



LETTERS

OP HET GRAFISCH TV-DISPLAY

D. M. de Boer

In de vorige artikelenreeks over het zelfbouw grafisch TV-display werd uitgebreid aandacht besteed aan de hardware van dit display. Ook werden twee belangrijke routines besproken, nl. het aan of uitzetten van één van de 65536 aanwezige stippen en het wissen van het beeld. Maar... een display is geen display als er geen teksten op kunnen worden weergegeven. Daarom volgt in dit artikel wellicht het belangrijkste stukje software dat voor dit display werd geschreven: Het 'tekenen' van letters, met ongekende flexibiliteit.

Hardware of Software?

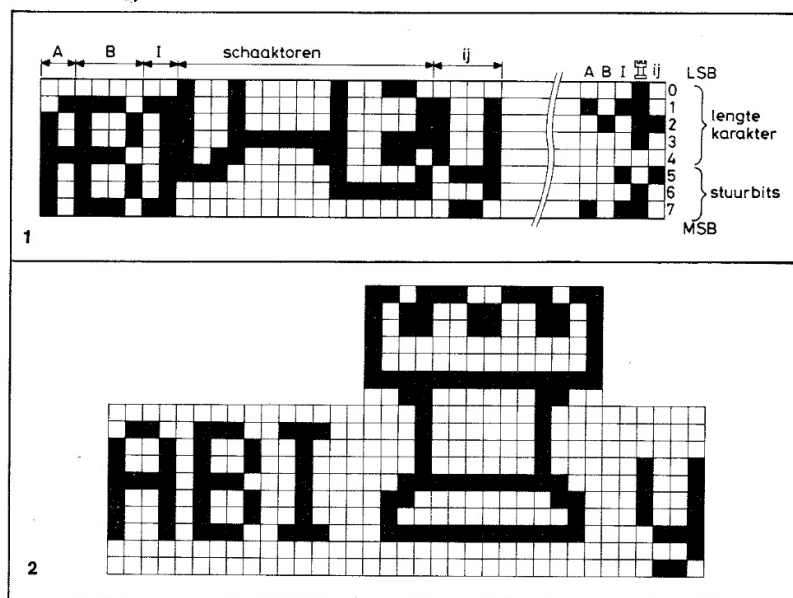
Letters op het grafisch display kunnen op verschillende manieren worden gerealiseerd. Allereerst de 'klassieke' methode, een karaktergenerator met een speciaal stuk RAM waarin de karakters in ASCII-vorm zijn opgeslagen. Het voordeel van deze methode is dat het display zeer snel is, en dat gemakkelijk letters van het scherm terug gelezen kunnen worden. Nadeel van de 'hardware methode' is, dat er componenten moeten worden aangeschaft en dat de gebruiker met handen en voeten gebonden is aan de eigenschappen van de schakeling.

De software oplossing geeft een ongekende flexibiliteit voor wat betreft lettervorm, letterhoogte, letterbreedte, letterafstand, letterplaats enz. Bovendien is de oplossing goedkoper (de hele 'schakeling' past in twee EPROM's 2708). Het nadeel van software, de traagheid, lijkt mee te vallen. Met de 1 MHz 6502 kunnen ongeveer 125 karakters per seconde worden getekend. De 6800 is iets trager, de 8080 haalt ongeveer 70 karakters per seconde. We gaan er hierbij vanuit dat er voor een karakter 48 punten moeten worden getekend (8 x 6 matrix). In dit geval wordt de letter wit getekend, en de achtergrond zwart, of omgekeerd. Wanneer we het tekenen van de achtergrond achterwege laten wordt het geheel nog iets sneller. Eén nadeel blijft, het teruglezen van een karakter van het scherm is niet eenvoudig. Dit probleem is te omzeilen door de ASCII-code van de letters op het scherm in het computergeheugen op te slaan. Vaak is het helemaal niet nodig om letters terug te lezen, de kopie is dan natuurlijk overbodig. De software oplossing biedt echter nog een aantal zeer aantrekkelijke mogelijkheden. Er is nl. automatisch een soort (zeer flexibele) programmeerbare

karaktergenerator ontstaan. De karakterset kan op elke plaats in het RAM (of EPROM) staan.

De benodigde karakters voor een schaakprogramma kunnen bij het schaakprogramma worden geschreven, bij kaartspelletjes kan een kaartsymbolengenerator worden gebruikt enz. Dus voor elke specifieke toepassing kunnen de bij de toepassing behorende symbolen gewoon in het computergeheugen worden gezet. Natuurlijk kunnen ook verschillende karaktersets gelijktijdig in het geheugen staan, terwijl het programma bepaalt welke karakterset moet worden gebruikt. Doordat alle karakters slechts éénmaal op het scherm worden getekend, en daarna onveranderd blijven staan, kunnen meerdere karaktersets ook dóór elkaar worden gebruikt. Grafische karakters zijn natuurlijk niet nodig omdat dit display volledige grafische mogelijkheden bezit.

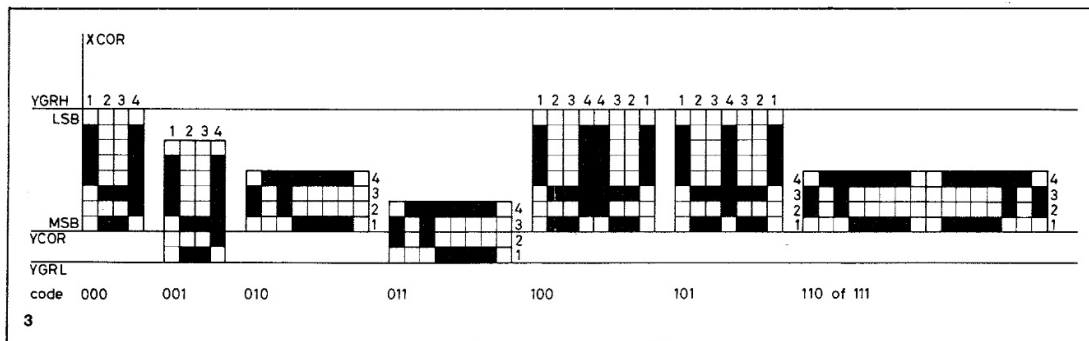
Al deze enorme mogelijkheden hebben ons doen kiezen voor de software karaktergeneratie. Natuurlijk blijft het mogelijk om naast deze software een 'normale' karaktergenerator toe te passen, en deze aan de display hardware toe te voegen. Op deze manier worden snelheid en flexibiliteit gekoppeld.



Software karaktergeneratie

Bij de karaktergeneratie hebben we een programma nodig dat karakters volgens voorgeschreven vorm op het scherm tekent. Daarbij wordt gebruik gemaakt van een tabel waarin de lettervormen liggen opgeslagen. Deze tabel kan worden beschouwd als een software karakterset. De manier waarop de letters liggen opgeslagen hangt natuurlijk af van het gebruikte programma. In het door ons ontwikkelde programma is gekozen voor een maximale vrijheid voor wat betreft karakterbreedte. Het puntjesraster waarin het karakter moet komen te staan is 8 puntjes hoog en kan 1 tot 31 puntjes breed zijn. Bij 'normale' letters zal de breedte variëren van 1 tot 5 puntjes. Wanneer een letter een verticale symmetrie-as heeft (bijvoorbeeld A) hoeft de letter maar half in de

karaktergenerator te staan, het programma zorgt d.m.v. spiegeling voor de andere helft. Bovendien bestaat de mogelijkheid een karakter gekanteld in de karakterset op te bergen. We denken hierbij vooral aan schaakstukken. Een gekanteld karakter is altijd 8 puntjes breed en kan 1 tot 31 puntjes hoog zijn. Wanneer we met symmetrische figuren (verticale symmetrie-as) werken wordt het figuur 16 puntjes breed, en nog steeds tussen de 1 en 31 puntjes hoog. Op deze manier kunnen schaakstukken zeer gedetailleerd worden opgeslagen. Alleen het paard is niet symmetrisch, en moet in twee delen worden gesplitst. Op dezelfde wijze kunnen raketjes, maanlanders, poppetjes enz. als karakter in de karaktergenerator worden gezet. Een en ander is verduidelijkt in afb. 1. Hier ziet u een stukje van de karakertabel. De bits die '0' zijn, zijn open gelaten, de bits die '1' zijn, zijn gekleurd. Achter elkaar zijn 5 verschillende situaties getekend. Normaal staan de karakters natuurlijk op alfabetische volgorde, deze volgorde is als voorbeeld gekozen. In afb. 2 ziet u hoe de verschillende karakters op het scherm verschijnen. Buiten de vorm van de letters moet het programma natuurlijk ook weten waar het karakter in de karakterset is te vinden, en hoeveel bytes het karakter in beslag neemt. Daarom moet aan de karakterset een tabel worden toegevoegd waaruit deze informatie is te halen. In deze tabel staat voor elk karakter aangegeven hoelang het karakter is, en of het karakter gespiegeld, gedraaid en/of geschoven moet worden. Op de werking van de tabel gaan we later in dit verhaal wat dieper in. Doordat alle karakters strak op elkaar staan wordt zeer efficiënt met de ruimte omgesprongen. In een normale karaktergenerator zijn alle letters minstens 5 hokjes breed i.v.m. de breedte van de letter 'M'. De vier bovenstaande letters zouden 20 bytes in beslag hebben genomen, nu zijn dat maar 16 bytes. De schaaktoren zou niet mogelijk zijn geweest.



Instelmogelijkheden

Zoals we hebben gezien kunnen karakters in de karaktergenerator gespiegeld, gedraaid en geschoven worden (zie afb. 1 en 2). Omdat deze instellingen per karakter verschillen moeten zij ook voor elk karakter zijn opgegeven. Er zijn echter ook instellingen welke voor een bepaalde situatie voor alle karakters geldig zijn. We denken hierbij aan vergrotingsfactoren in de hoogte en de breedte, plaats waar het karakter moet komen te staan enz. In het door RB ontwikkelde programma zijn de volgende instellingen mogelijk:

XCOR Deze geheugenplaats geeft het horizontale adres van het karakter (0-255).

YCOR Deze geheugenplaats geeft het verticale adres van het karakter (0-255).

BREED Deze geheugenplaats geeft een factor waarmee de karakterbreedte wordt vermenigvuldigd. Dus voor BREED=3 worden de letters 3 x zo breed (0-255).

HOOG Hetzelfde als breed, maar nu voor de hoogte.

KLEUR Elk bitje van deze geheugenplaats heeft een eigen betekenis.

bit 0: Wanneer dit bit '0' is, zullen de gegevens van een karakter zoals beginadres, lengte, kantelen/draaien en/of schuiven worden bepaald aan de hand van de tabel. Wanneer dit bit '1' is kunnen we deze gegevens zelf bepalen (speciale effecten!!).

bit 1: Als dit bit '0' is, wordt de XCOR verhoogd met de lengte van het karakter met daarbij opgeteld een vast aantal plaatsjes volgens geheugen 'SPATIE'. Op deze manier wordt een variabele letterafstand verkregen (weinig ruimte voor de 'i', veel ruimte voor de 'm'). Wanneer dit bit '1' is wordt de XCOR met een vast getal verhoogd (volgens LTRAF). Hierdoor komen alle letters op een vaste afstand van elkaar te staan.

bit 2: Dit bit bepaalt of de letters op een witte of zwarte achtergrond komen te staan. '1' = zwart, '0' = wit.

bit 3 en 4: Deze bits bepalen de 'kleur' van de letter volgens onderstaande tabel:

4 3

0 0 n.v.t. ('lezen')

0 1 inverteren

1 0 wit

1 1 zwart.

bit 5: Wanneer bit 5 '1' is, wordt aan het eind van een regel automatisch een CR en LF gegenereerd (we beginnen dus op een nieuwe regel).

bit 6: Door bit 6 '0' te maken wordt de achtergrond niet getekend. Hierdoor hoeven per letter minder puntjes te worden getekend.

bit 7: Door bit 7 '0' te maken wordt de letter zelf niet en de achtergrond wèl getekend. Voor speciale effecten.

Door bit 6 en 7 beiden 0 te maken, gebeurt er op het scherm niets. Wel wordt door het programma beginadres en lengte van het karakter bepaald.

SPATIE Op deze geheugenplaats kan de afstand tussen twee letters worden aangegeven. Geldt alleen als bit 1 van 'KLEUR' nul is (variabele letter afstand).

LTRAF Op deze plaats kan een vaste letterafstand worden aangegeven (normaal = 6). Alle letters staan op een vaste afstand van elkaar. Voor LTRAF = 0 blijft XCOR ongewijzigd, en komen alle letters op dezelfde plaats op het scherm. Geldt alleen als bit van 'KLEUR' één is.

ASCII Met deze geheugenplaats kunnen we aangeven welk karakter we uit de karakterset willen halen. Wanneer bit 0 van 'KLEUR' logisch '1' is, gaan we uit van bekende gegevens ten aanzien van lengte, beginadres, kantelen/spiegelen/schuiven. In dit geval wordt niet naar 'ASCII' gekeken.

TBLHI en **TBLLO** Deze twee geheugenplaatsen geven het beginadres van de tabel. Omdat alle ASCII karakters onder de 20 (hex) niet afgedrukt hoeven te worden moeten

we het beginadres met \$20 verminderen. Voor bijvoorbeeld de ASCII code \$22 wordt nu automatisch het 3e getal uit de tabel gehaald.

KARHI en **KARLO** Door deze 2 plaatsen wordt het 16 bits beginadres van de karakterset aangegeven.

KNTLVO bevat de x coördinaat van de beginkantlijn. Wanneer automatisch een CR en LF wordt gegenereerd (bit 5 van 'KLEUR' = 1) zal de XCOR op de waarde van KNTLVO worden gezet.

KNTLAC bevat de x coördinaat van de eindkantlijn. Zodra XCOR voorbij deze kantlijn komt zal er òf een LF en CR worden gegenereerd, òf XCOR wordt niet meer opgehoogd. Door deze kantlijn op FF te zetten schakelen we de kantlijn uit.

Bovenstaande instellingen zullen in het begin wat onoverzichtelijk lijken. Voor maximale flexibiliteit is het aantal instelmogelijkheden zeker niet overdreven. Bij normaal gebruik hoeven al deze instellingen, op ASCII na, slechts éénmaal te worden gedaan. De coördinaten die bepalen op welke plaats de letter op het scherm verschijnt worden na het tekenen van een letter steeds automatisch goed gezet voor de volgende letter. Dit automatisme is overigens uitschakelbaar. Voor bijzondere toepassingen kunnen alle instellingen naar hartelust worden gewijzigd.

Optionele instellingen

De geheugenplaatsen 'LENGTE', 'MODE', 'SUMLO' en 'SUMHI' bevatten gegevens die karakteristiek zijn voor een bepaalde letter. Bij normaal gebruik (BIT0 van 'KLEUR' = 0) worden deze gegevens door het programma uit de tabel gehaald. Bits 0...4 uit de tabel geven aan hoeveel bytes een bepaald karakter inneemt. dit gedeelte wordt in 'LENGTE' gezet. Bits 5...7 geven aan of een karakter gedraaid, gespiegeld en/of geschoven is. Deze stuurbits worden in 'MODE' gezet.

Door de lengte van alle voorgaande karakters op te tellen bij het beginadres van de karakterset bepaalt het programma op welk adres het te tekenen karakter begint. Dit adres wordt op SUMLO en SUMHI gezet.

Wanneer bit 0 van 'KLEUR' = 1 moeten de bovenstaande gegevens worden ingevoerd. Dit kan bijzonder handig zijn als er voor een bepaalde toepassing ergens in het geheugen één enkel karakter staat. De lengte van het karakter was voor de karakterset beperkt tot max. 31 bytes. Doordat de stuurbits nu apart in 'MODE' staan, kunnen we de volle 8 bits gebruiken voor de opgave van de lengte. Dus voor bijzondere karakters welke niet in de karakterset staan, maar ergens 'los' in het geheugen, is het maximale karakterformaat 8 hoog x 255 breed, of 255 hoog x 8 breed. Bij symmetrie wordt dit 255 hoog x 16 lang. Natuurlijk kunnen onafhankelijk hiervan nog verbredings- en verhogingsfactoren ingevoerd worden.

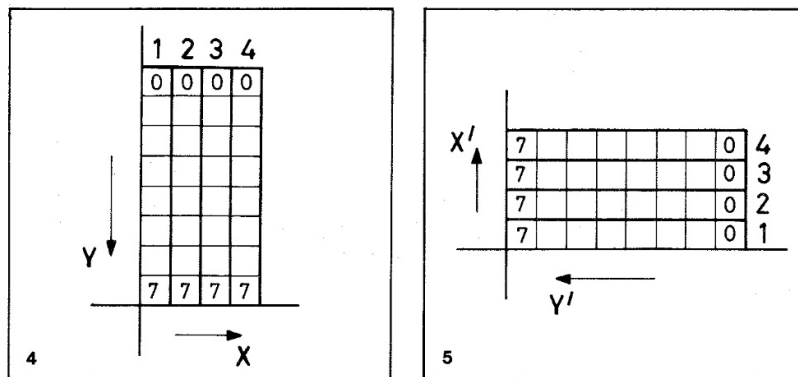
Kantelen/schuiven en spiegelen

In de tabel kunnen we voor elk karakter aangeven of het gekanteld, geschoven en/of gespiegeld moet worden. Wanneer we geen gebruik maken van de karakterset (zie 'Optionele instellingen') kunnen we deze gegevens kwijt op geheugenplaats 'MODE'. De

codes zijn als volgt:

7	6	5	
1	X	0	spiegelen middenpoot dubbel
1	0	1	spiegelen middenpoot niet dubbel
X	1	X	draaien
0	X	1	schuiven

Bij symmetrische karakters hebben we dus niet de mogelijkheid tot schuiven. Bit 5 geeft nu aan of het laatste byte uit de karakterset bij spiegelen moet worden herhaald. Het effect heeft u al gezien in afb. 1 en 2 bij de letters A en I. Om het effect van de verschillende stuurcodes nog eens duidelijk te laten zien, hebben we alle mogelijke stuurbitcombinaties op de letter y los gelaten. Het resultaat ziet u in afb. 3. De normale stuurcode voor deze letter is 001 (geschoven).



Hulpvariabelen

Voor de realisatie van het programma is het nodig verschillende hulpvariabelen in te voeren. In afb. 3 ziet u al twee van deze hulpvariabelen, nl. YGRH en YGRL. Wanneer de achtergrond (ook) moet worden getekend zal het programma alle vakjes binnen deze grenzen (waar geen letters staan) met de achtergrondkleur opvullen. Op deze manier verdwijnen alle 'resten' van oude teksten. YGRH ligt 8 x de verhogingsfactor boven YCOR, en YGRL ligt 2 x de verhogingsfactor onder YCOR. De verhogingsfactor is het getal dat in 'HOOG' staat. Bij normaal gebruik zal dit 1 zijn. Bij stuurcode '000' (normale situatie afb. 3) wordt de letter zo op het scherm gezet dat een letter met het puntje links onder op de coördinaten (XCOR), (YCOR) komt te staan. Wanneer we moeten schuiven (stuurcode 001 of 011) wordt aan het begin van het programma YCOR gelijk gemaakt aan YGRL, waardoor de hele letter 2 punten zakt. Aan het eind van het programma wordt YCOR weer op z'n oorspronkelijke waarde terug gezet

Tijdens het tekenen van de letter geven steeds de twee hulpvariabelen XCOR2 en YCOR2 aan waar het eerstvolgende puntje moet worden getekend. In afb. 4 ziet u dit in tekening gebracht. Als eerste wordt bitje 7 van byte 1 getekend. Voor elk volgend bitje wordt YCOR2 met één verlaagd (lagere Y coördinaat is hoger in het beeld). Wanneer dit proces 8 x is herhaald, wordt YCOR2 weer op de beginwaarde gezet en wordt XCOR2

met één verhoogd. Op dezelfde wijze worden de volgende bytes getekend. Het aantal bytes dat wordt getekend komt overeen met het getal dat in 'lengte' staat. Bij een gekanteld karakter willen we graag hetzelfde programma gebruiken. Dat kan door het coördinatenstelsel aan te nemen volgens afb. 5. De richting van de coördinaten is hier hetzelfde als in afb. 4 (Y coördinaat oplopend van bit 0 naar 7, X coördinaat oplopend van byte 1 naar byte 4). We zien dus dat niet alleen X en Y coördinaat zijn verwisseld, maar dat bovendien de richtingen tegengesteld zijn. De richtingsverandering is te verkrijgen door de coördinaten te inverteren. In de normale situatie maakten we $XCOR2$ gelijk aan $XCOR$ en $YCOR2$ gelijk aan $YCOR$, nu moeten we $XCOR2$ gelijk maken aan $/YCOR$ en $YCOR2$ aan $/XCOR$. Bij het doorgeven van adressen aan het display moeten we coördinaten natuurlijk weer terug draaien. Wanneer nu een volgend bitje moet worden getekend zullen we net als vroeger $YCOR2$ met één verlagen. Doordat $YCOR2$ nu in wezen een geïnverteerde X coördinaat is, zal het uiteindelijke effect op het scherm een verhoogde X coördinaat zijn. Hierdoor zal het nieuwe bitje rechts van het vorige bitje worden getekend.

De symmetrie is zeer eenvoudig te realiseren. Wanneer symmetrie gewenst is wordt er bij het bereiken van het laatste te tekenen byte een 'wissel' omgezet waardoor niet het eerstvolgende byte uit de karakterset wordt gehaald, maar juist het vorige byte. Op die manier wordt (bij een karakterlengte van 4) na byte 4 weer de bytes 3, 2 en 1 getekend, waardoor het spiegeleffect ontstaat. Bij gekantelde symmetrie (code 11X) ligt de situatie iets moeilijker. We stellen de hulpcoördinaten nu zo in alsof hetzelfde karakter nog een keer strak naast het reeds getekende karakter moet komen te staan. We schuiven nu het te tekenen byte echter niet naar links, maar naar rechts. Hierdoor zal niet bit 7 maar bit 0 als eerste worden getekend. Op deze manier komt ook in dit geval het spiegelbeeld tot stand.

Bijschriften deel 1:

afb. 1 Zó staan de karakters in de karaktergenerator. De laatste 5 bytes bevatten karakteristieke gegevens voor de karakters uit de karakterset.

afb. 2 Dit is het resultaat op het scherm.

afb. 3 Het effect van de 3 bits stuurcode op de letter y.

afb. 4 Dit is de normale richting van X en Y coördinaten.

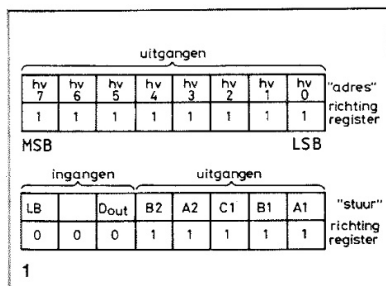
afb. 5 Bij gedraaide karakters nemen we een gedraaid coördinatenstelsel aan, waardoor het programma vrijwel op dezelfde manier verloopt.

In het vorige deel zijn we uitgebreid ingegaan op de manier waarop de karakters in het geheugen staan, de instelmogelijkheden van het programma, en de wijze waarop het programma karakters spiegelt, draait en/of schuift. In dit deel zullen we het stroomdiagram en het programma voor de 6502 behandelen.

Het initialiseren (afb. 2)

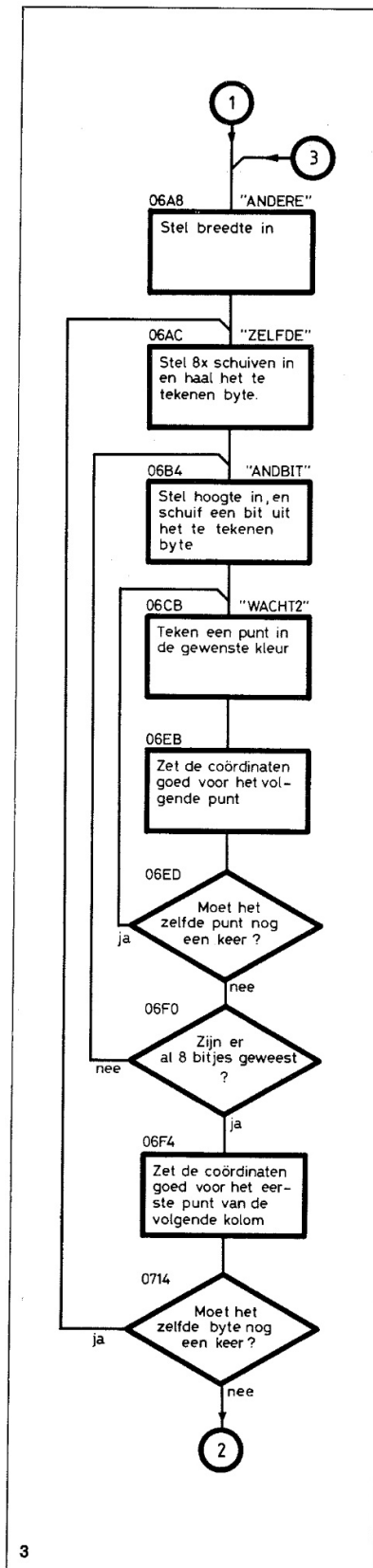
Bij het initialiseren krijgen de diverse hulpvariabelen hun beginwaarde. Omdat het programma is uitgevoerd als subroutine gaan we ervan uit dat de richtingsregisters van de PIA's reeds goed staan. In afb. 1 ziet u nog eens welke aansluitingen ingang zijn en welke als uitgang dienst doen. Het initialiseren begint met het destilleren van de gewenste voor- en achtergrondkleur uit geheugenplaats 'KLEUR'. De voorgrondkleur komt in 'VOOR' te staan, de achtergrond in 'ACHTER'. Als bit 0 van 'KLEUR' de waarde '0' heeft zal het programma de lengte, de stuurbits en het beginadres van het karakter bepalen aan de hand van de gegevens uit de tabel. De lengte en de stuurbits worden rechtstreeks uit de tabel gehaald en op resp. 'LENGTE' en 'MODE' gezet. Het beginadres van het karakter wordt bepaald door de lengten van alle voorgaande karakters op te tellen bij het beginadres van de karakterset. We zijn inmiddels aangeland op adres 0646 (afb. 2 lijst 1), en op dit moment kennen we alle gegevens van het karakter.

We beginnen nu met het berekenen en instellen van de hulpvariabelen. Allereerst YGRL en YGRH. Met deze coördinaten is een balk gedefinieerd van 10 puntjes hoog, waartussen alle (normale) karakters komen te liggen. Deze coördinaten worden later in het programma gebruikt om de achtergrond boven, onder en tussen de letters aan te vullen.



Op adres 065E wordt bepaald of het karakter al dan niet geschoven is. Wanneer het karakter geschoven is wordt YCOR overeenkomstig ingesteld. Later in het programma wordt hij ook weer terug gezet, zodat YCOR uiteindelijk ongewijzigd uit de strijd komt. Om snelheidsredenen wordt straks bij het tekenen van één byte steeds alléén de Y-coördinaat in het display veranderd bij normale karakters, en steeds alléén de

X-coördinaat bij gekantelde karakters. Daarom worden nu reeds zowel het X adres als het Y adres in het display gezet (adres 0666). De laatste stap is het instellen van de



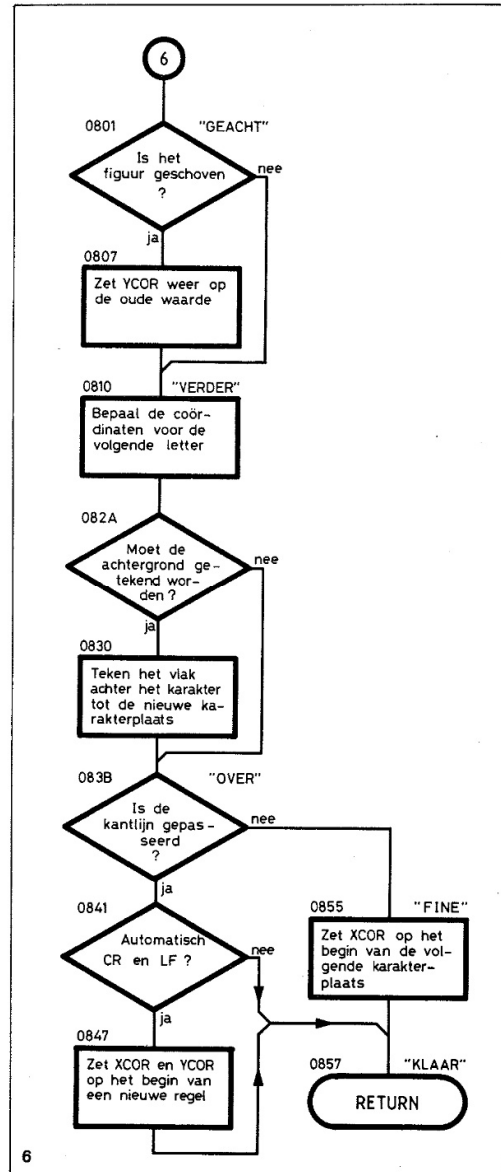
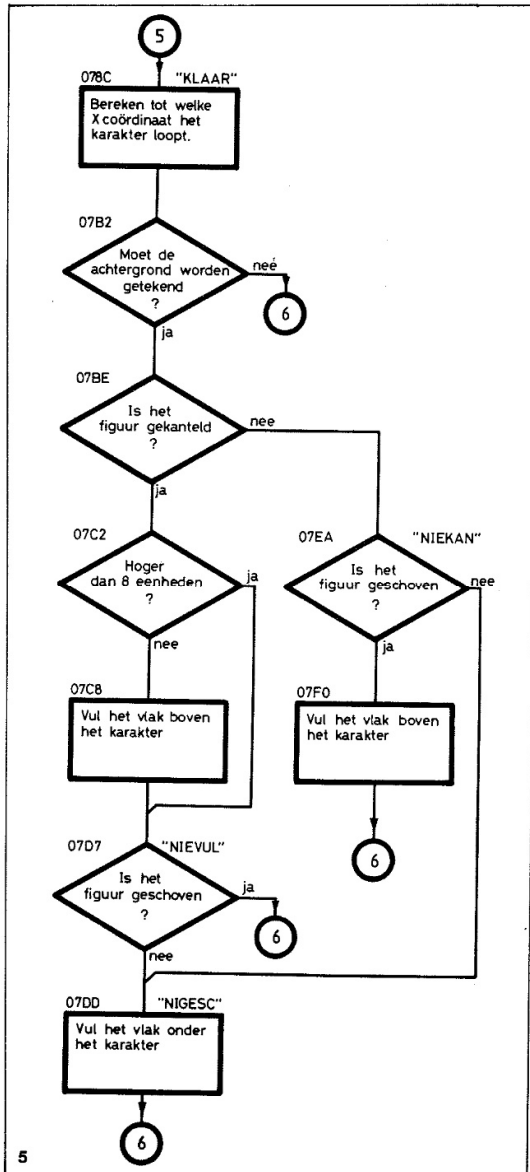
hulpcoördinaten YCOR2, YCOR3, XCOR2, BREED2 en HOOG2. Zoals in het vorige deel al ter sprake kwam worden deze hulpcoördinaten bij gekantelde karakters 'gedraaid' overgenomen. De coördinaten XCOR2 en YCOR2 zijn de 'lopende' coördinaten. YCOR3 blijft tijdens het tekenen ongewijzigd, en wordt gebruikt om na het tekenen van 8 bitjes steeds YCOR2 weer op de oude waarde te kunnen zetten.

Het tekenen van een karakter

In afb. 3 en 4 ziet u het stroomdiagram van de kern van het programma. Afb. 3 geeft de procedure voor het tekenen van één byte uit de karakterset, afb. 4 zorgt dat steeds een nieuw karakterbyte wordt aangevoerd. Omdat elk karakter zowel in de breedte als in de hoogte kan worden uitgerekt zijn er wat lussen en tellers nodig. Elk byte wordt vertikaal neergezet (uitgaande van de normale situatie), met MSB onder. Wanneer we een karakter 3x zo breed willen hebben, moet elk byte 3x worden getekend. Daarom moeten we aan het begin van dit programma een teller instellen die bijhoudt hoeveel maal hetzelfde byte is getekend. Deze teller heeft de naam 'BREED3' gekregen. In het volgende blokje (06AC) wordt geheugenplaats 'TEL' op 08 gezet. Met deze teller houden we bij of er al 8 bitjes van het byte zijn getekend. Indexregister Y wordt gebruikt om het te tekenen byte aan te wijzen en staat nu op 0. Op adres 06B0 wordt hierdoor het eerst te tekenen byte opgehaald en op 'TEMP' gezet. De laatste in te stellen teller is de teller voor de verhogingsfactor. Hiervoor gebruiken we indexregister X. Na alle instellingen gaan we het eerste bitje uit het te tekenen byte schuiven. Normaal schuiven we naar links (MSB eerst). Alleen als we met het spiegelbeeld van een gekanteld figuur bezig zijn, moeten we naar rechts schuiven. Bit 6 van 'VLAG' geeft deze situatie aan. In ieder geval staat na het schuiven het te tekenen bitje in de carry.

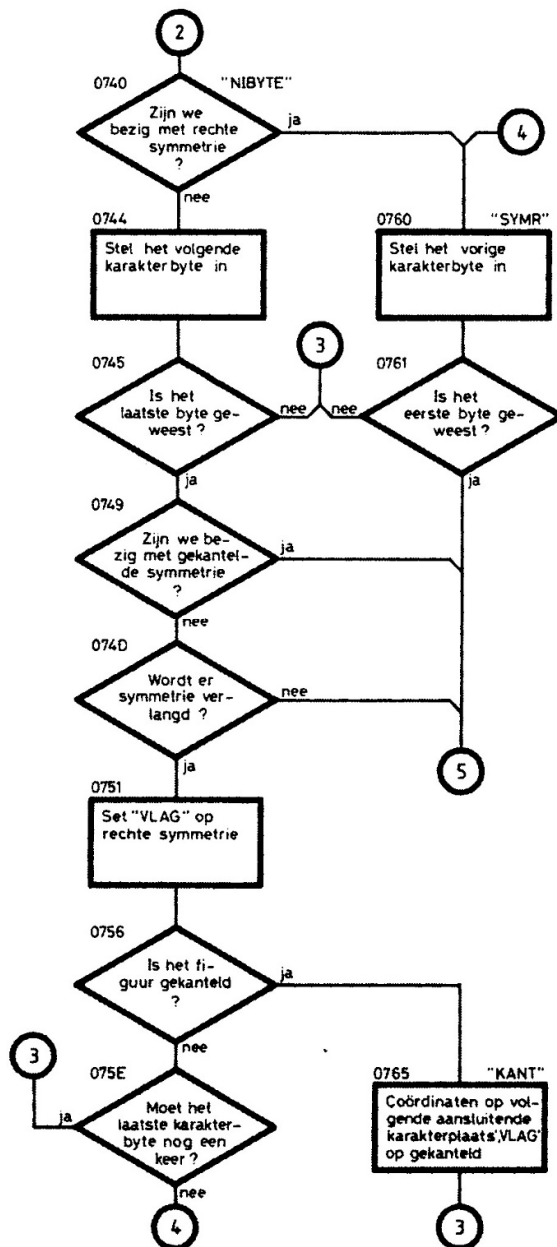
Teken een punt in de gewenste kleur

We komen nu in het hart van alle lussen (afb. 3). Omdat dit programmadeel veel herhaald wordt is dit stukje programma ontworpen op snelheid en niet zozeer op een klein aantal programmastappen. Bij het tekenen van één stip kunnen zich de volgende situaties voordoen: De stip moet worden getekend in de voorgrondkleur (carry=1). De stip moet worden getekend in de achtergrondkleur (carry=0), de stip moet niet getekend worden (bit 6 of 7 van 'KLEUR' = 0 én carry = 0 resp. 1). In de eerste twee gevallen wordt de kleur ingesteld, waarna naar adres 06CB wordt gesprongen. Hier wordt alléén het nieuwe Y-adres ingeklokt, omdat het X-adres gedurende het tekenen van een byte ongewijzigd blijft. (Voor een gekanteld karakter wordt uiteraard alléén het X-adres ingeklokt) Vervolgens wordt de schrijfpuls gegeven en de stip is getekend. Wanneer de stip niet getekend moest worden, omdat bit 6 of 7 (6 voor de achtergrond en 7 voor de voorgrond) '0' was, wordt YCOR2 zó ingesteld dat hij de plaats aanwijst van het eerstvolgend te tekenen bitje. In alle gevallen geldt dat de verhogingsfactor (staat op dit moment in indexregister X) aangeeft hoeveel maal een stip herhaald moet worden, vóórdat het volgende bit uit het tekenen byte geschoven wordt. Dit wordt aangegeven door de binnenste lus (afb. 3). Steeds als deze lus wordt doorlopen, wordt register X met één verminderd. Pas als dit register 0 is, wordt op adres 06F0 gekeken of er nog een bit van dit byte getekend moet worden. Zolang 'TEL' (welke we hadden ingesteld op 8) nog geen '0' is, wordt het te tekenen byte nog een keer geschoven.



Natuurlijk wordt ook de verhogingsfactor opnieuw ingesteld, omdat ook het nieuwe bitje misschien een aantal keren boven elkaar zal moeten worden getekend. Wanneer het byte is 'leeg geschoven' moeten we de coördinaten weer goed zetten voor de volgende kolom. Dit gebeurt vanaf adres 06F4. YCOR2 krijgt hier weer zijn oorspronkelijke waarde, terwijl XCOR2 met één wordt verhoogd. Vervolgens worden X- en Y-adres in het display gezet, zonder een schrijfpuls te genereren. Bij karakters in de normale positie behoeft alleen het X-adres ingeklokt te worden (XCOR2 X-adres), omdat het Y-adres voor elk bit opnieuw wordt ingeklokt. Bij een gedraaid karakter is dit net andersom, hier moet XCOR2 in het Y-buffer geheugen worden gezet, terwijl het X-adres voor elk bit opnieuw wordt ingeklokt. We belanden nu bij de laatste lus in afb. 3. Hetzelfde byte moet net zoveel keer getekend worden als de verbredingsfactor aangeeft. Op adres 0714 wordt gekeken of de teller 'BREED3' al bij 0 is. Zolang dit niet het geval is wordt de

hele voorgaande procedure herhaald, zij het één positie verder op het beeld. Zodra er een nieuw byte nodig is belanden we in afb. 4.



4

nieuwe byte in de juiste afmetingen getekend wordt. Zodra het laatste byte getekend is komen we via 0749 op adres 074D, waar gekeken wordt of het karakter gespiegeld moet worden. Wanneer dit niet het geval is rollen we door naar afb. 5. Als symmetrie wel wordt gevraagd zetten we voorlopig bit 7 van 'VLAG' ten teken van rechte symmetrie. Op 0756 wordt nu gekeken of het figuur ook werkelijk rechtop staat. We nemen voorlopig aan dat dit zo is. De laatste voorwaardelijke sprong vinden we op 075E. Hier wordt gekeken of de middenpoot dubbel moet worden genomen zoals bijvoorbeeld bij de A of

Het bepalen van het volgende byte

In afb. 4 ziet u het stukje flowdiagram dat niet alleen bepaalt wélk byte nu getekend moet worden, maar bovendien alle handelingen verricht om een karakter te spiegelen (symmetrie). We maken onderscheid tussen 'rechte' symmetrie (zoals bij de A en de I, zie het vorige deel) en 'gekantelde' symmetrie (zoals bij de schaaktoren). De rechte symmetrie wordt aangegeven door bit 7 van 'VLAG', de gekantelde symmetrie door bit 6. We komen in afb. 4 op adres 0740 binnen. Zolang nog niet alle bytes uit de karakterset getekend zijn, wordt op adres 0744 het volgende byte uit de karakterset ingesteld. Hierna springen we terug naar afb. 3, waar het

enkel zoals bijvoorbeeld bij de I (zie het vorige deel). Bij een dubbele middenpoot springen we weer terug naar afb. 3 waar het hele proces op exact dezelfde wijze wordt herhaald. Het enige verschil met de eerste keer is dat nu bit 7 van 'VLAG' log 1 is. Hierdoor is op adres 0740 (bovenaan in afb. 4) de 'wissel' omgezet, en wordt steeds het vorige karakterbyte ingesteld. Dit proces gaat door tot het 1e byte van het karakter uit de karakterset is getekend. Op dit moment is het gespiegelde gedeelte ook klaar en gaan we naar afb. 5. Bij het laatste geval, de gekantelde symmetrie gaan we bij 0756 (afb. 4) naar 0765. Hier worden de hulpcoördinaten ingesteld op de eerstvolgende karakterplaats, zij het zonder tussenruimte. Geheugenplaats 'VLAG' wordt nu ingesteld op gekantelde symmetrie, en register Y wordt 0 gemaakt, zodat hij weer het eerste karakterbyte aanwijst. Doordat 'VLAG' ingesteld is op gekantelde symmetrie wordt de wissel op adres 0740 nu niet bediend (het programma neemt dus steeds gewoon het volgende byte). Wel wordt steeds het karakterbyte naar rechts geschoven in plaats van naar links, waardoor bit 0 het eerst getekend wordt. Als op deze manier het spiegelbeeld tot stand is gekomen gaan we via 0745 en 0749 naar afb. 5.

Het opvullen van de ruimte tussen de letters

Het belangrijkste werk is gedaan, de letter is getekend. We zijn er echter nog niet, door mogelijkheden van kantelen en schuiven hebben niet alle letters dezelfde hoogte. Wanneer we de achtergrond willen tekenen moeten alle beeldstippen tussen YGRL en YGRH, waar geen letter zit, de achtergrondkleur krijgen. Alleen dan worden alle resten van oude tekst en/of tekeningen overschreven (zie afb. 3 vorig deel). Het opvullen van de achtergrond gebeurt volgens afb. 5.

Alléén gekantelde karakters kunnen hoger dan 8 punten zijn. In dit geval kan de ruimte boven de letter niet worden opgevuld (07C2).

In afb. 6 ziet u het laatste stuk van het flowdiagram. Hier worden de coördinaten van de volgende letterplaats bepaald. Pas als dit bekend is wordt de ruimte achter de letter opgevuld tot de volgende letterplaats.

Bijschriften deel 2:

afb. 1 De aansluitingen op de PIA.

afb. 2 Het eerste stukje van het stroomdiagram. Hier krijgen de diverse geheugens hun beginwaarde.

afb. 3 Dit stuk stroomdiagram tekent één byte in de gewenste hoogte- en breedte.

afb. 4 Het gedeelte dat zorg draagt voor het eventueel spiegelen van het karakter.

afb. 5 Met dit deel van het stroomdiagram wordt de ruimte onder en boven de letters in de achtergrondkleur getekend.

afb. 6 Het laatste deel. Hier wordt de volgende letterpositie berekend.

lijst 1 Het programma. De vet gedrukte adressen zijn aangepast op de I/O van de KIM, en moeten voor andere systemen worden aangepast. Wanneer een ander startadres gewenst wordt moeten de instructies op 07D4, 07E5, 07FE en 0838 aangepast worden.

0600	D8		START	CLD		
0601	A5	A5		LDA	KLEUR	
0603	29	18		AND	#\$18	
0605	85	BA		STA	VOOR	
0607	A5	A5		LDA	KLEUR	
0609	29	04		AND	# \$04	Initialiseren
060B	0A			ASL		
060C	09	10		ORA	# \$10	
060E	85	BB		STA	ACHTER	
0610	A2	00		LDX	# \$00	
0612	86	B2		STX	VLAG	
0614	A5	A5		LDA	KLEUR	
0616	29	01		AND	# \$01	Gegevens bekend?
0618	D0	2C		BNE	BEKEND	Zo ja, naar 'BEKEND'.
061A	A5	D2		LDA	KARLO	
061C	85	C0		STA	SUMLO	Kopieer het beginadres.
061E	A5	D3		LDA	KARHI	
0620	85	C1		STA	SUMHI	
0622	A4	A0		LDY	ASCII	
0624	B1	D4		LDA	(TBLL0), Y	
0626	29	E0		AND	# \$EO	Stel 'MODE' in.
0628	85	C2		STA	MODE	
062A	B1	D4		LDA	(TBLL0), Y	
062C	29	1F		AND	# \$1F	Stel 'LENGTE' in.
062E	85	C3		STA	LENGTE	
0630	88		TELOP	DEY		
0631	C0	1F		CPY	# \$1F	Eén karakter terug.
0633	F0	11		BEQ	BEKEND	Indien laatste karakter naar 'BEKEND'.
0635	B1	D4		LDA	(TBLL0), Y	
0637	29	1F		AND	# \$1F	Karakterlengte uit de tabel.
0639	18			CLC		
063A	65	C0		ADC	SUMLO	
063C	85	C0		STA	SUMLO	Tel alle lengten bij elkaar op om het startadres van het karakter te vinden.
063E	A9	00		LDA	# \$00	
0640	65	C1		ADC	SUMHI	
0642	85	C1		STA	SUMHI	
0644	90	FA		BCC	TELOP	Onvoorwaardelijk terug.
0646	A0	00	BEKEND	LDY	# \$00	
0648	A5	A2		LDA	YC0R	
064A	85	BC		STA	YGR L	
064C	A6	A4		LDX	HOO G	
064E	38		LOOP	SEC		
064F	E9	08		SBC	# \$08	
0651	E6	BC		INC	YGR L	Bepaald YGR L en YGR HI.
0653	E6	BC		INC	YGR L	
0655	CA		DEX			
0656	D0	F6		BNE	LOOP	
0658	85	BD		STA	YGR H	
065A	A9	20		LDA	# \$20	
065C	24	C2		BIT	MODE	
065E	F0	06		BEQ	VERDER	Indien geschoven
0660	30	04		BMI	VERDER	YC0R goed zetten.
0662	A5	BC		LDA	YGR L	
0664	85	A2		STA	YC0R	
0666	AD	02 17	WACHT1	STUUR	WACHT1	Wacht op syncpuls van de TV.
0669	10	FB		BPL	WACHT1	
066B	A5	A1		LDA	XC0R	
066D	8D	00 17		STA	ADRES	
0670	EE	02 17		INC	STUUR	X en Y-adres in het display zetten.
0673	EE	02 17		INC	STUUR	
0676	A5	A2		LDA	YC0R	
0678	8D	00 17		STA	ADRES	
067B	8E	02 17		STX	STUUR	
067E	A5	A1		LDA	XC0R	
0680	85	B8		STA	XC0R2	
0682	A5	A2		LDA	YC0R	
0684	85	B6		STA	YC0R2	Hulpcoördinaten instellen voor rechtstaand figuur
0686	85	B7		STA	YC0R3	
0688	A5	A3		LDA	BREED	
068A	85	B4		STA	BREED2	
068C	A5	A4		LDA	HOO G2	
068E	85	B3		STA	HOO G2	
0690	50	16		BVC	ANDERE	Spring indien niet gedraaid
0692	A5	A1		LDA	XC0R	
0694	49	FF		EOR	# \$FF	
0696	85	B6		STA	YC0R2	
0698	85	B7		STA	YC0R3	
069A	A5	A2		LDA	YC0R	Hulpcoördinaten instellen voor gekanteld figuur
069C	49	FF		EOR	# \$FF	
069E	85	B8		STA	XC0R2	
06A0	A5	A3	</			

06AA 85 B5			STA	BREED3	Opnieuw in.
06AC A9 B8	ZELFDE		LDA	#S08	Stel 8x
06AE 85 B9			STA	TEL	schuiven in.
06B0 B1 C0			LDA	(SUMLO),	Zet het te tekenen
06B2 85 B1			STA	TEMP	byte in 'TEMP'.
06B4 A6 B3	ANDBIT		LDA	HOOG2	Stel hoogte in.
06B6 24 B4			BIT	VLAG	
06B8 70 B4			BVS	RECHTS	Schuif een bit
06BA 06 B1			ASL	TEMP	uit 'TEMP'
06BC 50 B2			BVC	NORM	Normaal naar links
06BE 46 B1	RECHTS		LSR	TEMP	
06C0 24 A5	NORM		BIT	KLEUR	
06C2 90 5A			BCC	ACHTGR	Voor achtergrondkleur
06C4 10 61			BPL	NIET	naar 'ACHTGR'.
06C6 A5 BA			LDA	VOOR	Voorgr. tekenen?
06C8 8D 02 17			STA	STUUR	Stel voorgrond in
06CB AD 02 17	WACHT2		LDA	STUUR	Wacht op synclups
06CE 10 FB			BPL	WACHT2	van de TV.
06D0 24 C2			BIT	MODE	'Normaal naar 'YKLOK'
06D2 50 5A			BVC	YKLOK	om Y in te klokken.
06D4 A9 FF			LDA	#\$FF	
06D6 45 B6			EOR	YCOR2	
06D8 8D 00 17			STA	ADRES	Klok het X-adres in.
06DB EE 02 17			INC	STUUR	
06DE A9 02			LDA	#S02	
06E0 0D 02 17			ORA	STUUR	
06E3 8D 02 17			STA	STUUR	Schrijfpuls
06E6 29 18	RUST		AND	#S8	Terug naar de
06E8 8D 02 17			STA	STUUR	rusttoestand
06EB C6 B6			DEC	YCOR2	Als hetzelfde punt
06ED CA			DEX		nog een keer moet,
06EE D0 DB			BNE	WACHT2	naar 'WACHT2'.
06F0 C6 B9	NIBIT		DEC	TEL	Zijn we al bij het
06F2 D0 C0			BNE	ANDBIT	laatste bitje?
06F4 A5 B7			LDA	YCOR3	
06F6 85 B6			STA	YCOR2	Zet de coördinaten goed
06F8 E6 B8			INC	XCOR2	voor de volgende kolom.
06FA AD 02 17	WACHT3		LDA	STUUR	Wacht op synclups
06FD 10 FB			BPL	WACHT3	van de TV.
06FF A5 B8			LDA	XCOR2	
0701 8D 00 17			STA	ADRES	Klok het nieuwe
0704 EE 02 17			INC	STUUR	X-adres in.
0707 EE 02 17			INC	STUUR	
070A A9 FF			EOR	#\$FF	
070C 8D 00 17			STA	ADRES	Klok het gedraaide
070F A9 00			LDA	#S00	X-adres in.
0711 8D 02 17			STA	STUUR	
0714 C6 B5			DEC	BREED3	Als hetzelfde byte
0716 D0 94			BNE	ZELFDE	'n keer moet: naar 'ZELFDE'.
0718 F0 26			BEQ	NIBYTE	Onvoorwaardelijk
071A F0 8C	ANDER2		BEQ	ANDERE	Tussensprong voor
071C D0 8A			BNE	ANDERE	long branches
071E 50 07	ACHTGR		BVC	NIET	Achtergr. tekenen?
0720 A5 B8			LDA	ACHTER	
0722 8D 02 17			STA	STUUR	Stel achtergrond in.
0725 70 A4			BVS	WACHT2	Onvoorwaardelijk
0727 C6 B6	NIET		DEC	YCOR2	Verplaats de pointer
0729 CA			DEX		zonder te tekenen.
072A D0 FB			BNE	NIET	
072C F0 C2			BEQ	NIBIT	Onvoorwaardelijk
072E A5 B6	YKLOK		LDA	YCOR2	
0730 8D 00 17			STA	ADRES	
0733 A9 02			LDA	#S02	Klok het Y-adres
0735 0D 02 17			ORA	STUUR	in.
0738 8D 02 17			STA	STUUR	
073B EE 02 17			INC	STUUR	Schrijfpuls
073E D0 A6			BNE	RUST	Onvoorwaardelijk
0740 24 B2	NIBYTE		BIT	VLAG	Bij spiegelen verder
0742 30 1C			BMI	SYMR	op 'SYMR'.
0744 C8			INY		Volgende byte
0745 CA C3			CNY	LENGTE	Zolang laatste byte
0747 D0 D1			BNE	ANDER2	niet is geweest: terug.
0749 24 B2			BIT	VLAG	Als we klaar zijn met
074B 70 3F			BVS	KLAAR	spiegelen, of als er
074D 24 C2			BIT	MODE	geen symmetrie gevraagd
074F 10 3B			BPL	KLAAR	wordt naar 'KLAAR'.
0751 88			DEY		
0752 A9 80			LDA	#S80	Vlag op rechte
0754 85 B2			STA	VLAG	symmetrie.
0756 A9 20			LDA	#S20	
0758 24 C2			BIT	MODE	
075A 70 09					

0763 30 27	BMI KLAAR	Onvoorwaardelijk
0765 A5 B7	KANT LDA YCOR3	
0767 A6 A3	LDX BREED2	
0769 38	SEC	
076A E9 08	COR SBC #S08	Coördinaten instellen op volgende aansluitende karakterplaats.
076C CA	DEX	
076D D0 FB	BNE COR	
076F 85 B6	STA YCOR2	
0771 85 B7	STA YCOR3	
0773 A5 A2	LDA YCOR	
0775 8D 00 17	STA ADRES	
0778 A2 02	LDX #S02	Nieuwe adressen inklokken.
077A 8E 02 17	STX STUUR	
077D A0 00	LDY #S00	
077F 8C 02 17	STY STUUR	
0782 49 FF	EOR #SFF	
0784 85 B8	STA XCOR2	XCOR2 goed zetten
0786 A9 40	LDA #S40	
0788 85 B2	STA VLAG	Zet 'VLAG' op gekantelde symmetrie
078A D0 8E	BNE ANDER2	Onvoorwaardelijk
078C A5 C3	KLAAR LDA LENGTE	
078E 85 B1	STA TEMP	Lengte in 'TEMP'
0790 24 C2	BIT MODE	
0792 50 06	BVC TEST	
0794 08	PHP	
0795 A9 08	LDA #S08	Indien gekanteld: karakterlengte is 8
0797 85 B1	STA TEMP	
0799 28	PLP	
079A 10 0A	TEST BPL GEWOON	Is er symmetrie?
079C 06 B1	ASL TEMP	Bij symm. lengte x 2
079E A9 20	LDA #S20	
07A0 24 C2	BIT MODE	
07A2 F0 02	BEQ GEWOON	Bij enkele middenpoot lengte - 2
07A4 C6 B1	DEC TEMP	
07A6 A6 A3	GEWOON LDX BREED	
07A8 A5 A1	LDA XCOR	
07AA 18	LOOP1 CLC	
07AB 65 B1	ADC TEMP	Bereken de nieuwe X-coördinaat, en zet hem in 'TEMP'
07AD CA	DEX	
07AE D0 FA	BNE LOOP1	
07B0 85 B1	STA TEMP	
07B2 24 A5	BIT KLEUR	
07B4 50 4B	BVC GEACHT	
07B6 A5 A1	LDA XCOR	
07B8 85 B8	STA XCOR2	Coördinaten van het te vullen vlak.
07BA A5 BD	LDA YGRH	
07BC 85 B7	STA YCOR3	
07BE 24 C2	BIT MODE	
07C0 50 28	BCV NIEKAN	Indien niet gekanteld naar 'NIEKAN'
07C2 A5 C3	LDA LENGTE	
07C4 C9 08	CMP #S08	Indien hoger dan 8 maar 'NIEVUL'
07C6 B0 0F	BCS NIEVUL	
07C8 A5 A2	LDA YCOR	
07CA A6 A4	LDX HOOG	
07CC 38	SEC	
07CD E5 C3	LOOP2 SBC LENGTE	Bereken de gekantelde hoogte
07CF CA	DEX	
07D0 D0 FA	BNE LOOP2	
07D2 85 B6	STA YCOR2	
07D4 20 58 08	NIEVUL JSR TKNBLK	Teken boven het kar.
07D7 A5 BC	LDA YGRL	Indien niet geschoven strook onder het karakter tekenen.
07D9 C5 A2	CMP YCOR	
07DB F0 24	BEQ GEACHT	
07DD A5 BC	NIGESC LDA YGRL	
07DF 85 B6	STA YCOR2	
07E1 A5 A2	LDA YCOR	Vul de strook onder het karakter.
07E3 85 B7	STA YCOR3	
07E5 20 58 08	JSR TKNBLK	
07E8 F0 17	BEQ GEACHT	Onvoorwaardelijk
07EA A5 BC	NIEKAN LDA YGRL	Indien niet geschoven onderstrook gaan tekenen.
07EC C5 A2	CMP YCOR	
07EE D0 ED	BNE NIGESC	
07F0 A5 A4	LDA HOOG	
07F2 0A	ASL	
07F3 0A	ASL	Bereken karakterhoogte.
07F4 0A	ASL	
07F5 85 B3	STA HOOG2	
07F7 38	SEC	
07F8 A5 A2	LDA YCOR	
07FA E5 B3	SBC HOOG2	Vul de strook boven het karakter.
07FC 85 B6	STA YCOR2	
07FE 20 58 08	JSR TKNBLK	
0801 A5 BC	GEACHT LDA YGRL	
0803 C5 A2	CMP YCOR	Was het karakter geschoven?
0805 D0 09	BNE VERDER	
0807 A6 A4	LDX HOOG	
0809 C6 A2	LOOP3 DEC YCOR	

080B C6 A2	DEC YCOR	Zet YCOR weer op de oude waarde.
080D CA	DEX	
080E D0 F9	BNE LOOP3	
0810 A5 B1	VERDER LDA TEMP	
0812 85 B8	STA XCOR2	Bereken nieuwe letterplaats voor variabele letterafstand
0814 18	CLC	
0815 65 A6	ADC SPATIE	
0817 85 B1	STA TEMP	
0819 A9 02	LDA #S2	Moet er vaste letterafstand ingesteld worden?
081B 24 A5	BIT KLEUR	
081D F0 09	BEQ VARI	
081F 08	PHP	
0820 A5 A1	LDA XCOR	Bereken de nieuwe letterplaats voor vaste letterafstand
0822 18	CLC	
0823 65 A7	ADC LTRAF	
0825 85 B1	STA TEMP	
0827 28	PLP	
0828 A5 B8	VARI LDA XCOR2	
082A C5 B1	CMP TEMP	Ruimte tussen de letters opvullen?
082C B0 0D	BCS OVER	
082E 50 0B	BVC OVER	
0830 A5 BC	LDA YGRL	
0832 85 B6	STA YCOR2	Vul het vlak vanaf laatste karakter tot de nieuwe letterplaats.
0834 A5 BD	LDA YGRH	
0836 85 B7	STA YCOR3	
0838 20 58 08	JSR TKNBLK	
083B A5 B1	OVER LDA TEMP	
083D C5 D0	CMP KNTLAC	Is de achter-kantlijn gepasseerd?
083F 90 14	BCC FINE	
0841 A9 20	LDA #S20	
0843 24 A5	BIT KLEUR	Automatisch CR en LF?
0845 F0 10	BEQ KLAAR	
0847 A6 A4	LDX HOOG	
0849 A5 A2	LDA YCOR	
084B 18	LOOP4 CLC	
084C 69 0A	ADC #S0A	Verzorg een LF.
084E CA	DEX	
084F D0 FA	BNE LOOP4	
0851 85 A2	STA YCOR	
0853 A5 D1	LDA KNTLVO	Verzorg een CR
0855 85 A1	FINE STA XCOR	
0857 60	KLAAR RTS	Terug naar hoofdprog.
0858 A5 BB	TKNBLK LDA ACHTER	
085A 8D 02 17	STA STUUR	Stel begincoördinaten in.
085D A4 B6	LDY YCOR2	
085F A6 B8	LDX XCOR2	
0861 AD 02 17	LOOP5 LDA STUUR	
0864 10 FB	BPL LOOP5	
0866 8E 00 17	STX ADRES	
0869 EE 02 17	INC STUUR	
086C EE 02 17	INC STUUR	
086F 8C 00 17	STY ADRES	Vul een vlak, begrensd door XCOR2, TEMP YCOR2 en YCOR3.
0872 EE 02 17	INC STUUR	
0875 8D 02 17	STA STUUR	
0878 E8	INX	
0879 E4 B1	CPX TEMP	
087B D0 E4	BNE LOOP5	
087D 88	DEY	
087E C4 B7	CPY YCOR3	
0880 D0 DD	BNE LOOP5	
0882 60	RTS	Terug.

De gebruikte (zero page) locaties zijn in vier groepen onder te verdelen:

Programma constanten

Dit zijn variabelen die doorgaans alleen aan het begin van het totale programma goed moeten worden gezet.

00D0 KNTLAC Kantlijn voor en
00D1 KNTLVO achter
00D2 KARLO Beginadres van de
00D3 KARHI karakterset
00D4 TBLLO Beginadres van de (bij de karakterset
00D5 TBLHI behorende) tabel –\$20

Stuur getallen

Deze variabelen beïnvloeden rechtstreeks de manier waarop een karakter op het scherm verschijnt.

00A0 ASCII geeft aan welk karakter
00A1 XCOR X-coördinaat van het karakter
00A2 YCOR Y-coördinaat van het karakter
00A3 BREED Verbredingsfactor
00A4 HOOG Verhogingsfactor
00A5 KLEUR Acht stuurbits (zie RB mei 1979)
00A6 SPATIE Ruimte tussen twee letters (Variabele afstand)
00A7 LTRAF De afstand tussen het begin van de ene letter en het begin van de andere letter (Vaste afstand)

Karakterconstanten

Deze variabelen kunnen naar keuze door de subroutine zelf bepaald worden, of (voor speciale effecten) door een ander programma. In het laatste geval wordt niet naar 'ASCII' gekeken.

00C0 SUMLO start adres van het
00C1 SUMHI te tekenen karakter
00C2 MODE Kantelen/Schuiven/spiegelen?
00C3 LENGTE Aantal bytes dat het karakter in de karakterset nodig heeft.

Interne variabelen

Deze geheugenplaatsen worden alleen gebruikt tijdens het uitvoeren van het programma.

00B1 TEMP Tijdelijk register
00B2 VLAG Wordt geset bij 2e helft symmetrie
00B3 HOOG2 \
00B4 BREED2 |

00B5 BREED3 | Hulpcoördinaten in verband met kantelen
00B6 YCOR2 | en terugzetten van beginwaarden
00B7 YCOR3 |
00B8 XCOR2 /
00B9 TEL Aantal uit te schuiven bitjes
00BA VOOR Voorgrondkleur (zwart, wit, inverteren)
00BB ACHTER Achtergrondkleur (zwart, wit)
00BC YGRL Onderste begrenzing tekstbalk
00BD YGRH Bovenste begrenzing tekstbalk

RECTIFICATIE

In lijst 1 van RB juni 79 is helaas een foutje geslopen, op adres 06B7 moet B2 staan i.p.v. B4 ('VLAG').

Tabel Character map

ASCII value Character		ASCII-table 2 ASCII table 1	
hex		hex	hex
20	space	82	00 00
21	!	01	BE
22	"	A2	0E 00
23	#	A3	28 FE 28
24	\$	05	48 54 FE 54 24
25	%	05	C6 26 10 C8 C6
26	&	05	6C 92 AC 40 A0
27	'	01	0E
28	(	03	38 44 82
29	)	03	82 44 38
2A	*	A3	44 28 FE
2B	+	A3	10 10 7C
2C	,	22	80 60
2D	-	82	10 10
2E	.	01	80
2F	/	05	C0 20 10 08 06
30	0	05	7C A2 92 8A 7C
31	1	03	8C FE 80
32	2	04	C4 A2 92 8C
33	3	04	44 92 92 6C
34	4	04	1E 10 10 FC
35	5	04	9E 92 92 62
36	6	04	7C 92 92 64
37	7	04	C2 22 12 0E
38	8	82	6C 92
39	9	04	4C 92 92 7C
3A	:	01	28
3B	;	02	80 68
3C	<	04	10 28 44 82
3D	=	82	28 28
3E	>	04	82 44 28 10
3F	?	05	04 02 B2 12 0C
40	@	05	7C 82 BA B2 9C
41	A	82	FC 12
42	B	04	FE 92 92 6C
43	C	04	7C 82 82 44
44	D	04	FE 82 82 7C
45	E	04	FE 92 92 82
46	F	04	FE 12 12 02
47	G	04	7C 82 A2 64
48	H	82	FE 10

49	I	A2	82 FE
4A	J	04	60 80 80 7E
4B	K	05	FE 10 28 44 82
4C	L	04	FE 80 80 80
4D	M	A3	FE 04 08
4E	N	05	FE 04 08 10 FE
4F	O	82	7C 82
50	P	04	FE 12 12 0C
51	Q	05	7C 82 A2 42 BC
52	R	04	FE 12 32 CC
53	S	04	4C 92 92 64
54	T	A3	02 02 FE
55	U	82	7E 80
56	V	A3	3E 40 80
57	W	A3	7E 80 70
58	X	A3	C6 28 10
59	Y	A3	06 08 F0
5A	Z	05	C2 A2 92 8A 86
5B	[	04	FE 82 82 82
5C	\	05	06 08 10 20 C0
5D	]	04	82 82 82 FE
5E	↑	A3	08 04 FE
5F	—	61	F0
60	`	03	00 01 02
61	a	04	40 A8 A8 F0
62	b	04	FE 88 88 70
63	c	03	70 88 88
64	d	04	70 88 88 FE
65	e	04	70 A8 A8 10
66	f	03	08 FC 0A
67	g	24	1C A2 A2 7E
68	h	04	FE 08 08 F0
69	i	01	FA
6A	j	03	40 80 7A
6B	k	04	FE 20 50 88
6C	l	03	82 FE 80
6D	m	05	F8 08 F0 08 F0
6E	n	04	F8 08 08 F0
6F	o	82	70 88
70	p	24	FE 22 22 1C
71	q	24	1C 22 22 FE
72	r	04	F8 10 08 08
73	s	04	90 A8 A8 48
74	t	03	08 7E 88
75	u	82	78 80

76	v	A2	78 80
77	w	A3	78 80 60
78	x	A3	88 50 20
79	y	24	1E A0 A0 7E
7A	z	04	88 C8 A8 98
7B	{	03	10 6C 82
7C		01	FF
7D	}	03	82 6C 10
7E		43	80 70 08
7F	⋮	04	AA 54 AA 54