar is kunnen we het (met uitgeschakelde voeding!) in de daarvoor bestemde connector steken. Na inschakelen zal een willekeurig puntjespatroon op het scherm verschijnen. Om het display te clearen moeten we punt 13 van de interfaceconnector even aan massa leggen. Wanneer alles tot zover werkt kunnen we beginnen met de interfaceprint.

De interfaceprint (afb. 23)

De laatste print is de interfaceprint (zie afb. 24), de schakel tussen computer en display. Wanneer u deze print in elkaar heeft gezet, gaan we met de hand een adres invoeren. Het geheugenkaartje moet in de meest linkse voet worden gestoken. Om de kaart te testen moeten we draadjes aan hv 7, A1, B1, C1 en A2 solderen. Verbind Ø2 via een schakelaartje met massa. Na inschakelen zal een willekeurig patroon op de TV verschijnen. De lijnen A1, B1 en C1 bepalen welke actie wordt verlangd, de lijnen A2 en B2 bepalen of er moet worden gelezen, geïnverteerd wit schrijven of zwart schrijven. Allereerst moeten we het display clearen. We voeren aan C1, B1 en A1 de code 100 toe door A1 en B1 aan massa te leggen. Op het moment dat we de schakelaar even sluiten (Ø2 aan massa) zal het display zwart worden. Alvorens de verbindingen te verwijderen moet het schakelaartje weer open worden gezet. Nu gaan we adres $ 7F in register X zetten: Met code 001 kunnen we register X laden. We maken hv 7, C1 en B1 een logische '0' en halen de schakelaar een keer heen en weer. Hetzelfde adres moet in register Y komen. Om dit register te vullen moeten we de code 010 toevoeren. Dus C1, A1 en hv 7 weer aan massa en de schakelaar heen en weer doen. Nu staat adres $ 7F7F in het buffergeheugen. Wanneer we een witte

afb. 22. Componentenopstelling van het geheugenkaartje.
afb. 23. De interfaceprint, de schakel tussen computer en display.
afb. 24. De componentenopstelling van de interfaceprint.



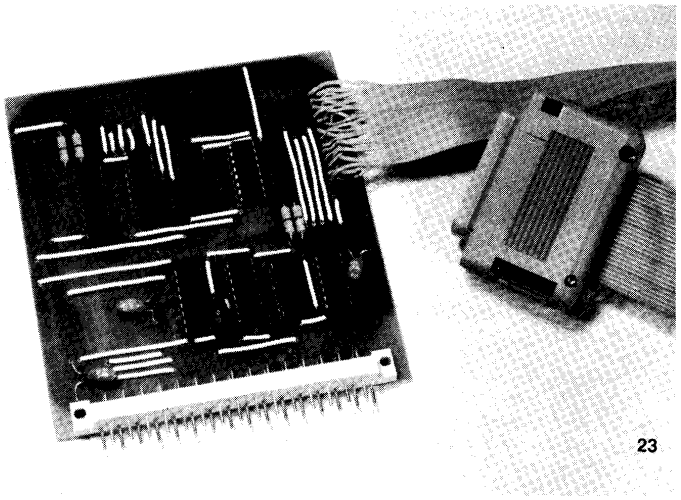
22

condensatoren: 1µF 35V tantaal (5x)
weerstanden : 470Ω (2x)

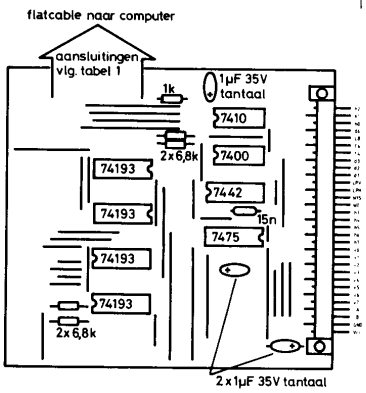
punt willen schrijven moeten we aan B2, A2 de code 10 toevoeren. Dus, A2 aan massa. Vervolgens moet aan C1, B1, A1 de code voor schrijven worden gegeven (011), dus C1 aan massa. Bij het heen en weer halen van de schakelaar moet er een witte stip op het midden van het TV-scherm verschijnen.

Software

Bovenstaande handelingen worden in de praktijk veel sneller gedaan door de microcomputer. Wanneer u zelf vast wat wil proberen: De lijnblanking van het display wordt via de flat cable aan de computer toegevoerd. Wanneer de computer een punt wil schrijven moet hij wachten tot deze syncpuls geweest is.



23



De verbindingen tussen computer en display

Het display moet worden aangesloten op de I/O van de computer. In tabel 2 vindt u de aansluitingen van het display. Het doel van deze aansluitingen zullen we nog even kort toelichten.

DE SIGNALEN hv0...hv7
Deze signalen (voor de computer: uitgangen) geven een 8-bits adres door. Met dit adres kan de plaats op het TV-scherm worden gedefinieerd. Het adres kan, afhankelijk van de stand van de stuurlijnen A1...C1, een horizontaal of een verticaal adres zijn. Het horizontale adres loopt van 00 (links) tot FF (rechts). Het verticale adres loopt van 00 (boven) tot FF (onder).

DE SIGNALEN A1...C1
Met deze drie stuurlijnen (voor de computer uitgangen) geven we het display door wat we willen doen. We hebben de keus uit het inklokken van een horizontaal of verticaal adres, het schoonmaken van het hele schema en het geven van een lees/schrijf commando. De overige bitcombinaties worden (voorlopig?) niet gebruikt. **Tabel 3 geeft aan welke actie bij welk bitpatroon wordt genomen.**

DE SIGNALEN A2, B2
Deze stuurlijnen (voor de computer uitgangen) geven aan welke 'kleur' de stip moet krijgen. Met kleur bedoelen we hier zwart, wit, inverteren of lezen. Bij inverteren wordt de stip wit als hij

zwart was, en zwart als hij wit was. Bij lezen komt er over de datalijn een '1' of een '0' terug, afhankelijk of de geadresseerde stip zwart of wit was. In **tabel 4 ziet u welke kleur bij welke bit-combinatie hoort.**

HET SIGNAAL $\overline{\Phi 2}$
Dit is het enige signaal dat niet op de I/0 van de computer komt. Het moet aangeven wanneer de data op de stuurlijnen geldig is. Het gebeurt wel eens dat bij de overgang van bijvoorbeeld 001 naar 010 voor zeer korte tijd de toestand 011 ontstaat, waardoor ten onrechte een 'schrijf'-commando wordt gegenereerd. De $\overline{\Phi 2}$ moet in het systeem worden aangesloten op een punt dat tijdens deze overgang hoog is. Bij 6502 en 6800 systemen zal dit $\overline{\Phi 2}$ zijn. Pas als $\overline{\Phi 2}$ laag wordt zal de aangeboden data op de stuurlijnen worden gedecodeerd.

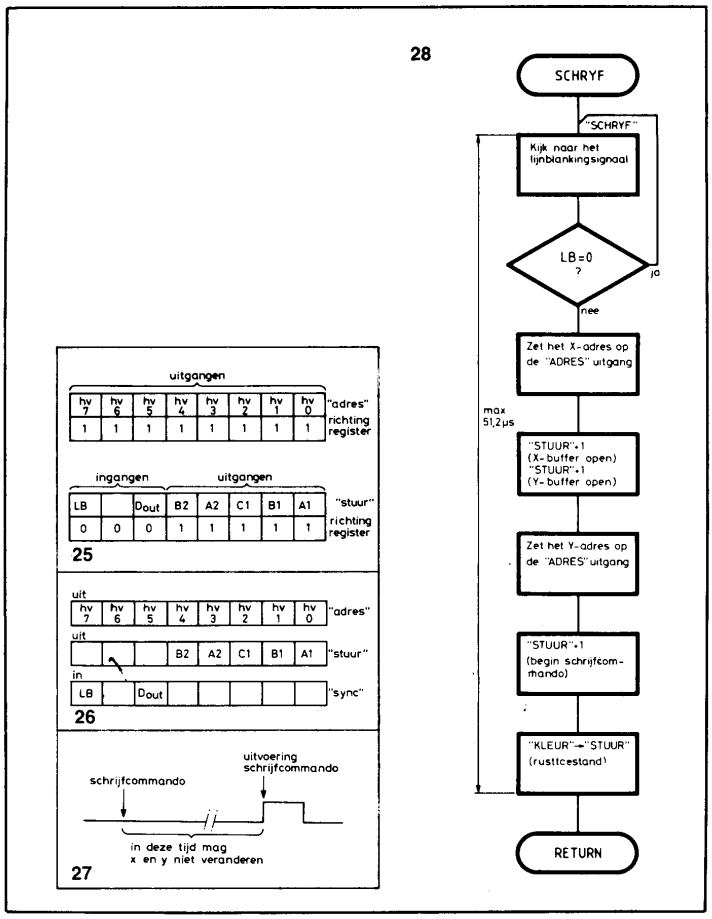
Dout
Dit signaal (voor de computer ingang) wordt uitsluitend gebruikt bij lezen. Dout zal 1 worden bij een zwarte stip, en '0' bij een witte stip.

TABEL 2

ADERSIGNAAL		
1	hv0	hor. of vert. adres
2	hv1	hor. of vert. adres
3	hv2	hor. of vert. adres
4	hv3	hor. of vert. adres
5	hv4	hor. of vert. adres
6	hv5	hor. of vert. adres
7	hv6	hor. of vert. adres
9	hv7	hor. of vert. adres
8	$\overline{\Phi 2}$	clockpuls van computer.
10	A1	Stuurlijnen voor de
11	B1	soort actie.
12	C1	(Clear, schrijf enz.)
13	A2	Stuurlijnen voor de
14	B2	te schrijven kleur
15	Dout	Geeft bij een leesactie de kleur van de stip.
16	LB	Lijnsyncpuls voor synchronisatie van de computer.
24	GND	Massa.

TABEL 3

C1 B1 A1	functie
0 0 0	rust
0 0 1	klok horizontaal adres in
0 1 0	klok verticaal adres in
0 1 1	lees/schrijf geadresseerde punt
1 0 0	wis gehele beeld



TABEL 4

B2 A2	kleur
0 0	lees
0 1	inverteer
1 0	schrijf wit
1 1	schrijf zwart

tabel 2. Aansluitingen tussen computer en display.
tabel 3. De functie van de stuurlijnen A1...C1.
tabel 4. De functie van de stuurlijnen A2...B2.

afb. 25. Aansluitingen op een PIA (6800 en 65XX systemen).
afb. 26. Aansluitingen voor 8080 systemen.
afb. 27. Het schrijfcommando moet worden gegeven als 'LB' laag is. Het commando wordt echter pas uitgevoerd in de eerste 1,6 µs na de positieve flank van 'LB'.
afb. 28. Flowdiagram van de schrijf subroutine.

HET SIGNAAL LB
Door middel van LB (lijnblanking) kan de computer synchroniseren met het display. De computer moet zijn schrijf- (of lees)-commando geven tijdens het zichtbare deel van een lijn op het scherm (LB = laag). Tijdens de terugslag zet het display de aangeboden data op het scherm.

De I/0 van de computer

Bij de rest van dit verhaal gaan we er vanuit dat het adres voor het display op één uitgangspoort zit, en dat de stuurlijnen met LB en Dout op een tweede poort zitten. Van deze tweede poort moeten de bits 5, 6 en 7 als ingang **worden geprogrammeerd. In afb. 25 ziet u de aansluitingen voor 6800 en 65XX systemen. Port A ('ADRES') wordt gebruikt voor de adressen en Port B ('STUUR') wordt gebruikt voor de stuursignalen en de 2 inkomende lijnen. De stand van de richtingsregisters is onder de registers aangegeven.**

Wanneer een programma wordt gestart moeten we altijd beginnen met het setten van de richtingsregisters. (Dus in $ FF in PADD en $ 1F in PBDD). Bij 8080 georiënteerde systemen volgen we afb. 26. Hier krijgen de ingaande signalen een aparte ingangspoort.

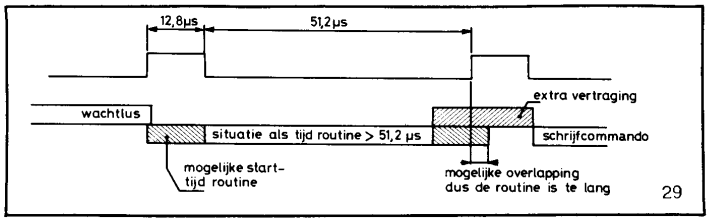
Het schrijven van een punt op het scherm

Wanneer we een punt op het scherm willen zetten komt er iets meer kijken. Allereerst moeten horizontaal en verticaal adres worden ingeklokt, de gewenste kleur moet worden bepaald en tijdens het zichtbare deel van een lijn moet het schrijfcommando worden gegeven. Het commando wordt echter pas in de eerstvolgende lijnblankingpuls uitgevoerd. In afb. 27 is een en ander nog eens getekend. Nieuwe adressen mogen dus pas 1,6 us ná de positieve flank van de lijnblanking worden ingeklokt. Afb. 28 geeft de flowchart van de schrijfroutine. We gaan er vanuit dat de kleur één keer van tevoren is bepaald. Dit kan eenvoudig worden gedaan door de juiste bitcombinatie van A2 en B2 op 'stuur' te zetten. Om vroegtijdige acties te voorkomen moeten we de stuurlijn A1 . . . C1 in de ruststand ('000') zetten of laten staan.

In hex krijgen we dus:

$00 voor lezen
$08 voor inverteren
$10 voor wit schrijven
$18 voor zwart schrijven

Dit komt overeen met de codes uit tabel 4, ervan uitgaande dat A2 op bit 3 is aangesloten, en B2 op bit 4. De eerste stap in de schrijfroutine is nu het inklokken van beide adressen. Het horizontale adres zullen we voortaan X-adres noemen, het verticale adres zullen we voortaan Y-adres noemen. Wanneer we een serie van punten willen schrijven (wat meestal het geval zal zijn) moeten we er vóór het inklokken van de adressen zeker van zijn dat een eventueel eerder gegeven commando is uitgevoerd. Daarom begint de routine met het wachten tot de lijnblanking hoog is. Op dit moment weten we zeker dat een eventueel eerder gegeven schrijfcommando is uitgevoerd, zodat we rustig nieuwe adressen kunnen doorgeven. De benodigde codes voor het inklokken van X en Y adres en de code voor het schrijfcommando zijn expres in numerieke volgorde gekozen. Hierdoor kunnen we steeds met een 'INC'-instructie naar de volgende code springen (voor 8080: 'INR'). Het flowdiagram (afb. 28) begint dus met



een wachtlus. Zodra 'lijnblanking' hoog is wordt het X adres op de uitgangen ('ADRES') gezet. Met een 'INC'- (of 'INR')-instructie komt op stuurlijnen C1 . . . A1 de code 001 te staan, zodat het adres op de uitgangen van de computer als X-adres wordt ingeklokt. Zolang deze code op de stuurlijnen staat zal de X-adresbuffer van het display precies de uitgangen van de computer volgen. Voordat het Y-adres op deze uitgangen gezet wordt moeten we nog een keer d.m.v. 'INC' de stuurlijnen ophogen. Nu wordt de Y-adresbuffer van het display actief, en zal aanvankelijk het nog niet verandere X-adres inklokken.

Maar, zoals al eerder gezegd blijft de Y-buffer de uitgangen van de computer volgen. We kunnen nu dus het Y-adres op de 'ADRES'-uitgangen zetten (afb. 25 of 26). Door nog een keer 'INC' (INR) toe te passen wordt ook de Y-adresbuffer gesloten, en wordt automatisch een schrijfcommando gegenereerd.

Als laatste actie moeten de stuurlijnen C1...A1 weer in de rusttoestand ('000') worden geplaatst. Het liefst moet hierbij de kleur gehandhaafd blijven, zodat bij het schrijven van de volgende stip de kleur vast klaar staat. De snelste methode om dit te bereiken is door een kopie van de schrijfkleur elders in het geheugen op te halen en op de 'STUUR'-uitgangen te zetten. Wanneer we van schrijfkleur willen veranderen moet dus de overeenkomstige code op de 'STUUR'-uitgangen én op de geheugenlocatie waar de kopie staat worden geschreven. De codes aan de 'STUUR'-uitgangen doorlopen nu achtereenvolgens de volgende waarden:

		B2	A2	C1	B1	A1	
1	rusttoestand	X	X	0	0	0	rust
2	na 'INC'	X	X	0	0	1	klok X
3	na 'INC'	X	X	0	1	0	klok Y
4	na 'INC'	X	X	0	1	1	actie
5	kopie schrijfkleur naar de uitgangen	X	X	0	0	0	rust

De lijnen B2 en A2 houden gedurende het hele proces de bitcombinatie voor de geprogrammeerde schrijfkleur.

Synchronisatie

Omdat het schrijfcommando alléén gegeven mag worden tijdens het zichtbare deel van een lijn (lijnblanking is laag) is dit hele proces aan een tijdlimiet gebonden. De routine start wanneer het LB signaal hoog is. Normaal beginnen we in de wachtlus en detecteren dan de positieve flank van het LB signaal.

Het is evenwel mogelijk dat we naar de wachtlus springen op het moment dat LB hoog is. In dat geval 'rollen' we meteen door de wachtlus heen. De eigenlijke routine kan dus beginnen van af de positieve flank tot net na de negatieve flank in het LB-signaal. In afb. 29 is dit gebied gearceerd aangegeven.

Zoals al gezegd moet het schrijfcommando worden gegeven wanneer LB laag is. Dit houdt in dat het hele proces niet te lang mag duren. Het LB-signaal blijft gedurende 51,2 µs laag, zodat de **hele** routine (inclusief 1 × door de wachtlus) niet langer dan 51,2 µs mag duren. Wanneer het gebruikte computersysteem te traag is, moeten we zorgen dat het schrijfcommando niet eerder komt dan na 76,8 us. (afb. 29). Na deze tijd is het zeker dat het LB-signaal voor de tweede maal laag is. Het inklokken van X- en Y-adres mag overigens wél gebeuren als LB hoog is.

Het programma

In dit deel geven we de routine welke nodig is om een stip op het scherm te zetten. Wanneer de routine uitgevoerd wordt als subroutine kan op eenvoudige wijze elke figuur op het scherm worden gezet. De onderstaande routines zijn nog niet geassembleerd, omdat de verschillende geheugenadressen door de gebruiker moeten worden bepaald.

6502 (1 MHz)

```
SCHRIJF  LDA   STUUR   4
         BPL   SCHRIJF  2
         LDA   XADR    3
         STA   ADRES   4
         INC   STUUR   6
         INC   STUUR   6
         LDA   YADR    3
         STA   ADRES   4
         LDA   KLEUR   3
         INC   STUUR   6
         STA   STUUR   4
         (RTS)
```

totaal benodigd: 45 µs

Bij het berekenen van de instructietijd zijn we er vanuit gegaan dat de PIA's niet op pagina 0 zitten. De locatie XADR, YADR en KLEUR werden uiteraard wél op pagina 0 gekozen. De totaal benodigde tijd blijft binnen de eis van 51,2 µs. De laatste instructie (RTS) is uiteraard alleen nodig als deze routine als subroutine wordt uitgevoerd. Het geheel zou nog 6 µs sneller worden als het X adres in indexregister X, en het Y adres in indexregister Y zou staan.
In de praktijk blijkt echter dat deze registers vaak al een andere taak hebben.

8080 (2 MHz)

```
SCHRIJF  IN    SYNC     5
         ANA   A        2
         JP    SCHRIJF  5
         MOV   A,E      2,5
         OUT   ADRES    5
         MOV   A,L      2,5
         INR   A        2,5
         OUT   STUUR    5
         INR   A        2,5
         OUT   STUUR    5
         MOV   A,D      2,5
         OUT   ADRES    5
                        44,5
```

Op dit moment is het X en Y adres ingeklokt. De tussenoptelling leert ons dat er niet meer voldoende tijd is om het schrijfcommando te geven. Het LB signaal kan nl. al na 51,2 µs weer omhoog gaan, en zal dit na 64 µs zeker hebben gedaan. Voor de 'schrijfpuls' zijn minstens 2 'OUT'-instructies nodig, zodat de totaaltijd minstens op 54,4 µs komt. Er zijn 2 mogelijkheden om de 'schrijfpuls' in de volgende laag periode van het LB-signaal te laten vallen. De eerste methode is het inbouwen van een extra vertraging. We weten nl. dat LB binnen 64 µs omhoog gaat, en dit 12,8 µs blijft. Met een extra vertraging kunnen we zorgen dat pas na 76,8 µs het schrijfcommando volgt. Methode 2 is: gewoon kijken naar LB en wachten totdat deze omhoog gaat: de routine gaat dan als volgt verder:

```
WACHT    IN    SYNC     5
         ANA   A        2
         JP    WACHT    5
         MOV   A,L      2,5
         ORI   03H      3,5
         OUT   STUUR    5
         MOV   A,L      2,5
         OUT   STUUR    5
         (RET)
```

We wachten weer op de positieve flank van LB. De tijd die daarna verstrijkt (vóórdat het OUT-commando het begin van de schrijflus verzorgt) is groter dan 12,8 µs, zodat we zeker weten dat LB laag is. Bij dit 8080 programma zijn we er vanuit gegaan dat X-adres in register E staat, en het Y-adres in register D. De schrijfkleur moet in register L staan.

6800 (1 MHz)

```
SCHRIJF  LDA A  STUUR    4
         BPL    SCHRIJF  4
         LDA A  XADR     3
         STA A  ADRES    5
         INC    STUUR    6
         INC    STUUR    6
         LDA A  YADR     3
         STA A  ADRES    5
         LDA A  KLEUR    3
         INC    STUUR    6
         STA A  STUUR    5
         (RTS)
```

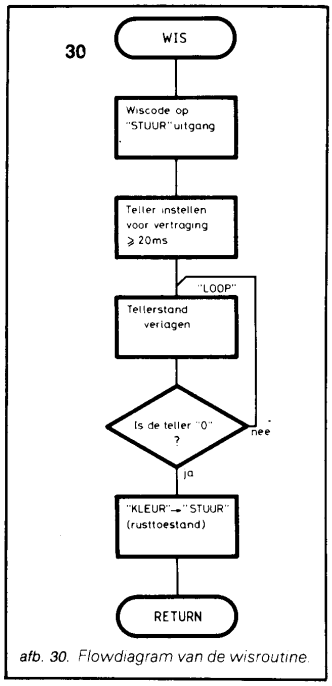
totaal benodigde tijd: 50 µs

Ook hier zijn de PIA's niet en locaties 'XADR', YADR' en 'KLEUR' wel op pagina 0 geplaatst. Uiteraard kan ook accumulator B worden gebruikt i.p.v. accumulator A.

Het schoonmaken van het scherm

De wiscode (100) verzorgt niet automatisch een schoon scherm. In de vorige delen schreven we al dat naar een evenwicht tussen software en hardware is gezocht. Wel, in dit geval zorgt de hardware uitsluitend dat de WE lijn van alle RAM's laag wordt. Wanneer we het scherm willen schoonmaken moeten alle geheugenplaatsen '0' of '1' worden, afhankelijk van de gewenste kleur. Om dit te bereiken moet de wiscode voor ca. 20 µs worden aangeboden. In deze tijd zijn alle adressen een keer doorlopen, en is het beeld wit of zwart. Het wissen gebeurt dus niet speciaal tijdens de terugslag, en het is dan ook niet nodig te synchroniseren met het LB-signaal. De kleurbepaling gebeurt normaal met de stuurlijnen A2 B2. Doordat we met de wiscode 8 WE lijnen gelijktijdig laag maken is het niet mogelijk om als 'kleur' inverteren of lezen in te vullen. We zijn nu immers niet in staat om één van de 8 bitjes te selecteren. Wel kunnen we natuurlijk het beeld wit of zwart maken. De hex code welke we op 'STUUR' moeten zetten wordt voor de verschillende kleuren:

$14 zwart scherm
$1C wit scherm
$04) niet gedefinieerd
$0C) ('spikkeltjes'-scherm)



afb. 30. Flowdiagram van de wisroutine.

Het flowdiagram voor de wisroutine vindt u in afb. 30. Nadat de wiscode 20 ms op de uitgang 'STUUR' heeft gestaan moet de oorspronkelijke kleur weer worden terug gezet en moeten de stuurlijnen A1 . . . C1 weer op 000 (rust) komen te staan. Zoals al eerder opgemerkt moet altijd een kopie van de kleurcode in RAM worden bewaard. Door aan het eind van de wisroutine deze kopie, die we hier 'KLEUR' dopen, op de 'STUUR'-uitgangen te zetten komen alle stuurlijnen weer in de goede positie. Het schrijven van de wisroutine zal m.b.t. het flowdiagram (afb. 29) geen moeilijkheden opleveren.