

Part of the dutch translation of the MSX Red book

Copy supplied by Michiel de Vries

Scanned and converted to PDF by HansO, 2001

DVERZICHT VAN DE JUMP-TABEL

Adres	naam	naar	functie
0000H	CHKRAM	02D7H	controleer RAM bij aanzetten
0004H	-	-	2 byte-adres van karakterset in ROM
0006H	-	-	nummer van UDP data-poort
0007H	-	-	nummer van UDP data-poort
0008H	SYNCHR	2683H	controleer karakter in BASIC programma
0009H	-	-	NOP
000CH	ROSLI	0186H	lees RAM in een willekeurig slot
0010H	CHARSTR	2686H	haal volgende BASIC-karakter
0013H	-	-	NOP
0014H	WRSLI	01D1H	Schrijf naar RAM in willekeurig slot
0017H	-	-	NOP
0018H	OUTDO	1845H	stuur output naar huidig kanaal
0019H	-	-	NOP
001CH	CALSLI	0217H	CALL een routine in willekeurig slot
001FH	-	-	NOP
0020H	DCOMPR	146AH	vergelijk registratoren HL en DE
0023H	-	-	NOP
0024H	ENASLI	025EH	maak een bepaald slot permanent actief
0027H	-	-	NOP
0029H	GETYPR	2689H	zoek type van BASIC operand
002BH	-	-	vijf bytes : nr. van de versie
002EH	CALLF	0205H	CALL een routine in een bepaald slot
0030H	-	-	5 maal NOP
0033H	-	-	interrupt-routine, aftasting toetsenbord
0038H	KEYINI	0C3CH	interrupteer I/O hardware
003BH	INITIO	045DH	initialiseer functietoetsen
003EH	INIFNK	135DH	initialiseer functietoetsen
0041H	DISSCR	0577H	maak scherm inactief
0044H	ENASCR	0570H	maak scherm actief
0047H	WRTUDP	057FH	schrijf naar een UDP register
0049H	RDVRM	07D7H	lees een byte uit URAM
004DH	WRTVRM	07CDH	schrijf een byte naar URAM
0050H	SETVRM	07ECH	maak UDP klaar voor lees-operatie
0053H	FILVRM	0815H	vul een URAM-blok met een bepaalde waarde
0056H	LDVRM	07DFH	kopieer blok URAM naar geheugen
0059H	LDVRM	0744H	kopieer blok geheugen naar URAM
005CH	CHGUD	084FH	verander de UDP-mode
005FH	CHGCLR	0777H	verander de UDP-kleuren
0062H	-	-	NOP
0065H	NTI	1398H	Non-maskable interrupt-routines
0068H	CLRSFR	0648H	wis alle sprites
006CH	INITXI	050EH	initialiseer UDP voor 40 x 24 tekstmode
006FH	INIT32	0538H	initialiseer UDP voor 32 x 24 tekstmode
0072H	INIGRP	05D2H	initialiseer UDP voor grafische mode
0075H	ININLI	061FH	initialiseer UDP voor veelkleuren-mode
0078H	SETIXI	0594H	stel UDP in op 40 x 24 tekstmode
007BH	SETI32	05B4H	stel UDP in op 32 x 24 tekstmode
007EH	SETIGRP	0602H	stel UDP in op grafische mode
0081H	SETINLI	0655H	stel UDP in op veelkleurenmode
0084H	CALPAT	06E4H	bereken adres van sprits-patroon
0087H	CALATR	06F8H	bereken adres van sprits-attributen
008AH	GSPTSIZ	0704H	zoek grootte van sprite
008DH	GRPPAT	1510H	print teken op grafisch scherm

Deze datagebieden gelden voor de Engelse ROM. In de Japanse ROM zijn kleine verschillen, die betrekking hebben op de decodering van het toetsenbord en de karakterset. De ROM's verschillen enkel van elkaar wat deze gebieden betreft; het grote deel van de code is voor beide gevallen identiek.

Terminologie

In dit hoofdstuk wordt dikwijls verwezen naar standaard-routines en naar de variabelen in het werkgebied (workspace). Bij die verwijzingen gebruiken we de naam die Microsoft aanbeveelt, of hoofdletters. Bijvoorbeeld: "de standaardroutine FILVRM" of "SCRINDB wordt geset". Naar subroutines zonder naam wordt verwezen middels een adres tussen haakjes: "het scherm wordt gewist (0777h)". Wanneer de 280 statusvlaggen worden vermeld, gebruiken we conventionele aanduidingen uit de assembler-taal. De uitdrukking "vlag C" betekent bijvoorbeeld dat de Carry-vlag geset wordt; "vlag NZ" wil zeggen dat de Zero-vlag geset wordt. De termen "E1" en "D1" duiden respectievelijk aan dat de interrupts in - of buiten werking worden gesteld.


```

adres 01B6H
-----
naam RDSLI
in A-slot-identificatie, HL-adres
uit A-gelezen byte
wijzigt AF,BC,DE,DI

```

Standaard-routine om een byte in het geheugen te lezen, in elk slot. De slot-identificatie bestaat uit het nummer van het primaire slot, het nummer van het secundaire slot, en een vlag.

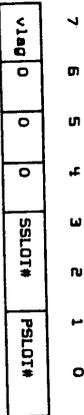


Fig. 34 : slot-identificatie

De vlag is normaal 0, maar moet 1 zijn indien in de identificator een secundair slot-nummer vervat zit. Eerst worden het adres en de identificator verwerkt (027EH) tot een aantal bitmaskers, die met het betrokken slotregister vergeleken zullen worden. Vervolgens wordt de identificator verwerkt (0293H), dan wordt eerst het secundaire slotregister zo gewijzigd, dat de gelezen data van dat slot geleest kan worden (0293H). Dan wordt het primaire slot in het geheugen bereikt van de ZBO geschakeld, het byte wordt gelezen en het primaire slot hersteld. In zijn oorspronkelijke staat via de ROPRIN-routine in het geheugen. In slot 0 wordt nog, indien een secundair slotnummer in de identificator was opgenomen, het secundaire slotregister in zijn oorspronkelijke staat hersteld (01ECH).

Let er wel op dat, tenzij het om het slot gaat dat het hergeheel bevat, elke poging om uit geheugenbladzijde 3 te lezen (0000H tot FFFH) een crash veroorzaakt, doordat ROPRIN zichzelf buiten het bereik van de ZBO schakelt. Alle routines die geheugen in- of uitschakelen stellen ook de interruptie buiten werking.

```

adres 01D1H
-----
naam WRSLI
in A-slot-identif., HL-adres, E-te schrijven byte
uit n.v.t.
wijzigt AF,BC,D,DI

```

Dit is de standaard-routine om een byte in het geheugen te schrijven, in een willekeurig slot. De werking ervan is fundamenteel dezelfde als van de RDSLI-routine, behalve dat in het geheugen de WRPRIN-routine in plaats van ROPRIN gebruikt wordt.

```

adres 01FFH
-----
naam CALBMS
in IX-adres
uit n.v.t.
wijzigt AF,BC,DE,HL,IV,DI

```

Deze standaard-routine roept een subroutine aan in de BRSIC Interpreter, veruult elk slot. Dit zal altijd gebeuren vanuit een machinaal-programma dat in het uitbreidingsregister op bladzijde 1 (7000H tot 7FFFH) loopt. Het register van IV wordt gelezen met de identificator van het NSX-Register (00H), en de routine gaat verder naar CALSLI.

```

adres 0205H
-----
naam CALLF
in n.v.t.
uit n.v.t.
wijzigt AF,BC,DE,HL,IX,IV,DI

```

Via deze standaardroutine kan een CALL naar elk adres in elk slot gebeuren. Slot-identificatie en het CALL-adres worden als parameters gegeven en niet via registers, om als hook gebruikt te kunnen worden (zie hoofdstuk 6). Een voorbeeld :

```

RST 30H
DEFB (slot ident.)
DEFB (adres)
RET

```

Eerst wordt de slot-identificatie opgehaald, en in het NSB van de controle overgegeven wordt het adres in IX gelezen, waarna de

```

adres 0217H
-----
naam CALSLI
in IX(NSB)-slot-identif., IX-adres
uit n.v.t.
wijzigt AF,BC,DE,HL,DI

```

Dit is de eigenlijke algemene standaardroutine waarmee een CALL kan plaatsvinden naar elk slot. Ze werkt in de grond op dezelfde manier als RDSLI, behalve dat nu CLPRIN in het CALLF zijn enkel gebruikt, in plaats van ROPRIN. CALBMS en CALLF zijn enkel speciale toegangs-adressen tot deze routine, die maken dat de programmeur zelf minder code moet schrijven.

```

adres 025EH
-----
naam ENASLI
in A-slot-identif., HL-adres
uit n.v.t.
wijzigt AF,BC,DE,DI

```

Deze standaardroutine schakelt een bladzijde vanuit elk slot in het geheugen. Het wordt gebruikt in de processor. In tegenstelling tot de RDSLI, WRSLI en CALSLI-routines gebeurt

hier het inschakelen van het primaire slot rechtstreeks, en niet via een routine in het werkgebied. Dit houdt in dat het opgeven van een adres op bladzijde 0 (0000H tot 3FFFH) voor een ogenblikkelijke crash zorgt.

adres 027EH

Op dit adres start een routine die gebruikt wordt door alle routines die geheugen schakelen. De routine bewaakt een adres (in HL) en een slot-identificatie ("FxxxSP") en een groep bitmaskers. Een slot-identificatie "FxxxSP" en een adres op bladzijde 1 (tussen 4000H en 7FFFH) zou bijvoorbeeld het volgende resultaat opleveren :

```
register B : 00 00 PP 00 (DR-masker)
register C : 11 11 00 11 (AMD-masker)
register D : PP PP PP PP (geduplicieerd)
register E : 00 00 11 00 (bladzijde-masker)
```

De B en C registers worden afgelijnd van het nummer van het primaire slot en het bladzijde-masker. Later worden deze registers gebruikt om het nieuwe primaire slot-nummer in te voegen in de bestaande inhoud van het primaire slot-register. Register D bevat vier keer het primair slot-nummer, en het E-register bevat het bladzijde-masker. Dit masker wordt gemaakt aan de hand van de hoogste twee bits van het adres (om de bladzijde te bepalen). Dan worden de bits naar de passende plaats verschoven. Deze registers worden later gebruikt tijdens het inschakelen van de secundaire slots.

Op het einde van deze routine wordt bit 7 van de slot-identificatie getest, om te zien of er een secundair slot werd gespecificeerd. Was dit zo, dan wordt de H vlag gezet.

adres 02A3H

Deze routine wordt gebruikt door de standaard-routines die geheugen omschakelen, om de inhoud van een secundair slot-register te wijzigen. De slot-identificatie wordt in A verschaft; registers D en E bevatten de bitmaskers, zoals in de vorige routine.

Bits 6 en 7 van D worden eerst naar het primair slotregister gecopieerd. Daardoor wordt bladzijde 3 ingeschakeld van het primair slot dat door de slot-identificatie werd bepaald, en wordt het secundair slotregister geïnverteerd. Dit wordt dan gedaan (adres FFFF) en met het geïnverteerde bladzijde-masker worden de gewenste twee bits vrijgemaakt. Het secundair slot wordt tot op de goede plaats geschoven en eras nummer wordt toegevoegd. De verkeerde combinatie wordt in het secundair slotregister geschreven, en het primair slotregister wordt in zijn oorspronkelijke stand teruggezet.

adres 02D7H

```
naam  CKRPH
in     n.v.t.
uit    n.v.t.
wijzigt Af, Bc, DE, HL, SP
```

40

Deze standaardroutine initialiseert het geheugen bij het aanzetten van de machine. Ze controleert op een niet-destructieve manier of er RMI aanwezig is op bladzijden 2 en 3 van de zesien mogelijke slots. Daarna beschrijft ze de primaire en secundaire slotregisters zodanig dat het grootste gevonden gebied in het geheugenbereik van de processor geschakeld wordt. Het hele werkgebied wordt met nullen gevuld (van F80H tot FCFH). In EXPTBL en SLTBL wordt ingevuld welke uitbreidings-interfaces aanwezig zijn. Interrupt mode 1 wordt ingesteld, en de routine springt naar de rest van de initialiseringsroutine op 7C76H.

adres 03FBH

```
naam  ISCNTC
in     n.v.t.
uit    n.v.t.
wijzigt Af, E1
```

Deze standaardroutine controleert of de "CTRL/STOP" of de "STOP"-toetsen werden ingedrukt. De BASIC interpreter gebruikt ze op het einde van elk statement, om te zien of het programma beëindigd dient te worden. Vooraft wordt de inhoud van BASROM gelezen. Verschilt die van nul, dan wordt de routine afgebroken. Dit heeft tot doel, te voorkomen dat gebruikers inbreken in uitbreidings-ROM's die BASIC programma's bevatten. Verder wordt INFLG continu uitgelezen, om te zien of de interrupt-routines de codes van "CTRL/STOP" of "STOP" daar hebben gezet (codes 03H of 07H). Wordt "STOP" ingedrukt, dan wordt de cursor zichtbaar gemaakt (OSDAM) en wordt INFLG voortdurend bekeken tot een van beide codes weer voorkomt. Dan wordt de cursor van het scherm gehaald (02A7H) en als de "STOP"-toets werd ingedrukt stopt de routine.

Wordt een "CTRL/STOP"-code ontdekt, dan wordt de toetsenbord-bufler leeggemaakt via de KILBUF-routine. In TRPTBL wordt nagegaan of een "ON STOP GOSUB"-statement actief is. Is dat het geval, dan wordt het daabetreffende deel van TRPTBL aangepast (OE7H) en de routine stopt : de "GOSUB" wordt verder door de interpreter afgehandeld. Wordt "CTRL/STOP" ontdekt, dan zorgt de ENGLI-routine ervoor dat bladzijde 1 van de NSX-ROM ingeschakeld wordt - voor het geval de routine door een uitbreidingsROM wordt gebruikt - en de routine wordt overgedragen aan de "STOP"-statement routine (63E6H).

adres 046BH
naam KILBUF
in n.v.t.
uit n.v.t.
wijzigt HL

Deze standaardroutine maakt de toetsenbord-bufler leeg. Die bufler (KEYBUF) kan 40 karakters bevatten, die op voorhand ingeloopt kunnen worden en één na één verwerkt worden. Bij die bufler horen twee pointers : PUTPNT en GETPNT. PUTPNT duidt aan waar de interrupt-routines een ingetocst karakter moeten zetten, terwijl GETPNT aangeweest op welk adres in de bufler de toepassingsprogramma's karakters kunnen uitlesen. Het aantal karakters in de bufler wordt aangegeven door het verschil tussen

41

deze twee pointers, KEYBUF wordt leeggemaakt door simpelweg deze twee pointers gelijk te maken.

adres 046FH
naam BREAKX
in n.v.t.
uit n.v.t.
wijzigt Af

Deze standaardroutine pakt rechtstreeks rij 6 en 7 van het toetsenbord, om te zien of CTRL en "STOP" tegelijk worden ingedrukt. Is dat zo, dan wordt KEYBUF leeggemaakt en rij 7 van OLDEY aangepast, dit laatste om te voorkomen dat de Interrupt-routines de toetsen zouden detecteren. Voor toepassings-programma's is deze routine vaak beter te gebruiken dan ISCNIC, omdat ze ook werkt wanneer de Interrupts niet worden ontvangen (bijvoorbeeld tijdens input of output naar cassette) en de controle rechtstreeks aan het programma wordt teruggegeven, en niet via de Interpreter.

adres 048DH
naam INITIO
in n.v.t.
uit n.v.t.
wijzigt Af,EI, EI

Deze standaardroutine initialiseert de PSG en de Centronics status-poort. Register 7 van de PSG wordt op 80H gezet, waardoor poort 8 van de PSG voor output en poort 9 voor input zal gebruikt worden. Register 15 van de PSG wordt met CFH geladen, om de hardware die de joystick-aansluiting stuurt, te initialiseren. Register 14 van de PSG wordt uitgelezen en het keyboard mode bit wordt in KANMOD gezet. Dit is van geen belang voor Europese machines.

Tenslotte wordt FPH op de Centronics statuspoort gezet (in/Dut-poort 50H), waardoor het strobesignaal hoog wordt. Dan gaat de routine naar GICINI, waar de initialisatie afgevoerd wordt.

adres 048DH
naam GICINI
in n.v.t.
uit n.v.t.
wijzigt EI

Deze standaardroutine initialiseert de PSG en de variabelen in het werkgedaal die te maken hebben met het "PLAY"-statement. QUEIRB, UCRA, UCBB, en UCBC worden eerst op de waarden gezet die in hoofdstuk 6 gegeven worden. De registers 8, 9 en 10 van de PSG worden op ampitude 0 gezet, en register 7 op 8BH. Daardoor wordt de toongenerator op elk kanaal aangesloten, en de toongenerator van elk kanaal afgesloten.

adres 0508H TABEL
Deze tabel van zes bytes bevat de parameters voor het "PLAY"-statement zoals die oorspronkelijk in UCBA, UCBB en UCBC werden gezet door de standaardroutine GICINI : octaal 4, lengte 4, tempo 120, volume 88H, envelope 00FFH.

adres 0509H
naam INITXI
in n.v.t.
uit n.v.t.
wijzigt Af,BC,DE,HL,EI

Door deze standaard routine wordt de UDP geïnitialiseerd voor de 40 x 24 tekstmode. Het scherm wordt tijdelijk buiten werking gesteld via de standaardroutine DISSCR; SCRND en OLDSCR worden op 0 gezet. De parameters, nodig voor de standaardroutine CPUFI worden ingevuld door LINL40 naar LINLEN, IXIM41 naar NARIB5 en IXICGP naar CGPB65 te kopiëren. De kleuren voor de UDP worden vervolgens ingevuld door de standaardroutine CHGCLR en het scherm wordt gewist (077EH). De karakterzet in gebruik wordt gecopieerd naar de Karaktertabel in VRM (071EH). Tenslotte worden de UDP-mode en de basis-adressen vastgelegd door de standaardroutine SETXI, en het scherm wordt opnieuw in werking gesteld.

adres 0538H
naam INIT32
in n.v.t.
uit n.v.t.
wijzigt Af,BC,DE,HL,EI

Door deze standaardroutine wordt de UDP geïnitialiseerd voor de 32 x 24 tekstmode. Het scherm wordt tijdelijk buiten werking gesteld via de standaardroutine DISSCR en SCRND en OLDSCR worden op 01H gezet. De parameters, nodig voor de standaardroutine CPUFI worden ingevuld door LINL32 naar LINLEN, IXIM41 naar NARIB5, IXICGP naar CGPB65, IXSP61 naar PATB65 en IXEMIR naar AIRB65 te kopiëren. De kleuren voor de UDP worden vervolgens ingevuld door de standaardroutine CHGCLR en het scherm wordt gewist (077EH). De karakterzet in gebruik wordt gecopieerd naar de Karaktertabel in VRM (071EH) en alle sprites worden gewist. (068BH). Tenslotte worden de UDP-mode en de basisadressen vastgelegd door de standaardroutine SETI32, en het scherm wordt opnieuw actief gemaakt.

adres 0570H
naam EWSGR
in n.v.t.
uit n.v.t.
wijzigt Af,BC,EI

Deze standaardroutine maakt het scherm actief, door bit 6 van het UDP-moderegister 1 te zetten.

adres 0577H
 naam DISSCR
 in n.v.t.
 uit n.v.t.
 wijzigt Af,BC,DE,HL,EI

Deze standaardroutine maakt het scherm inactief, door bit 6 van het UDP moderegister 1 te resetten.

adres 057FH
 naam URUDP
 in B-data byte, C-nummer van UDP moderegister
 uit n.v.t.
 wijzigt Af,B,EI

Deze standaardroutine wordt gebruikt om een byte in een van de UDP-moderegisters te zetten. Eerst wordt het registernummer naar de commandoport van de UDP geschreven, en dan het databyte. Dit wordt daarna gecopieerd naar het werkgebied, R605h tot R675h, waar de inhoud van de registers bijgehouden wordt.

F30F
 adres 0594H
 naam SETIXI
 in n.v.t.
 uit n.v.t.
 wijzigt Af,BC,DE,HL,EI

Het deze standaardroutine wordt de UDP gedeeltelijk klaargemaakt voor de 40 x 24 tekstmode. De Hodebits H1, H2 en H3 in de UDP-registers 0 en 1 worden in de goede stand gezet. Daarna worden de vijf basisadressen van de tabellen in UR4h, te beginnen met IXINH, van het werkgebied gecopieerd naar de moderegisters 2, 3, 4, 5 en 6 (0577H).

adres 0584H
 naam SETI32
 in n.v.t.
 uit n.v.t.
 wijzigt Af,BC,DE,HL,EI

Het deze standaardroutine wordt de UDP gedeeltelijk klaargemaakt voor de 32 x 24 tekstmode. De Hodebits H1, H2 en H3 in de UDP-registers 0 en 1 worden in de goede stand gezet. Daarna worden de vijf basisadressen van de tabellen in UR4h, te beginnen met I32NH, van het werkgebied gecopieerd naar de moderegisters 2, 3, 4, 5 en 6 (0577H).

adres 0502H
 naam INIGRP
 in n.v.t.
 uit n.v.t.
 wijzigt Af,BC,DE,HL,EI

Deze standaardroutine initialiseert de UDP voor de grafische mode. Het scherm wordt tijdelijk buiten werking gesteld via de standaardroutine DISSCR en in SCRHD wordt 2 gezet. De parameters, nodig voor de standaardroutine GRPPI, worden klaargezet door het copieren van GRPPI naar P4IBAS en van GRPPI naar P4IBAS. Het pilootpatroon voor de karaktercodes wordt dan gecopieerd naar de Namentabel van de UDP. Het scherm wordt schoneggemaakt (0761H) en alle sprites gewist (0588H). Tenslotte wordt de UDP mode ingesteld en worden de basisadressen aangepast via de standaardroutine SETGRP, en het scherm wordt opnieuw actief gemaakt.

adres 0602H
 naam SETGRP
 in n.v.t.
 uit n.v.t.
 wijzigt Af,BC,DE,HL,EI

Door deze standaardroutine wordt de UDP gedeeltelijk geïnitieerd voor de grafische mode. Hodebits H1, H2 en H3 in de UDP-moderegisters 0 en 1 worden in de goede stand gezet. De vijf basisadressen van de UR4h-tabellen worden dan vanuit het werkgebied naar de UDP-moderegisters 2, 3, 4, 5 en 6 gecopieerd (0577H).

adres 061FH
 naam INIH1I
 in n.v.t.
 uit n.v.t.
 wijzigt Af,BC,DE,HL,EI

Door deze standaardroutine wordt de UDP klaargemaakt voor de veelkleurmode. Het scherm wordt tijdelijk buiten werking gesteld via de standaardroutine DISSCR en in SCRHD wordt 3 gezet. De parameters, nodig voor de standaardroutine GRPPI, worden klaargezet door het copieren van H1IRP naar P4IBAS en van H1IRP naar P4IBAS. Het pilootpatroon voor de karaktercodes wordt gecopieerd naar de Namentabel van de UDP, het scherm wordt schoongemaakt (0761H) en alle sprites gewist (0588H). Tenslotte worden de UDP mode en basisadressen ingevuld via de standaardroutine SETH1I, en het scherm wordt weer actief gemaakt.

adres 0659H
 naam SEH1I
 in n.v.t.
 uit n.v.t.
 wijzigt Af,BC,DE,HL,EI

Deze standaardroutine maakt de UDP gedeeltelijk klaar voor de veelkleurmode. Hodebits H1, H2 en H3 in de UDP-moderegisters 0 en 1 worden in de goede stand gezet. De vijf basisadressen van de UR4h-tabellen, te beginnen met H1INH, worden dan vanuit het werkgebied naar de UDP-moderegisters 2, 3, 4, 5 en 6 gecopieerd.

adres 0677H

 Deze subroutine wordt gebruikt door de standaardroutine SETI32, SETI32, SETI32 en SETI32, om een blok van vijf basisadressen voor tabellen te kopiëren vanuit het werkgehebe naar de VDP registers E, 3, 4, 5 en 6. Wanneer de routine wordt aangeroepen, bevat HL het adres van de groep van vijf adressen. Een na aan worden de basis-adressen opgehaald, het nodige aantal plaatsen verschoven, en dan in het juiste moderegister geplaatst door de standaardroutine WRVDP.

adres 068BH

 naam CLRSPR
 in n.v.t.
 uit n.v.t.
 wijzigt AF,BC,DE,HL,EI

Deze standaardroutine wijst alle sprites. De patroontabel van de sprites wordt eerst over de hele lengte van 2 K met nullen gevuld via de standaardroutine FILVDP. Het vertikale coördinaat van de tweehonderd attributen-blokken van de sprites wordt vervolgens op -47 (D1H) gezet, waardoor de sprite boven de rand van het scherm staat. Het horizontale coördinaat wordt niet gewijzigd.

De patroonnummers in de attributentabel worden geïnitieëerd met de serie 0,1,2,3,4,...,31 voor 8x8 sprites, of met de serie 0,4,8,12,16,...,124 voor 16x16 sprites. Welke serie gebruikt wordt, bepaalt het grootste-bit in VDP moderegister 1. Ten slotte wordt het kleurbyte van elk attributenblok ingevuld met de kleur die in FORCLR zit; oorspronkelijk is dit wit.

Deze routine heeft geen invloed op de grootte- en verspreidings-bits in VDP moderegister 1. Door de standaardroutine INIT32, INIGRP en INITLI deze routine gebruiken vanaf adres 068BH, veranderen ze geen van drieën iets aan de patroontabel.

adres 06E4H

 naam CALPAT
 in A - nummer van sprite-patroon
 uit HL - adres van dat sprite-patroon
 wijzigt AF,DE,HL

Het deze standaardroutine wordt het adres van een sprite-patroon berekend. Eerst wordt het patroon-nummer met acht vermenigvuldigd. Zijn er 16 x 16 sprites gesorteerd, dan wordt dit resultaat nogmaals met vier vermenigvuldigd. Dit resultaat wordt opgeteld bij het basisadres van de patroontabel, dat in PTRBAS staat, en dit geeft het gevraagde adres.

Dit systeem van nummering klopt met het systeem dat de BASIC interpreter gebruikt, maar niet met dat van de VDP wanneer 16x16 sprites gekozen werden. Terwijl de interpreter bijvoorbeeld het tweede patroon als nummer 1 benoemt, is het eigenlijk patroon nummer 4 in de VDP. Dit houdt in dat, wanneer het om 16x16 sprites gaat, deze routine als hoogste patroon-nummer 53 zou

mogen toelaten. In feite wordt dit niet gecontroleerd. Hoge patroon-nummers zullen dus een adres opleveren boven 3FFFH. Wanneer dergelijke hoge adressen door de andere VDP-routines overgenomen worden, zullen die terug in het lagere bereik, vanaf 0000H terecht komen, en de tabel met karakterpatronen in VRAM in de war brengen.

adres 06F9H

 naam CALAIR
 in A - nummer van de sprite
 uit HL - adres van attribut van die sprite
 wijzigt AF,DE,HL

Deze standaardroutine berekent het adres van het attributenblok van een sprite. Het sprite-nummer, tussen 0 en 31, wordt vermenigvuldigd met vier en dit resultaat opgeteld bij het basisadres van de attributentabel, dat in AIRBAS staat.

adres 070FH

 naam GSPSIZ
 in n.v.t.
 uit R - aantal bytes in sprite-patroon (8 of 32)
 wijzigt AF

Deze standaardroutine levert het aantal bytes op, dat elk sprite-patroon in de patroontabel inneemt. Dit wordt bereikt door het grootste-bit in VDP-moderegister 1 te onderzoeken.

adres 070FH

 naam LDIRNV
 in BC-lengte,DE-RAM-adres,HL-VRAM-adres
 uit n.v.t.
 wijzigt AF,BC,DE,EI

Deze standaardroutine copieert een geheugenblok vanuit VRAM naar het RAM-geheugen. Het startadres in VRAM wordt doorgegeven via de standaardroutine SETRD en vervolgens worden de opeenvolgende bytes uitgelezen van de VDP Datapoort, en in het geheugen gezet.

adres 071EH

 Deze routine ken een karakterset van 2 K lang in elke mode naar de VDP Karaktertabel gecopieerd worden. Het basisadres van de tabel in VRAM wordt uit CGPMS gehaald. Het startadres van de karakterset wordt uit CGPNT ingelezen. Om de karakter-data te lezen, wordt de standaardroutine RDSLI gebruikt, zodat de karakterset zich ook in een uitbreidingsRAM mag bevinden.

Bij het inschakelen van de machine wordt in CGPNT het adres gezet, dat op adres 0004H staat, dat wil zeggen: adres 1BBFH. CGPNT kan makkelijk gewijzigd worden, waardoor je soms interessante resultaten verkrijgt. Nogal verwarrend bijvoorbeeld: POKE AHP20, AHC7: SCREEM 0.

adres 0744H
 naam LDIRRU
 in BC - lengte, DE - adres in URAM, HL - adres in RAM
 uit n.v.t.
 wijzigt AF,BC,DE,HL,EI

Deze standaardroutine copieert een geheugenblok van RAM naar URAM. Het startadres in URAM wordt doorgegeven via de standaardroutine SETURI, en vervolgens worden de opeenvolgende bytes uit het geheugen naar de UDP dataport geschreven.

adres 0777H

Deze routine wist het scherm in elke UDP-mode. In 40x24 en 32x24 tekemode wordt eerst de Namentabel gevuld met ASCII-spaties. Het basisadres voor die tabel wordt uit NHTBAS gehaald. Dan wordt de cursor linksbovenaan het scherm gezet (0A7FH), en LINITB (eind van de lijn-tabel) opnieuw geïnitialiseerd. Tenslotte wordt, wanneer dit toegestaan is, de tekst van de functie-toetsen op het scherm gezet door de standaardroutine FMS5B.

In grafische mode wordt eerst de border gekleurd via UDP Register 7 (0B32H). Dan wordt de kleurentabel gevuld met de code voor de achtergrondkleur, die uit BAKCLR gehaald wordt, voor alle pixels (0 en 1). Tenslotte wordt de karaktertabel met nullen gevuld.

In vaeikleurmode wordt eerst de border gekleurd via UDP Register 7 (0B32H), waarna de karaktertabel opgevuld wordt met de achtergrondkleur, die uit BAKCLR gehaald wordt.

adres 07CDH
 naam WRURM
 in A - databyte, HL - adres in URAM
 uit n.v.t.
 wijzigt EI

Het deze standaardroutine kan een byte naar URAM geschreven worden. Eerst wordt het URAM adres doorgegeven via de standaardroutine SETURI, en dan wordt het databyte naar de dataport van de Video Display Processor geschreven. De beide omschrijftelijk overbodige EX (SP) HL instructies in deze routine - en nog in andere routines - zijn nodig in verband met de timing-eisen van de Video Display Processor.

adres 07D7H
 naam RDUURM
 in HL - adres in URAM
 uit A - (HL)
 wijzigt AF,EI

Het deze standaardroutine kan 1 byte uit de URAM gelezen worden. Eerst wordt het adres doorgegeven via de standaardroutine SETRU, en dan wordt de inhoud ervan uitgelezen via de dataport van de Video Display Processor.

adres 07DFH
 naam SETURI
 in HL - adres in URAM
 uit n.v.t.
 wijzigt AF,EI

Deze standaardroutine maakt de Video Display Processor klaar om opeenvolgende bytes naar URAM te schrijven via de dataport. Het adres in HL wordt naar de commandoport van de Video Display Processor geschreven in het formaat: LSB eerst, dan HSB (zie fig. 7). Adressen boven 3FFF tellen door vanaf 0000H, aangezien de hoogste twee bits van het adres niet in aanmerking genomen worden.

adres 07E6H
 naam SETRU
 in HL - adres in URAM
 uit n.v.t.
 wijzigt AF,EI

Deze standaardroutine maakt de Video Display Processor klaar om opeenvolgende bytes uit URAM te lezen via de dataport. Het adres in registerpaar HL wordt naar de commandoport van de Video Display Processor geschreven in het formaat: LSB eerst, dan HSB (zie fig. 7). Adressen boven 3FFF tellen door vanaf 0000H, aangezien de hoogste twee bits van het adres niet in aanmerking genomen worden.

adres 07F7H
 naam CHGCLR
 in n.v.t.
 uit n.v.t.
 wijzigt AF,BC,HL,EI

Deze standaard routine stelt de kleuren voor de Video Display Processor in. Eerst wordt in SCRMDD gekoken wat er moet gebeuren. In de 40x24 tekemode wordt de inhoud van BAKCLR en FORCLR in register 7 van de "1" en "0"-pixels bepaald wordt, waardoor de kleur van de "1" en "0"-pixels bepaald wordt. Daarnaast zijn die kleuren respectievelijk blauw en wit. In deze mode kan geen aparte borderkleur opgegeven worden; grafische mode of vaeikleurmode wordt de inhoud van BDRCLR in register 7 bepaald is. In 32x24 tekemode wordt de inhoud van BAKCLR en FORCLR ook nog in de hele kleurentabel gezet, waardoor de kleur van "1" en "0"-pixels bepaald wordt.

adres 0B15H
 naam FIURM
 in n.v.t.
 uit n.v.t.
 wijzigt AF,BC,EI

Deze standaard routine vult een URAM-blok met een bepaalde

waarde (het databyte in register A). Eerst wordt het startadres in URAM, in register HL, doorgegeven via de standaardroutine SEIWTI. Vervolgens wordt het data byte BC maal naar de dataport van de Video Display Processor geschreven, om BC opeenvolgende URAM-adressen te beschrijven.

```

adres 083BH
-----
naam IOTEXI
in n.v.t.
uit n.v.t.
wijzigt AF,BC,DE,HL,EI

```

Door deze standaard routine wordt de Video Display Processor teruggezet naar 40x24 of 32x24 tekstmode, wanneer hij op het ogenblik dat de routine aangeroepen wordt, in grafische of veelkleuremode staat. De routine wordt door de Hoofdruis van de BASIC Interpreter en de "INPUT"-statement-verwerker gebruikt. Bij elke CALL naar de standaardroutine INITXI of INIT32 wordt het mode byte, 0 of 1, in OLDSCR gecopieerd. Wanneer vervolgens naar grafische mode of veelkleuremode gegaan wordt, en daarna terug naar een van de twee tekstmodes voor input vanaf het toetsenbord, zorgt deze routine ervoor dat de Video Display Processor naar de correcte mode terugkeert.

Vooraf wordt SCRMOD getest. Blijkt dat het scherm al in een tekstmode staat, dan stopt de routine zonder meer. Is dat niet het geval, dan wordt de inhoud van OLDSCR doorgegeven aan de standaardroutine CHGMOD.

```

adres 084BH
-----
naam CLS
in vlag Z
uit n.v.t.
wijzigt AF,BC,DE,EI

```

Deze standaardroutine wist het scherm schoon in elke mode. Deze routine roept 0777H aan. Hier wordt eigenlijk het "CLS"-statement verwerkt. Omdat vlag NZ aangeeft dat er na dit statement nog tekst staat (niet toegelaten), doet de routine niets indien ze aangeroepen wordt met de Zero-vlag op 0.

```

adres 084FH
-----
naam CHGMOD
in A = gewenste scherm-mode (0, 1, 2 of 3)
uit n.v.t.
wijzigt AF,BC,DE,HL,EI

```

Met deze standaardroutine wordt een bepaalde scherm-mode ingesteld. De inhoud van register A wordt bekeken en vervolgens wordt de controle overgedragen aan de standaardroutine INITXI, INIT32, INIGRP of INIMLT.

```

adres 085DH
-----
naam LPTOUT
in A = te printen karakter
uit vlag C indien op CTRL-STOP gedrukt werd
wijzigt AF

```

50

Deze standaard routine stuurt een karakter naar de printer via de Centronics-poort. De toestand van de printer wordt continu onderzocht, via de standaardroutine LPTSTI, tot hij klaar is voor ontvangst. Dan wordt het karakter naar de Centronics Data-poort geschreven (In/Dut-poort 91H) en wordt het Strobo-siginaal van de Centronics Status-poort (In/Dut-poort 90H) kortstondig omlaag gebracht. De BREAKY-routine wordt gebruikt om te testen of "CTRL/STOP" werd ingedrukt, terwijl de printer bezet is. Wordt die toets-combinatie ingedrukt, dan wordt een "CR"-code (0DH) naar de Centronics Data-poort geschreven, om de regelbuffer van de printer leeg te maken, en de routine stopt met vlag C.

```

adres 0864H
-----
naam LPTSTI
in n.v.t.
uit A = 0 en vlag Z als de printer bezet is
wijzigt AF

```

Deze standaardroutine onderzoekt het BUSY-siginaal van de Centronics status-poort. In/Dut-poort 90H wordt uitgelezen en bit 1 wordt bekeken. Is dit 0, dan is de printer klaar, is het 1 dan is hij bezet.

```

adres 088EH
-----
naam POSII
in H = kolom, L = rij
uit n.v.t.
wijzigt AF, EI

```

Deze standaardroutine bepaalt de coördinaten van de cursor. Die coördinaten worden aan de standaardroutine QUITD doorgegeven in deze volgorde: "ESC" "v" rij+1FH, kolom+1FH. Merk op dat de "home"-positie van het BIOS 1,1 is in plaats van 0,0 zoals de BASIC Interpreter gebruikt.

```

adres 089DH
-----
naam CNVCHR
in A = karakter
uit vlag Z, NC = header; vlag NZ, C = grafisch; Z, C = normaal
wijzigt AF

```

Deze standaardroutine controleert karakters met grafische headers en zet ze indien nodig om. Normaal worden karakters met een code beneden 20H door de output-verwerkers geïnterpreteerd als controlecodes. Een karaktercode in dit gebied kan als een schrijfbaar karakter behandeld worden door het te doen voortgaan van een controlecode 01H (grafische header) en 40H bij zijn code op te tellen. Om te vermijden dat karaktercode 0DH als een "CR"-code wordt geïnterpreteerd, moet het naar de output-verwerkers gestuurd worden als 01H, 4DH. Deze routine wordt gebruikt door de outputverwerkers, zoals de standaardroutine CHPUT, om naar zulke groepen karaktercodes te zoeken.

Is het karakter een grafische header, dan wordt GRPHED op 1 gezet en stopt de routine, zoniet wordt GRPHED 0. Licht de code

51

van het karakter buiten het bereik 40H tot 5FH, dan gebeurt er niets mee. Ligt het binnen dat bereik, en grafische header (waarmee gesignaleerd wordt dat er voor een grafische header kwam) dan wordt het omgezet door er 40H van af te trekken.

adres 098CH
 naam CHPUI
 in A - karakter
 uit n.v.t.
 wijzigd EI

Deze standaardroutine zet een karakter op het scherm in 40x24 of 32x24 tekstmode. Eerst wordt SCRHD bekeken en indien de Video Display Processor in grafische- of veelkleurmode staat, stopt de routine zonder iets te doen. Is dit niet zo dan wordt de cursor weggelaald (0A2EH), het karakter gedecodeerd (09DDFH) en de cursor teruggezet (09E1H), temeer wordt de kolompositie van de cursor in IYPOS gezet, voor gebruik door het "PRINT"-statement, en de routine stopt.

adres 09DFH

Deze routine wordt door de standaardroutine CHPUI gebruikt om een karakter te decoderen, en dan het nodige te doen. Eerst wordt de standaardroutine CNMCHR aangeroepen om te kijken of het een grafisch karakter is. Is het karakter een grafische header (011H), dan stopt de routine zonder meer. Is het karakter een omgezet grafisch karakter, dan wordt het geadapteerde waarde in ESCNT gekeken of voordien een ESC (1BH)-code werd ontvangen. Is dit zo, dan wordt de controle overgedragen aan de "ESC"-verwerker (09BFH). Was dat niet het geval, dan wordt gecontroleerd of het karakter een code heeft onder 20H. Zo ja, wordt verder gegaan met de controlecode-verwerker (0914H). Is dat niet zo, dan wordt bekeken of het om een "DEL" (7FH)-code gaat. Zo ja, wordt verder gegaan in de wis-routine (0A2EH).

In de veronderstelling dat het karakter schrijfbaar is, worden de coördinaten van de cursor opgehaald uit CSRY en CSRX, en in registerpaar HL gezet (H - kolom, L - rij). Die coördinaten worden vervolgens omgezet naar een adres in de Namentabel van de Video Display Processor en het karakter wordt daar gezet (0A2EH). De positie van de cursor kolom wordt opgehoogd (0A44H) en, indien daardoor de uiterst rechtse kolom niet overschreden werd, stopt de routine. Was dat wel het geval, dan wordt de overeenkomstige plaats in LINIB op nul gezet, om aan te duiden dat die logische regel langer is dan een schermregel; het kolom-nummer wordt 1 gemaakt en er wordt een "LF" uitgevoerd (aan regel lager op het scherm).

adres 099BH

Deze routine voert de lijnfeed uit voor de controlecode-verwerker van de standaardroutine CHPUI. De cursor-rij wordt opgehoogd (0A51H) en indien die de onderste rij niet overschrijft, stopt de routine. Zoniet, wordt het scherm naar boven gescreld en de jaagte regel gewist (0A6BH).

adres 0914H

Hier worden de controlecodes verwerkt voor de standaardroutine CHPUI. De tabel vanaf adres 092FH wordt doorzocht tot de karaktercode wordt gevonden, en de controle wordt overgedragen aan de routine op het bijbehorende adres.

adres 092FH TABEL

Hier begint een tabel met controlecodes, met het bijbehorende adres, dat door de standaardroutine CHPUI herkend wordt:

CODE NAAR FUNCTIE

07H 113H BELL, doe "beep"
 08H 0A4CH BS, cursor naar links
 09H 0A71H TAB, cursor naar volgende TAB-positie
 0AH 090BH LF, cursor een regel lager
 0BH 0A7FH HOME, cursor naar 0,0
 0CH 077EH FORMFEED, wis scherm en cursor naar 0,0
 0DH 0A61H CR, cursor naar uiterst linkse kolom
 1BH 098BH ESC, begin van een "ESCAPE"-groep"
 1CH 0A5BH RIGHT, cursor naar rechts
 1DH 0A4CH LEFT, cursor naar links
 1EH 0A57H UP, cursor naar omhoog
 1FH 0A61H DOWN, cursor naar omlaag

adres 0953H TABEL

Op dit adres begint een tabel met "ESC"-controlecodes, met voor elke code een bijbehorend adres dat door de standaardroutine CHPUI herkend wordt.

CODE NAAR FUNCTIE

6AH 077EH ESC, "1" wis scherm en cursor op 0,0
 7BH 077EH ESC, "2" wis scherm en cursor op 0,0
 7BH 0A2EH ESC, "K" wis tot eind van de regel
 7BH 0805H ESC, "J" wis tot eind van het scherm
 6CH 0A2EH ESC, "I" wis regel
 4CH 0A84H ESC, "L" voeg regel tussen
 4DH 0A85H ESC, "M" wis regel
 5BH 0986H ESC, "Y" vul coördinaten van cursor in
 5BH 0A57H ESC, "A" cursor omhoog
 42H 0A61H ESC, "B" cursor omlaag
 43H 0A44H ESC, "C" cursor naar rechts
 44H 0A55H ESC, "D" cursor naar links
 4BH 0A7FH ESC, "H" cursor naar 0,0
 7BH 0960H ESC, "X" verander cursor
 79H 0983H ESC, "V" verander cursor

adres 0980H

Deze routine voert de "ESC "X"-bewerking uit voor de controlecode-verwerker van de standaardroutine CHPUI. In ESCCNT wordt 01H gezet, om aan te geven dat het volgende karakter een parameter is.

adres 0983H

Deze routine voert de "ESC" "u"-bewerking uit voor de controlecode-verwerker van de standaardroutine CHPUI. In ESCCNT wordt 02H gezet, om aan te geven dat het volgende karakter een parameter is.

adres 0986H

Deze routine voert de "ESC" "v"-bewerking uit voor de controlecode-verwerker van de standaardroutine CHPUI. In ESCCNT wordt 04H gezet, om aan te geven dat het volgende karakter een parameter is.

adres 0989H

Deze routine voert de "ESC"-bewerking uit voor de controlecode-verwerker van de standaardroutine CHPUI. In ESCCNT wordt FFH gezet, om aan te geven dat het volgende karakter het tweede controle-karakter is.

adres 099FH

Deze routine verwerkt de "ESC"-groepen voor de standaardroutine CHPUI. Bevat ESCCNT FFH, dan is het behandelde karakter het tweede controlekarakter, en wordt de controle overgedragen aan de controlecode-verwerker (0919H), die de "ESC"-codetabel op 0953H doorzoekt.

Bevat ESCCNT 01H dat is het huidige karakter de enige parameter van de "ESC", "x"-groep. Is de parameter "y" (34H) dan wordt 00H in CSYLE gezet, wat een dichtvolgende cursor geeft. Als de parameter "s" (35H) is, wordt 00H in CSRSW gezet, wat de cursor onzichtbaar maakt.

Indien ESCCNT 02H bevat, is het huidige karakter de enige parameter in de "ESC", "v"-groep. Als die parameter "y" (34H) is, wordt in CSYLE 01H gezet, waardoor de cursor een half blokje wordt. Is de parameter "s" (35H) dat wordt in CSRSW 01H gezet, waardoor de cursor normaal zichtbaar wordt.

Bevat ESCCNT 04H dat is het huidige karakter de eerste parameter van de "ESC", "y"-groep, en is dit het lijn-coördinaat. Er wordt 1FH van afgetrokken en het resultaat wordt in CSRY gezet. ESCCNT wordt 03H.

Bevat ESCCNT 03H, dan is het huidige karakter de tweede parameter van de "ESC", "y"-groep, en is dit het kolom-coördinaat. Er wordt 1FH van afgetrokken en het resultaat wordt in CSRX gezet.

adres 09DAH

Deze routine wordt, bijvoorbeeld door de standaardroutine CHGET, gebruikt om de cursor op het scherm te zetten, terwijl hij normaal niet zichtbaar is. Als CSRSW verschilt van nul, stopt de routine zonder meer. Zoniet, wordt de cursor op het scherm gezet (09E6H).

adres 09E1H

Deze routine wordt, bijvoorbeeld door de standaardroutine CHPUI, gebruikt om de cursor op het scherm te zetten, terwijl hij normaal zichtbaar is. Als CSRSW nul is, stopt de routine zonder meer. SCRPOD wordt bekken en indien de Video Display Processor in grafische of vaalkleurmodus staat, stopt de routine zonder meer. Zoniet, worden de cursor-coördinaten omgerekend naar een adres in de Namentabel van de Video Display Processor, het karakter op die plaats wordt uitgelezen (0BDBH) en in CURSAU gezet.

Het 8-bytes pixelpatroon van het karakter wordt uit de Karaktertabel van de Video Display Processor gelezen, in de LINAUX-buifer (0B45H). Het pixelpatroon wordt geïnverteerd, van de acht bytes indien CSYLE een blokvolgende cursor aangeeft, of enkel de onderste vier indien CSYLE een halve cursor specificiert. Dan wordt het pixelpatroon teruggezet op de plaats van karaktercode 255 in de Karaktertabel van de Video Display Processor (0BBEH). Dan wordt karaktercode 255 op de plaats in de Namentabel gezet, waar momenteel de cursor staat (0BE6H) en de routine stopt.

Deze manier om de cursor op het scherm te zetten, door gebruik te maken van karaktercode 255, kan, onder bepaalde omstandigheden, eigenaardige resultaten opleveren. Een demonstratie daarvan kunnen we zien door de BASIC regel : "FOR N=1 TO 100: PRINT CHR\$(255);: NEXT" in te voeren, en dan op de "cursor omhoog"-toets te drukken.

adres 0A27H

Deze routine wordt, bijvoorbeeld door de standaardroutine CHGET, gebruikt om de cursor van het scherm te halen terwijl hij normaal onzichtbaar is. Bevat CSRSR niet nul, dan stopt de routine zonder meer. Zoniet, wordt de cursor van het scherm gehaald (0A33H).

adres 0A2EH

Deze routine wordt, bijvoorbeeld door de standaardroutine CHPUI, gebruikt om de cursor van het scherm te halen terwijl hij normaal zichtbaar is. Bevat CSRSW nul, dan stopt de routine zonder meer. SCRPOD wordt gecontroleerd en, wanneer het scherm in grafische- of vaalkleurmodus staat, stopt de routine zonder meer. Zoniet, worden de cursor-coördinaten omgerekend naar een adres in de Namentabel van de Video Display Processor, en het karakter dat in CURSAU staat wordt op die plaats gezet (0B66H).

adres 0A44H

Deze routine voert de "ESC", "c"-bewerking uit voor de controlecode-verwerker van de standaardroutine CHPUI. Indien het kolom-coördinaat van de cursor de meest rechtse kolom aangeeft (dat staat in LINAUX), dan stopt de routine zonder meer. Zoniet, wordt het kolom-coördinaat opgehoogd, en CSRX aangepast.

adres 0A6CH

Deze routine voert de "BS"/LINKS-bewerking uit voor de controlcode-verwerker van de standaardroutine CHPUI. Het kolom-codrdinaat van de cursor wordt verlaagd en CSRX aangepast. Wanneer het kolom-codrdinaat lager werd dan de uiterst linkse positie, wordt hij op de uiterst rechtse positie gezet (uit LINLEN) en wordt een "DHK00G"-bewerking uitgevoerd.

adres 0A65K

Deze routine voert de "ESC", "D"-bewerking uit voor de controlcode-verwerker van de standaardroutine CHPUI. Staat de cursor al op de meest linkse positie, dan stopt de routine zonder meer. Zoniet, wordt het kolom-codrdinaat verlaagd en CSRX aangepast.

adres 0A67H

Deze routine voert de "ESC", "A"/"DHK00G"-bewerking uit voor de controlcode-verwerker van de standaardroutine CHPUI. Indien de cursor al op de bovenste rij staat, stopt de routine zonder meer. Zoniet, wordt het rij-codrdinaat verlaagd en CSRY aangepast.

adres 0A6BH

Deze routine voert de "RECHTS"-bewerking uit voor de controlcode-verwerker van de standaardroutine CHPUI. Het kolom-codrdinaat van de cursor wordt opgehoogd en CSRX aangepast. Gaat het codrdinaat verder dan de uiterst rechtse positie, die in LINLEN staat, dan wordt hij op de meest linkse positie gezet (OIH) en wordt er een "DHLAAG"-bewerking uitgevoerd.

adres 0A61H

Deze routine voert de "ESC", "B"/"DHLAAG"-bewerking uit voor de controlcode-verwerker van de standaardroutine CHPUI. Staat de cursor al op de onderste rij (aangegeven door CIRONT en DNSDFG (OCC3CH)), dan stopt de routine zonder meer. Zoniet, wordt het rij-codrdinaat opgehoogd en CSRY aangepast.

adres 0A71H

Deze routine voert de "IAB"-bewerking uit voor de controlcode-verwerker van de standaardroutine CHPUI. Er worden ASCII-spaties naar het scherm gestuurd (O80FH) tot CSRX een veldvuld van 8 plus 1 is (8105-kolommen 1, 9, 17, 25, 33).

adres 0A7FH

Deze routine voert de "ESC", "H"/"K0HE"-bewerking uit voor de controlcode-verwerker van de standaardroutine CHPUI. CSRX en CSRY worden gewoon op 1 en 1 gezet. Het codrdinaatstelsel van de cursor, zoals gehanteerd door het ROM 8105 is functioneel gelijk aan het stelsel dat de BASIC interpreter gebruikt, maar nummer de rijen op het scherm van 1 tot 24 en de kolommen van 1 tot 32 of 40.

adres 0A81H

Deze routine voert de "CR"-bewerking uit voor de controlcode-verwerker van de standaardroutine CHPUI. CSRX wordt gewoon op 1 gezet.

adres 0A85H

Deze routine voert de "ESC", "H"-bewerking uit voor de controlcode-verwerker van de standaardroutine CHPUI. Eerst wordt een "CR" uitgevoerd om het kolom-codrdinaat van de cursor op de uiterst linkse positie te zetten. Dan wordt het aantal regels vanaf de huidige regel tot de onderkant van het scherm bepaald. Is dat aantal nul, dan wordt gewoon de huidige regel gewist (OACEN). Het aantal regels wordt eerst gebruikt om het betreffende stuk van LINTB 1 byte verder te scrollen, en dan om het overeenkomstige stuk van het scherm een regel tegelijk omhoog te scrollen. Te beginnen met de regel onder de huidige regel, wordt elke regel gecopieerd vanuit de Namentabel van de Video Display Processor naar de LINTB-buffer (O8AH), en dan naar de laagste regel op het scherm gewist. (OAECH).

adres 0A84H

Deze routine voert de "ESC", "L"-bewerking uit voor de controlcode-verwerker van de standaardroutine CHPUI. Eerst wordt een "CR" uitgevoerd, om het kolom-codrdinaat van de cursor op de uiterst linkse positie te zetten. Dan wordt het aantal regels vanaf de huidige stand tot de onderkant van het scherm bepaald. Als dat aantal nul is, wordt gewoon de huidige regel gewist (OAECH). Eerst wordt het aantal regels gebruikt om het overeenkomstige stuk van LINTB naar beneden te scrollen met 1 byte. Dan wordt het gebruikt om het betreffende schermdeel een regel tegelijk naar beneden te scrollen. Te beginnen bij de voorlaatste regel op het scherm, wordt elke regel gecopieerd vanuit de Namentabel van de Video Display Processor naar de LINTB-buffer (O8AH), en dan naar de Namentabel, maar een regel lager (O83CH). Tenslotte wordt de huidige regel gewist (OAECH).

adres 0A83H

Deze routine voert de "DELETE"-bewerking uit voor de controlcode-verwerker van de standaardroutine CHPUI. Eerst wordt een "LINKS"-bewerking uitgevoerd. Kan dit niet, omdat de cursor al op de uiterste positie staat, dan stopt de routine zonder meer. In het andere geval wordt een spatie geschreven in de Namentabel van de Video Display Processor, op de plaats van de cursor (O8E6H).

adres 0A8CH

Deze routine voert de "ESC", "I"-bewerking uit voor de controlcode-verwerker van de standaardroutine CHPUI. Het kolom-codrdinaat van de cursor wordt op OIH gezet, en de routine gaat verder met de "ESC", "K"-routine.

adres OAEEM

Deze routine voert de "ESC", "K"-bewerking uit voor de controlecode-verwerker van de standaardroutine uit voor de overeenkomstige plaats in LINTB wordt eerst niet-nul gemaakt, om aan te geven dat de logische regel niet uitgebreid werd (0029H). De cursor-coördinaten worden omgezet in een adres in de Namentabel van de Video Display Processor (0BFEH) en de Video Shift. Dan worden spaties rechtstreeks naar de Dataport van de Video Display Processor geschreven tot de uiterst rechts kolom (aangegeven door LINLEN), bereikt wordt.

adres OBOSH

Deze routine voert de "ESC", "J"-bewerking uit voor de controlecode-verwerker van de standaardroutine CPUIT. De beginnen met de huidige regel tot het einde van het scherm, te worden openvolgende "ESC", "K"-bewerkingen uitgevoerd.

adres OB15H

naam ERAFNK
in n.v.t.
uit n.v.t.
wijzigt AF,DE,EI

Deze standaardroutine zorgt ervoor dat de tekst van de functie-toetsen niet meer op het scherm komt. Eerst wordt nul in CNDSPG gezet en indien de Video Display Processor in grafische of veelkleuren-mode staat, stopt de routine zonder meer. Staat de Video Display Processor in 40x24 tekst mode of 32x24 tekst mode, dan wordt de laatste schermegeel gewist (0BECN).

adres OB26H

naam FNKSB
in n.v.t.
uit n.v.t.
wijzigt AF,BC,DE,EI

Deze standaardroutine zorgt ervoor dat de tekst van de functie-toetsen op het scherm staat, als dat toegelaten is. Als CNDSPG nul bevat, stopt de routine zonder meer. Zoniet, gaat de routine verder met de standaardroutine DSPFNK.

adres OB2BH

naam DSPFNK
in n.v.t.
uit n.v.t.
wijzigt AF,BC,DE,EI

Deze standaardroutine maakt de teksten van de functie-toetsen op het scherm zichtbaar. CNDSPG wordt op 255 gezet. Indien de Video Display Processor in grafische- of veelkleuren-mode staat, stopt de routine zonder meer. Zoniet, wordt het rij-coördinaat van de cursor bekeken en indien de cursor op de laatste regel staat wordt een Linefeed code (0AH) naar de standaardroutine DUTDO

gestuurd, om het scherm omhoog te scrollen. Dan wordt register HL ingevuld met het startadres van de teksten die bij de functie-toetsen horen, in gewone stand ofwel in de SHIFT-stand, wanneer de SHIFT-toets is ingedrukt van LINLEN wordt vier afgetrokken, om minimum 1 spatie tussen de diverse teksten te laten, en gedeeld door vijf om de ruimte te bepalen die elke string zal innemen. Vervolgens worden de openvolgende karakters uit de string gelezen, gecontroleerd op grafische handlers via de standaardroutine CNUCHR en in de LINWRK-buifer geplaatst tot er ofwel geen karakters meer in de string voorkomen, ofwel de beschikbare ruimte in de zone gewu- is. Zijn de vijf strings volledig, dan wordt de inhoud van de LINWRK-buifer naar de laatste rij in de Namentabel van de Video Display Processor geschreven (0B2CH).

adres OB2CH

Deze routine wordt gebruikt door de andere routines die verba- houden met het op het scherm zetten van de tekst van de functie-toetsen. De inhoud van register A wordt in CNDSPG gezet. SCRMOD gecontroleerd, en de routine keert terug met vlag 'C' indien het scherm in grafische of veelkleurenmode staat.

adres OB45H

Deze routine copieert acht bytes van URAM naar de LINWRK-buifer. Het adres in URAM staat in registerpaar HL.

adres OB4AH

Deze routine copieert een volledige regel karakters, waarvan de lengte bepaald wordt door LINLEN, vanuit URAM naar de LINWRK-buifer. Het lijn-coördinaat van de cursor staat in register L.

adres OB4BH

Deze routine copieert acht bytes van de LINWRK-buifer naar URAM. Het adres in URAM staat in registerpaar HL.

adres OB4CH

Deze routine copieert een volledige regel karakters, waarvan de lengte door LINLEN bepaald wordt, van de LINWRK-buifer naar URAM. Het lijn-coördinaat van de cursor staat in register L.

adres OB4DH

Deze routine leest een byte uit URAM in register C. Het kolomcoördinaat staat in register H, het lijn-coördinaat in register L.

adres OB4EH

Deze routine bepaalt welk adres in de Namentabel van de Video Display Processor hoort bij een stel scherf-coördinaten. Het kolom-coördinaat staat in register H, en het lijn-coördinaat in register L. Het adres staat, op het einde van de routine, in registerpaar HL.

adres OB4FH

Het lijn-coördinaat wordt eerst met 32 of 40 vermenigvuldigd.

afhankelijk van de scherm-mode, en bij de kolom-coördinaat opgeteld. Dit resultaat wordt dan opgeteld bij het basisadres van de Nemen tabel van de Video Display Processor (uit NMBAS), om een eerste adres te verkrijgen.

Omwillen van de veranderlijke schermbreedte (in LINLEN), moet dit adres aangepast worden zodat het actieve gebied nog min of meer in het midden van het scherm staat. Het verschil tussen het "echte" aantal karakters per regel, 32 of 40, en de op dat moment gekozen schermbreedte, wordt gehalveerd en dan opgerond. Zo verkrijgen we de aanpassing voor de linkse kolom. Bij een Engelse machine, waar in FOXET tekst mode, 37 karakters gebruikt worden, blijven zo twee karakter-posities links en één rechts ongebruikt. Het statement : PRINT (41-WID)/2, waarbij WID de schermbreedte is, levert de aanpassing op voor de linker kolom in FOXET tekst mode.

Hieronder volgt een BASIC programma dat hetzelfde doet als deze routine.

```

10 CPR=40:NAM=BASE(0):WID=PEEK(&F3AE)
20 SCRD=PEEK(&HFC4F):IF SCRD=0 THEN 40
30 CRP=32:NAM=BASE(5):WID=PEEK(&HFB4F)
40 LH=(CPR+1-WID)/2
50 ADDR=NAM+(ROW-1)*CPR+(COL-1)*LH

```

Dit programma werd geschreven om te werken met de RIJ en KOLM-coördinaten van het ROM BIOS, waar de "HOME"-positie 1,1 is. Regel 50 kan vereenvoudigd worden, door de "-1"-factoren weg te halen, indien het coördinatenstelsel van de BASIC Interpreter gebruikt moet worden.

adres OC1DH

Deze routine berekent het adres van de beschrijving in LINTIB van een bepaalde rij. Het rij-coördinaat staat in register L, het adres staat op het einde van de routine in register DE.

adres OC29H

Deze routine zet het byte dat bij een bepaalde regel behoort in LINTIB, op een waarde verschillend van 0 wanneer ze op OC29H aangeropen wordt, en op 0 wanneer ze op OC2AH wordt aangeropen. Het rij-coördinaat staat in register L.

adres OC32H

Het deze routine wordt berekend hoeveel regels er op een scherm kunnen. Het aantal wordt in register A gezet. Normaal zal dit 24 zijn, indien de tekst van de functietoetsen niet op het scherm staat, en 23 indien die tekst er wel staat. De grootte van het scherm wordt bepaald door CRIONT, en kan vanuit BASIC gewijzigd worden, bijvoorbeeld met : POKE &HFB3B,14:SCREEN 0.

adres OC3CH

naam KEYINT
in n.v.t.
uit n.v.t.
wijzigt EI

Deze standaardroutine verwerkt de interrupts van de Z80. Die interrupts worden door de Video Display Processor elke 20 msec opgewekt (bij een Europese machine). Eerst wordt het statusregister van de Video Display Processor bekeken, en bit 7 gecontroleerd om zeker te zijn dat het om een interrupt gaat vanuit de Video Display Processor. Is dat niet zo, dan stopt de routine zonder meer. Zoniet, wordt de inhoud van het statusregister in STAFPL gezet, en bit 5 gecontroleerd om te zien of er sprake is van overlappen. Is de coincidentie-vlag gezet, dan wordt het overeenkomstige byte in IRPBL aangepast (OEFLX).

Vervolgens wordt INTCON (de "INTERVAL"-teller) verlaagd. Wordt die teller nul, dan wordt de bijbehorende plaats in IRPBL aangepast (OEFLX) en wordt de teller ingevuld met de inhoud van INTVAL.

JIFFY, de "TIME"-teller, wordt opgehoogd. Wanneer die zijn maximum bereikt, wordt die gewoon weer 0.

MUSICF wordt onderzocht, om te zien of een van de drie muziek-rijen die door het PLAY-statement worden aangemaakt actief is. Voor elke actieve rij wordt de afhandelingroutine aangeropen (113BH) om het volgende pakket naar de PSG te schrijven.

Vervolgens wordt SCNDNT verlaagd met 1, om te zien of de joystick en het toetsenbord waken worden afgestast. Is dat niet nodig, dan stopt de routine zonder meer. Die teller zorgt ervoor dat de werkzaamheid verhoogd wordt, en dat contactdenderproblemen zo weinig mogelijk voorkomen, door maar om de drie interrupts een afcasting uit te voeren. In de veronderstelling dat er een afcasting moet gebeuren, wordt joystick-aanluiting 1 actief gemaakt en de twee vuurknop-bits uitgelezen (12COH), gevolgd door dezelfde twee bits van joystick 2 (12C0H), spatioets op rij 8 van het toetsenbord (1226H). Deze vijf inputs, die allemaal te maken hebben met het "STRIG"-statement, worden tot één byte verwerkt, waarin een bit 0 betekent : ingedrukt, en een bit 1 : niet ingedrukt.

7	6	5	4	3	2	1	0
Joy 2	Joy 2	Joy 2	Joy 1	Joy 1	Joy 1	0	0
trg.B	trg.A	trg.A	trg.B	trg.A	trg.A	space	space

Fig. 35 : "STRIG"-inputs

Deze uitlezing wordt vergeleken met de vorige uitlezing, die in IRPBL staat, om een actief overgangsbite te maken, en IRPFLB wordt ingevuld met de laatste uitlezing. Normaal is dit overgangsbite 0, maar het bevat een "1"-bit op elke plaats waar een overgang van niet-ingedrukt naar ingedrukt heeft plaatsgehad. Dit byte wordt bit na bit onderzocht, en de overeenkomstige plaats in IRPBL wordt aangepast voor elk actief apparaat (OEFLX).

Vervolgens wordt de complete matrix van het toetsenbord

afhankelijk van de scherm-mode, en bij de kolom-coördinaat opgeteld. Dit resultaat wordt dan opgeteld bij het basisadres van de Nemen tabel van de Video Display Processor (uit NMBAS), om een eerste adres te verkrijgen.

Omwillen van de veranderlijke schermbreedte (in LINLEN), moet dit adres aangepast worden zodat het actieve gebied nog min of meer in het midden van het scherm staat. Het verschil tussen het "echte" aantal karakters per regel, 32 of 40, en de op dat moment gekozen schermbreedte, wordt gehalveerd en dan opgerond. Zo verkrijgen we de aanpassing voor de linkse kolom. Bij een Engelse machine, waar in FOXET tekst mode, 37 karakters gebruikt worden, blijven zo twee karakter-posities links en één rechts ongebruikt. Het statement : PRINT (41-WID)/2, waarbij WID de schermbreedte is, levert de aanpassing op voor de linker kolom in FOXET tekst mode.

Hieronder volgt een BASIC programma dat hetzelfde doet als deze routine.

```

10 CPR=40:NAM=BASE(0):WID=PEEK(&F3AE)
20 SCRD=PEEK(&HFC4F):IF SCRD=0 THEN 40
30 CRP=32:NAM=BASE(5):WID=PEEK(&HFB4F)
40 LH=(CPR+1-WID)/2
50 ADDR=NAM+(ROW-1)*CPR+(COL-1)*LH

```

Dit programma werd geschreven om te werken met de RIJ en KOLM-coördinaten van het ROM BIOS, waar de "HOME"-positie 1,1 is. Regel 50 kan vereenvoudigd worden, door de "-1"-factoren weg te halen, indien het coördinatenstelsel van de BASIC Interpreter gebruikt moet worden.

adres OC1DH

Deze routine berekent het adres van de beschrijving in LINTIB van een bepaalde rij. Het rij-coördinaat staat in register L, het adres staat op het einde van de routine in register DE.

adres OC29H

Deze routine zet het byte dat bij een bepaalde regel behoort in LINTIB, op een waarde verschillend van 0 wanneer ze op OC29H aangeropen wordt, en op 0 wanneer ze op OC2AH wordt aangeropen. Het rij-coördinaat staat in register L.

adres OC32H

Het deze routine wordt berekend hoeveel regels er op een scherm kunnen. Het aantal wordt in register A gezet. Normaal zal dit 24 zijn, indien de tekst van de functietoetsen niet op het scherm staat, en 23 indien die tekst er wel staat. De grootte van het scherm wordt bepaald door CRIONT, en kan vanuit BASIC gewijzigd worden, bijvoorbeeld met : POKE &HFB3B,14:SCREEN 0.

adres OC3CH

naam KEYINT
in n.v.t.
uit n.v.t.
wijzigt EI

Deze standaardroutine verwerkt de interrupts van de Z80. Die interrupts worden door de Video Display Processor elke 20 msec opgewekt (bij een Europese machine). Eerst wordt het statusregister van de Video Display Processor bekeken, en bit 7 gecontroleerd om zeker te zijn dat het om een interrupt gaat vanuit de Video Display Processor. Is dat niet zo, dan stopt de routine zonder meer. Zoniet, wordt de inhoud van het statusregister in STAFPL gezet, en bit 5 gecontroleerd om te zien of er sprake is van overlappen. Is de coincidentie-vlag gezet, dan wordt het overeenkomstige byte in IRPBL aangepast (OEFLX).

Vervolgens wordt INTCON (de "INTERVAL"-teller) verlaagd. Wordt die teller nul, dan wordt de bijbehorende plaats in IRPBL aangepast (OEFLX) en wordt de teller ingevuld met de inhoud van INTVAL.

JIFFY, de "TIME"-teller, wordt opgehoogd. Wanneer die zijn maximum bereikt, wordt die gewoon weer 0.

MUSICF wordt onderzocht, om te zien of een van de drie muziek-rijen die door het PLAY-statement worden aangemaakt actief is. Voor elke actieve rij wordt de afhandelingroutine aangeropen (113BH) om het volgende pakket naar de PSG te schrijven.

Vervolgens wordt SCNDNT verlaagd met 1, om te zien of de joystick en het toetsenbord waken worden afgestast. Is dat niet nodig, dan stopt de routine zonder meer. Die teller zorgt ervoor dat de werkzaamheid verhoogd wordt, en dat contactdenderproblemen zo weinig mogelijk voorkomen, door maar om de drie interrupts een afcasting uit te voeren. In de veronderstelling dat er een afcasting moet gebeuren, wordt joystick-aanluiting 1 actief gemaakt en de twee vuurknop-bits uitgelezen (12COH), gevolgd door dezelfde twee bits van joystick 2 (12C0H), spatioets op rij 8 van het toetsenbord (1226H). Deze vijf inputs, die allemaal te maken hebben met het "STRIG"-statement, worden tot één byte verwerkt, waarin een bit 0 betekent : ingedrukt, en een bit 1 : niet ingedrukt.

7	6	5	4	3	2	1	0
Joy 2	Joy 2	Joy 2	Joy 1	Joy 1	Joy 1	0	0
trg.B	trg.A	trg.A	trg.B	trg.A	trg.A	space	space

Fig. 35 : "STRIG"-inputs

Deze uitlezing wordt vergeleken met de vorige uitlezing, die in IRPBL staat, om een actief overgangbyte te maken, en IRPFLB wordt ingevuld met de laatste uitlezing. Normaal is dit overgangbyte 0, maar het bevat een "1"-bit op elke plaats waar een overgang van niet-ingedrukt naar ingedrukt heeft plaatsgehadt. Dit byte wordt bit na bit onderzocht, en de overeenkomstige plaats in IRPBL wordt aangepast voor elk actief apparaat (OEFLX).

Vervolgens wordt de complete matrix van het toetsenbord

onderzocht, naar ingedrukte toetsen. Elke druk op een toets wordt vertaald in een code, en in KEYBUF gezet (0012H). Bijlt, aan het einde van deze bemerking, dat KEYBUF leeg is, dan wordt REPCNT verlaagd met 1 om te zien of de auto-repeat periode voorbij is. Zoniet, stopt de routine. Was de periode wel herhaling van toetsen (SO ms), de OLDKEY toetsenbord-cabai wordt gereset en het toetsbord wordt nog eens afgeleest (ODKEY). Wanneer tijdens deze aflezing een toets wordt ingedrukt, geeft hij telkens weer overgang (zits hogere). Merk op, dat een toets alleen zal herhalen indien een toetsenbordprogramma KEYBUF leeg houdt, door karakters uit de buffer te lezen. Nu eindigt de Interrupt-routine.

adres 0012H

Dese routine verzorgt een volledige aflezing van de toetsenmatrix voor de interrupt-routine. Elk van de elf rijen wordt ingelezen via de PPI, en in opklimende volgorde in NEWKEY gezet. ENSTOP wordt dan gecontroleerd om te zien of welke start mogelijk zijn. Wanneer de inhoud daarvan niet nul is, en de toetsen : "CODE", "GRPH", "CTRL" en "SHIFT" tegelijk worden ingedrukt, wordt de controle aan de BASIC Interpreter overgedragen (4080H) via de standaardroutine CALBAS. Deze voorzetting is nuttig, omdat ze toelaat om zelfs een machinetaal-programma te beëindigen, zolang de interrupt worden verwerkt.

De inhoud van NEWKEY wordt vergeleken met de inhoud van OLDKEY, die het resultaat van de vorige aflezing bevat. Wordt een verandering opgemerkt, dan wordt REPCNT geladen met de beginwaarde voor de wachttijd tussen het afdrucken van twee toetsen bij auto-repeat (780 msec). De uitzetting van elke rij in NEWKEY wordt dan vergeleken met de overeenkomstige in OLDKEY, waarmee een actief overgangsbite wordt aangemaakt, en OLDKEY wordt aan de laatste uitzetting aangepast. Dit overgangsbite is normaal 0, maar bevat een "1"-bit op elke positie waar een overgang van niet-ingedrukte toets naar ingedrukte toets plaatsvindt. Bevat de rij zo een overgang, dan wordt die gereset en als een toets-code in KEYBUF gezet (0080H). De elf rijen afgewerkt, dan controleert de routine nog of er in KEYBUF karakters staan (door GETPNI van PUIPNI af te trekken), en stopt dan.

adres 0080H

naam CHSNGS
in N.V.T.
uit vlag NZ indien in KEYBUF karakters staan
wijzigt AF, EI

Dese standaardroutine controleert of KEYBUF karakters bevat. Staat het scherm in grafisch- of tekstmodus, dan wordt GETPNI van PUIPNI afgetrokken (0082H) en de routine stopt hier. Staat het scherm in de XORCH of XORCHY tekst mode, dan wordt eveneens de toestand van de SHIFT-toets bekeken en indien die sinds de laatste keer veranderd is, wordt via de standaardroutine DSPFNK de tekst van de andere via de functiecode op het scherm afgedrukt.

adres 0080H

Dese routine zet elk "1"-bit in een overgangsbite om in een toets-code. Eerst wordt het bit omgezet in een toets-nummer, bepaald door zijn positie in de matrix van het toetsenbord :

7	6	5	4	3	2	1	0	
(07H)	(06H)	(05H)	(04H)	(03H)	(02H)	(01H)	(00H)	RIJ 0
(0FH)	(0EH)	(0DH)	(0CH)	(0BH)	(0AH)	(09H)	(08H)	RIJ 1
B	A	E	/					RIJ 2
(17H)	(16H)	(15H)	(14H)	(13H)	(12H)	(11H)	(10H)	
J	I	H	G	F	E	D	C	RIJ 3
(1FH)	(1EH)	(1DH)	(1CH)	(1BH)	(1AH)	(19H)	(18H)	
R	Q	P	O	N	M	L	K	RIJ 4
(27H)	(26H)	(25H)	(24H)	(23H)	(22H)	(21H)	(20H)	
Z	Y	X	W	V	U	T	S	RIJ 5
(2FH)	(2EH)	(2DH)	(2CH)	(2BH)	(2AH)	(29H)	(28H)	
F3	F2	F1	CODE	CAP	GRAPH	CTRL	SHIFT	RIJ 6
(37H)	(36H)	(35H)	(34H)	(33H)	(32H)	(31H)	(30H)	
CR	SELE	BS	STOP	TAB	ESC	FS	F4	RIJ 7
(3FH)	(3EH)	(3DH)	(3CH)	(3BH)	(3AH)	(39H)	(38H)	
RECHTS	ONTLAAG	OMHOOG	LINKS	DEL	INS	NOTE	SPACE	RIJ 8
(47H)	(46H)	(45H)	(44H)	(43H)	(42H)	(41H)	(40H)	
4	3	2	1	0	/	7	*	RIJ 9
(4FH)	(4EH)	(4DH)	(4CH)	(4BH)	(4AH)	(49H)	(48H)	
7	6	5	4	3	2	1	0	RIJ 10
(57H)	(56H)	(55H)	(54H)	(53H)	(52H)	(51H)	(50H)	

FIG. 36 : Nummers van de toetsen

Het nummer van de toets wordt vervolgens in een toets-code omgezet die in KEYBUF geplaatst wordt (1021H). Wanneer alle acht bits verwerkt zijn, stopt de routine.

adres 0DASH TABEL

Op dit adres start een tabel met de toets-nummers 00H tot 2FH, voor verschillende combinaties van de controletoetsen. Een nul in die tabel betekent dat indrukken van die toetscombinatie, geen toets-code oplevert.

adres	0DASH	TABEL
NORMAL	37H 36H 5BH 34H 33H 32H 31H 30H	R1J 0
	3BH 5DH 5BH 5CH 3DH 2DH 39H 38H	R1J 1
	62H 61H 9CH 2FH 2EH 2CH 60H 27H	R1J 2
	6AH 69H 68H 67H 66H 65H 64H 63H	R1J 3
	72H 71H 70H 6FH 6EH 6DH 6CH 6BH	R1J 4
	7AH 79H 78H 77H 76H 75H 74H 73H	R1J 5
SHIFT	26H 5EH 25H 24H 23H 40H 21H 29H	R1J 0
	3AH 7DH 7BH 7CH 2BH 3FH 2BH 2AH	R1J 1
	42H 41H 9CH 3FH 3EH 3CH 7EH 2EH	R1J 2
	4AH 49H 48H 47H 46H 45H 44H 43H	R1J 3
	52H 51H 50H 4FH 4EH 4DH 4CH 4BH	R1J 4
	5AH 59H 58H 57H 56H 55H 54H 53H	R1J 5
GRAPH	FBH F4H BDH EFH BAH ABH ACH 09H	R1J 0
	06H 0DH 01H 1EH 1FH 17H 07H ECH	R1J 1
	11H C4H 9CH 1DH F2H F3H BBH 95H	R1J 2
	CBH DCH 13H 15H 14H CDH C7H BCH	R1J 3
	1BH DCH DBH CEH 1BH 0BH C9H DDH	R1J 4
	0FH 19H 1CH 1FH 1AH C0H 1EH DEH	R1J 5
SHIFT+GRAPH	00H F3H 00H 00H FCH F0H 00H 0AH	R1J 0
	04H 0EH 02H 16H F0H 1FH 00H 00H	R1J 1
	00H F2H 9CH F6H AFH AEH F7H 03H	R1J 2
	CAH DFH D6H 10H D4H CEH C1H FAH	R1J 3
	ABH CAH D7H C3H D3H 0CH C9H DEH	R1J 4
	FBH AAH F9H D0H D5H C5H 00H D1H	R1J 5
CODE	E1H E0H 9BH 9BH BEH D9H 9FH EBH	R1J 0
	B7H D4H EDH 9CH E2H E2H E2H E7H	R1J 1
	97H 84H 9CH A7H A6H B6H E5H B9H	R1J 2
	91H A1H B1H B1H 94H BCX BBH B0H	R1J 3
	93H B3H A3H A2H A4H E6H B5H B3H	R1J 4
	95H A0H 84H 8BH 8BH 82H 96H 89H	R1J 5
SHIFT+CODE	00H 00H 9DH 9CH BEH 9EH ADH DBH	R1J 0
	B6H E4H E6H 00H 00H 00H 00H E2H	R1J 1
	00H BEH 9CH ABH 00H BFH E4H BBH	R1J 2
	92H 00H B0H 94H 94H 00H 00H 00H	R1J 3
	00H 00H E3H 00H A5H 00H B4H B2H	R1J 4
	00H 00H 00H 00H 00H 90H 00H 00H	R1J 5
	7 6 5 4 3 2 1 0	KOLOM

adres 0ECSH

De controle wordt aan deze routine overgegeven, op het einde van de decodering van de functietoetsen (0F5H). De overeenkomstige plaats in FNKFLG wordt eerst gecontroleerd, om te zien of die bepaalde functietoets verbonden is met een "ON KEY GOSUB"-statement. Is dat zo, en geeft CURLIN tegelijk aan dat de BASIC interpreter een programme aan het verwerken is, dan wordt de overeenkomstige plaats in IRPBL aangepast (0EFIX) en de routine stopt hier.

Is de toets niet verbonden met een "ON KEY GOSUB"-statement, of is de interpreter niet met een programme bezig, dan wordt de tekst-string die bij die toets hoort, opgehaald. Het toetsnummer wordt met zachtjes vermenigvuldigd (elke string is 16 tekens lang) en het resultaat wordt opgeteld bij het startadres van de reeks strings in het werkgebeid. De opervolgende tekens uit die string worden in KEYBUF geplaatst (0F5H) tot het einde (een 0-byte).

adres 0EFIX

Deze routine dient om IRPBL aan te passen, wanneer een of ander apparaat een interrupt heeft gegenereerd tijdens een BASIC programma. Bij het begin bevat register R1 het adres in de tabel van het statusbyte dat bij het apparaat hoort. Eerst wordt bit 0 van het statusbyte gecontroleerd, en als het apparaat niet "AAN" is, stopt de routine zonder meer. Bit 2, de "Event"-vlag, wordt vervolgens bekeken. Is deze bit 1, dan stopt de routine; zoniet, wordt het bit gezet om aan te geven dat er een "Event" heeft plaatsgevonden. Bit 1, de "STOP"-vlag wordt nu bekeken. Wordt het apparaat onderbroken, dan gebeurt er niets meer. Is dat niet het geval, dan wordt OMSBR opgehoogd, waardoor de Verwerkings-lus van de Interpreter weet dat het "Event" nu afgehandeld dient te worden.

adres 0F06H

Dit stukje van de toets-decoder verwerkt enkel de "HOME"-toets. De toestand van de "SHIFT"-toets wordt bepaald via rij 6 van NEWKEY en al naar gelang het resultaat daarvan, wordt de toetscode voor "HOME" (0BH) of "CLS" (0CH) in KEYBUF gezet (0F5H).

adres 0F10H

Dit stuk van de toets-decoder verwerkt de toetsen met nummers tussen 30H en 57H, behalve "CAP", "F1" tot "FS", "STOP" en "HOME". Het nummer van de toets wordt gebruikt om de toetscode op te zoeken in de tabel op 1033H, en die code wordt in KEYBUF gezet (0F5H).

adres 0F1FH

Dit stuk van de toets-decoder verwerkt de "dode" toets, die op Europese machines voorkomt. Op Engelse machines geeft de toets in rij twee, kolom vijf, altijd de toetscode voor het Pond-teken (SCH), zoals te zien is in de tabel op adres 0DASH. Op Europese machines staat op dezelfde plaatsen de code 255. Dit is alleen

een vlag, om aan te geven dat de volgende toets, indien het een klikker is, gewijzigd dient te worden in een grafisch karakter met een accent.

De toestand van de "SHIFT" en "CODE"-toetsen wordt uitgelezen via rij 5 van NEWKEY en één van de volgende posities in KANAST: 1 - DDDD, 2 - DDDD+SHIFT, 3 - DDDD+CODE, 4 - DDDD+SHIFT+CODE.

adres 0F36H
Dit stukje van de toets-decoder verwerkt de "CAP"-toets. De huidige stand van CAPSI wordt omgekeerd, en de controle wordt overgedragen aan de standaardroutine CHGCRP.

adres 0F3DH
naam CHGCRP
in A - aan/uit-schakelaar
uit n.v.t.
wijzigt AF

} Dit 6 van FAA
LAMF uit: BIT 6 = 1
AAV: " " = 0

Deze standaardroutine zet de LED bij de "CAP"-toets aan of uit, al naargelang de inhoud van register A: 0 - aan, 1 - uit. Om de LED te schakelen wordt de hoekpoort van de PPI gebruikt, die een bit kan zetten en resetten. Omdat CAPSI niet wordt gewijzigd, heeft deze routine geen invloed op de karakters die door een druk op een toets worden geproduceerd: indien dit voordien hoofdletters waren, blijven dat hoofdletters, en kleine letters blijven kleine letters.

adres 0F46H
Dit stukje van de toets-decoder verwerkt de "STOP"-toets. De toestand van de "CTRL"-toets wordt bepaald via rij 6 van NEWKEY en de code voor "STOP" (04H) of voor "CTRL/STOP" (03H) wordt geproduceerd. Was het de code voor "CTRL/STOP" dan wordt die geproduceerd in INTFLG, waar de standaardroutine "ISCNT" ze later zal ophalen, en vervolgens in KEYBUF (0F55H). Was het de "STOP"-code, dan wordt ze ook in INTFLG gezet, maar niet in KEYBUF. Er wordt enkel een klik geproduceerd (0F54H). Dit betekent dat een toetsingsprogramma de "STOP"-toets niet via de ROM BIOS-routines kan afzetten.

adres 0F55H
Dit stuk van de toets-decoder zet een toetscode in KEYBUF en produceert een hoorbare klik. Eerst wordt het juiste adres in de buffer uit PUTINT gelezen, en op dit adres wordt de code gezet. Vervolgens wordt het adres opgehoogd (05BH). Indien dat adres doorloopt, en hetzelfde wordt als GETINT, dan stopt de routine hier: de buffer is vol. Is dat niet zo, dan wordt in PUTINT het nieuwe adres ingevuld.

CLIKS en CLIKFL worden beide bekeken, om te zien of er een klik geproduceerd moet worden. CLIKS is een algemene weg/weg niet-schakelaar, terwijl CLIKFL voorkomt dat er meerdere klikjes hoorbaar zijn bij het indrukken van een functie-toets. In het geval dat er een klik moet worden geproduceerd, wordt via de hoekpoort van de PPI de Key Click Output gezet, en na 50 msec wordt de controle overgegeven aan de standaardroutine CHG5ND.

adres 0F7AH
naam CHG5ND
in A - aan/uit-schakelaar
uit n.v.t.
wijzigt AF

Deze standaardroutine set of reset de Key Click Output via de hoekpoort van de PPI: 0 - reset, 1 - set. Deze audio-output is een wisselspanning gekoppeld, zodat aan de absolute polariteit niet al te zeer getild hoeft te worden.

adres 0F83H

Dit stuk van de toets-decoder verwerkt toetsen met nummers tussen 00H en 2FH. De toestand van de "SHIFT", "GRPH" en "CODE"-toetsen wordt bepaald via rij 5 van NEWKEY en gecombineerd met het nummer van de toets, zodat een zoek-adres in de tabel vanaf 0DASH verkregen wordt. Dan wordt de toets-code uit de tabel gelezen. Is die code nul, dan stopt de routine. Is de code FFH, dan wordt verdergegaan met de dode-toets-verwerker (0F1FH).

Ligt de code tussen 40H en 5FH of tussen 60H en 7FH, en de "CTRL"-toets werd ingedrukt, dan wordt de overeenkomstige toets-code in KEYBUF gezet (0F55H).

Ligt de code tussen 81H en 1FH, dan wordt eerst een grafische heeder-code (01H) in KEYBUF gezet (0F55H), gevolgd door de eigenlijke code, waarbij 40H werd opgeteld.

Ligt de code tussen 61H en 7BH, en CAPSI geeft aan dat de hoofdletters ingeschakeld zijn, dan wordt de code omgezet in een hoofdlettercode door er 20H van af te trekken. In de veronderstelling dat KANAST nu bevat (altijd, bij een Engelse machine), wordt de code in KEYBUF gezet en de routine stopt. Bij Europese machines, met een dode toets in plaats van een Pond-toets, kunnen de codes die met een klikker overeenkomen dan nog verder in grafische codes omgezet worden.

adres 0FC3H

Dit stuk van de toets-decoder verwerkt de vijf functie-toetsen. De toestand van "SHIFT" wordt onderzocht via rij 6 van NEWKEY, en er wordt vijf opgeteld bij het nummer van de toets, indien "SHIFT" ingedrukt is. Dan wordt de controle overgedragen naar 0E5H, waar verder verwerkt wordt.

adres 10E1H

Deze routine doorzoekt de tabel vanaf adres 1897H, om te bepalen bij welke groep toetsen het toets-nummer in register C hoort. Het overeenkomstige adres wordt uit de tabel gehaald en de controle wordt overgedragen aan dat deel van de toets-decoder. Merk op dat de eigenlijke tabel in feite midden in de standaardroutine OUTDD staat: dat is het gevolg van wijzigingen aan de Japanse ROM.

 adres 1033H

 In deze tabel staan de toets-codes van de toetsen met nummers 30H tot 57H, behalve de toetsen "CAP", "F1" tot "F5", "STOP" en "HOME". Indien in de tabel nul staat, wordt geen toetscode geproduceerd bij het indrukken van die toets.

00H 00H 00H 00H 00H 00H 00H 00H	RIJ 6
00H 10H 00H 00H 00H 10H 00H 00H	RIJ 7
10H 1FH 10H 10H 7FH 12H 00H 20H	RIJ 8
30H 30H 32H 31H 30H 00H 00H 00H	RIJ 9
20H 20H 20H 30H 30H 37H 30H 35H	RIJ 10

7 6 5 4 3 2 1 0 KOL0H

 adres 105BH

 Deze routine zet KANAST op nul en gaat door naar 10C2H.

 adres 1061H

 Op dit adres begint een tabel met de grafische karakters die op Europese machines de klinkers a,e,i,o,u vervangen.

 adres 10C2H

 Deze routine telt één op bij de pointer van de toetsenbord-buifer, GETPNT of PUTPNT, die in HL aangebrecht wordt. Als de pointer daarvoor hoger wordt dan het einde van de buifer, dan telt hij door naar het begin.

 adres 10CBH

 naam CHGET
 in n.v.t.
 uit A - karakter dat werd ingetypt
 wijzigt AF,EI

 adres 10F9H

 naam CKCNTC
 in n.v.t.
 uit n.v.t.
 wijzigt AF,EI

 Deze standaardroutine controleert of de "CTRL/STOP"- of de "STOP"-toets werd ingedrukt. De BASIC interpreter gebruikt ze tijdens het verwerken van statements die de processor druk bezig houden, zoals "WAIT" en "CHGET", om te zien of een programma beëindigd dient te worden. Registerpaar HL wordt nul gemaakt, en dan wordt de controle overgedragen aan de standaardroutine ISCNTC.

 adres 1102H

 naam WRIPSG
 in A - registernummer, E - databyte
 uit n.v.t.
 wijzigt EI

 adres 110EH

 naam ROPSG
 in A - registernummer
 uit A - uit de PSG gelezen byte
 wijzigt A

 adres 1133H

 naam BEEP
 in n.v.t.
 uit n.v.t.
 wijzigt AF,BC,E,EI

 Deze standaardroutine leest een byte uit één van de zestien registers van de PSG. Het nummer van het register wordt naar de Adrespoort van de PSG geschreven, en het databyte gelezen uit de Datapoort.

 adres 113BH

 naam BEEP
 in n.v.t.
 uit n.v.t.
 wijzigt AF,BC,E,EI

 Deze standaardroutine leest de PSG een piepgeluid produceren. Kanal A wordt kleargezet om een toon van 1316 Hz voort te brengen en wordt dan in werking gesteld met een amplitude van zeven. Na 99 msec wordt de controle overgedragen aan GICINI, waar de PSG opnieuw wordt geïnitialiseerd.

 adres 113BH

 naam BEEP
 in n.v.t.
 uit n.v.t.
 wijzigt AF,BC,E,EI

 De interrupt-processor gebruikt deze routine om een muziek-rij af te handelen. Omdat er drie rijen zijn, die elk een kanaal van de PSG bedienen, wordt in register A gespecificeerd welke rij moet worden afgehandeld: 0 - VOIC0A, 1 - VOIC0B, 2 - VOIC0C.

 Elke strekking in een "PLAY"-statement wordt door de BASIC interpreter vertaald in een reeks data-pakketten. Die worden in de toepasselijke rij geplaatst, gevolgd door een eindbyte (FFH). Hoe de pakketten moeten verwerkt worden, gedecodeerd en naar de

PSG toe vertaald, wordt aan de interrupt-verwerker overgelaaten. Op die manier kan de interpreter dus onmiddellijk met het volgende statement doorgaan, zonder op het einde van een bepaalde noot te moeten wachten.

De eerste twee bytes van een data-pakket bepalen hoeveel bytes er komen en de duur van het pakket. De hoogste drie bits van het eerste byte geven aan hoeveel bytes er in het pakket na de header komen. De rest van de header bevat de tijdsduur van het "event" in eenheden van 50 msec. Die teller geeft aan hoe lang het duurt voorlezer het volgende pakket uit de rij gelezen zal worden.

7	6	5	4	3	2	1	0
aantal bytes				tijdsduur (HSB)			
tijdsduur (LSB)							

Fig. 37 : header van een data-pakket

Deze header kan gevolgd worden door een aantal blokken, in een willekeurige volgorde, die informatie bevatten over de frekwentie of de amplitude :

7	6	5	4	3	2	1	0
0	0	x	x	x	x	x	0
frekwentie (LSB)							

Frekwentie-blok

7	6	5	4	3	2	1	0
x	1	x	x	x	x	x	x
envelope-frekwentie (HSB)							
envelope-frekwentie (LSB)							

Envelope-blok

7	6	5	4	3	2	1	0
1	x	x	x	x	x	x	0
Amplitude-blok							

Amplitude-blok

Fig. 38 : types van informatieblokken voor een data-pakket

Vooraf bepaalt de routine waar de tijdsduur-teller staat in de overeenkomstige kanaal-buifer (UCBA, UCBB, UCBC) via GETUP, en trekt daar 1 van af. Wordt daardoor de teller nul, dan is het tijd om het volgende pakket uit de rij te halen; zoniet, stopt

de routine.

Het nummer van de rij wordt in QUEUEN geplaatst, en uit de rij wordt een byte gelezen (IIEZH). Is dat FFH, dan is het einde van de rij bereikt, en de routine stopt (IIBOH). Zoniet wordt het aantal bytes in register C gezet, en het HSB van de tijdsduur in de overeenkomstige kanaalbuifer. Het tweede byte wordt gelezen (IIEZH) en het LSB van de tijdsduur wordt in de overeenkomstige kanaalbuifer gezet. Aan de hand van het aantal bytes wordt bepaald of er nog meer na de header komt. Komt er niets meer, dan stopt de routine. Komt er nog een pakket, dan worden de opeenvolgende bytes uit de rij gelezen, en overeenkomstig gehandeld, tot het vooraf bepaalde aantal bytes is gelezen.

Wordt een frekwentie-blok gevonden, dan wordt een tweede byte gelezen en beide bytes worden naar registers 0 en 1, 2 en 3, 4 en 5 van de PSG geschreven, afhankelijk van het nummer van de rij.

Indien een amplitude-blok werd gevonden, dan worden de amplitude- en mode-bits naar registers 9, 9 of 10 van de PSG geschreven, afhankelijk van het nummer van de rij. Is het modebit 1, wat betekent dat de amplitude gemoduleerd moet worden, dan wordt het byte ook naar register 13 van de PSG geschreven, om de vorm van de envelope te bepalen.

Wordt er een envelope-blok gevonden, of indien bit 5 van een amplitude-blok 1 is, dan worden nog twee bytes uit de rij gelezen, die naar registers 11 en 12 van de PSG geschreven worden.

adres 1180H

Deze routine wordt aangeroepen wanneer een eind-byte (FFH) in een data-pakket in een van de drie muziek-rijen gevonden wordt. Naar register B, 9 of 10 wordt een amplitude-waarde 0 geschreven, afhankelijk van het nummer van de rij, om het betreffende kanaal te sluiten. Het bit in MUSICF dat overeenkomt met dat kanaal wordt op 0 gezet, en de controle wordt overgedragen aan de standaardroutine STRINS.

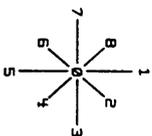
adres 11C4H
naam STRINS
in n.v.t.
uit n.v.t.
wijzigt Af, HL

Deze standaardroutine wordt gebruikt door de "PLAY"-statement verwerker, om een begin te maken met het decoderen van de muziekrijen door de interrupt-verwerker. Vooraf wordt MUSICF bekeken, en indien daaruit blijkt dat er al kanalen werkzaam zijn, stopt de routine zonder meer. Was dat niet zo, dan wordt PLYONI met een verlaagd, en indien er geen "PLAY"-strings meer in de rij staan, stopt de routine. Staan er nog wel, dan worden de drie tijds-tellers in UCBA, UCBB, UCBC op 0001h gezet, zodat bij de eerstvolgende interrupt het eerste nieuw groep gedecodeerd zal worden, en MUSICF wordt op 07H gezet, om alle drie de kanalen in werking te stellen.

 adres 11E2H
 Deze routine leest register A met het nummer van de momenteel uit te lezen rij, uit QUEUEN, en leest vervolgens een byte uit die rij (1940H).

 adres 11E3H
 naam: G1STICK
 in A - identificator van joystick (0, 1 of 2)
 uit A - code voor de stand van de joystick
 wijzigt AF, B, DE, HL, EI

 Deze standaardroutine leest de stand uit van een joystick of van de vier cursor-toetsen. Als de identificator 0 is, wordt de toestand van de cursor-toetsen uitgelezen via poort B van de PPI (12E6H) en omgezet naar een positie-code, die opgezocht wordt in de tabel vanaf 1293H. Bij een andere identificator wordt de aansluiting 1 of 2 uitgelezen (120CH) en de vier richtings-bits omgezet in een positie-code, die opgezocht wordt in de tabel vanaf 12E3H. Deze codes zijn de volgende :



 adres 120CH
 Deze routine leest de aansluiting van de joystick uit, die door register A bepaald wordt : 0 - aansluiting 1, 1 - aansluiting 2. De momentele waarde van register 15 van de PSG wordt uitgelezen, en dan teruggeschreven, met het "joystick-select"-bit op de juiste manier gezet. Vervolgens wordt de inhoud van register 14 van de PSG in register A gezet (11CBH), en de routine stopt.

 adres 12E6H
 Deze routine test rij B van de toetsenbord-matrix af. De momentele waarde van poort C van de PPI wordt ingelezen, en dan teruggeschreven met de vier bits die de rij op het toetsenbord aangeven, zo gewijzigd dat rij B aangegeven wordt. Vervolgens worden de kolommen in register A gelezen via poort B van de PPI.

 adres 12E3H
 naam: G1TRIG
 in A - identificator van trigger (0, 1, 2, 3 of 4)
 uit A - status-code
 wijzigt AF, BC, EI

 Deze standaardroutine controleert de stand van de vuurknop op een joystick of van de "SPACE"-toets, is de identificator 0, dan wordt rij B van de toetsenbord-matrix afgeleest (12E6H) en het resultaat omgezet in een statuscode. In de andere gevallen wordt

 joystick-aansluiting 1 of 2 uitgelezen (120CH) en het resultaat daarvan in een statuscode omgezet. De identificatoren zijn :

- 0 - spatie-toets
- 1 - joystick 1, vuurtoets A
- 2 - joystick 2, vuurtoets A
- 3 - joystick 1, vuurtoets B
- 4 - joystick 2, vuurtoets B

 wordt een bepaalde vuurtoets ingedrukt, dan is het resultaat van de aflezing FFH, zoniet 00H.

 adres 1273H
 naam: G1PDL
 in A - identificator van paddle (1 tot 12)
 uit A - waarde voor die paddle (0 tot 255)
 wijzigt AF, BC, DE, EI

 Deze standaardroutine leest de waarde van elke paddle die aan een joystick-aansluiting is aangesloten. Aan elk van de zes input-lijnen (vier voor de richting en twee vuurtoetsen) van elke aansluiting kan een paddle aangesloten worden, zodat er in totaal twaalf mogelijk zijn. De paddles aan aansluiting 1 worden geïdentificeerd als 1,3,5,7,9 en 11. De paddles aan aansluiting 2 zijn 2,4,6,8,10 en 12. Elke paddle is in de grond een monostabiele puls-generator, waarvan de puls-lengte door een regelbare weerstand gestuurd wordt. Via register 15 van de PSG wordt een start-puls gestuurd naar de opgegeven joystick-aansluiting. Vervolgens wordt geteld hoe vaak register 14 van de PSG uitgelezen moet worden tot die bepaalde input terug nul wordt. Elke eenheid komt op een MSX-machinette met één "WAIT"-toestand, overeen met ongeveer 12 µsec.

 adres 12ACH
 naam: G1PAD
 in A - functiecode (0 tot 7)
 uit A - status of waarde
 wijzigt AF, BC, DE, HL, EI

 Via deze standaardroutine kan een elektronisch tokenbord ("touchpad") uitgelezen worden, dat aan een van de joystick-aansluitingen is gekoppeld. De beschikbare functiescodes voor aansluiting 1 zijn :

- 0 - geef status (aan/uit)
- 1 - geef X-coördinaat
- 2 - geef Y-coördinaat
- 3 - geef stand van schakelaar

 De codes 4 tot 7 leveren hetzelfde op, maar van aansluiting 2. Code 0 (4) geeft FFH als resultaat indien het bord aangeeraakt wordt, zoniet 0. Code 3 (7) geeft FFH indien de schakelaar ingedrukt wordt, zoniet 0. De codes 1 en 2 (5 en 6) leveren de coördinaten op van het laatste aangeraakte punt. De coördinaten worden opgeslagen in het werkgebied, in de variabelen PADX en PADY, wanneer middels codes 0 of 1 activiteit ontdekt wordt. Merk op dat deze variabelen door beide joystick-aansluitingen worden gebruikt.

adres 1394H
 naam STNDR
 in A - code voor motor aan/uit
 uit n.v.t.
 wijzigt AF

Deze standaardroutine zet het relais dat de cassettemotor stuur, aan of uit via poort C van de PPI : 00H = uit, 01H = aan, FFH = omgekeerde van de huidige stand.

adres 1398H
 naam NMI
 in n.v.t.
 uit n.v.t.
 wijzigt niets

Deze standaardroutine verwerkt normaal een niet-maakbare interrupt van de Z80. Op een standaard MSX-machin doet ze niets.

adres 139DH
 naam INIFNK
 in n.v.t.
 uit n.v.t.
 wijzigt BC, DE, HL

Deze standaardroutine initialiseert de strings die bij de tien functietoetsen horen, met de inhoud die ze hebben bij het aanzetten van de machine. De honderdste bit van de string worden naar de FMSIR-bufter in het werkgebied gecopieerd.

adres 13A9H IABEL

Op dit adres begint de tabel van 160 bytes lang, die de oorspronkelijke tekst bevat voor de functietoetsen. Elke string is 16 tekens lang, en op ongebruikte posities staan nullen :

F1 tot F5	F6 tot F10
color	color 15,'4,'4 + "CR"
auto	color
goto	cont + "CR"
list	list . + "CR" + 2 x "omhoog"-code
run + "CR"	"CLS" + run + "CR"

adres 1449H
 naam RDUDP
 in n.v.t.
 uit A - inhoud van statusregister van de U.D.P.
 wijzigt A

Deze standaardroutine leest de inhoud van het statusregister van de Video Display Processor uit via de command-poort. Merk wel op dat deze uitlesing de bijbehorende vlaggen allemaal reset, en ook de interrupt-verwerker kan beïnvloeden.

adres 144CH
 naam RSLREG
 in n.v.t.
 uit A - inhoud van primair slot-register
 wijzigt A

Deze standaardroutine leest de inhoud van het primair slotregister via poort A van de Parallele periferie-interface.

adres 144FH
 naam WSLREG
 in A - te schrijven byte
 uit n.v.t.
 wijzigt niets

Deze standaardroutine schrijft, via poort A van de Parallele periferie-interface, een bepaalde waarde naar het primaire slot-register.

adres 1452H
 naam SNSPFI
 in A - nummer van een rij van het toetsenbord
 uit A - kolom-gegevens van die rij
 wijzigt AF,C,EI

Deze routine leest een volledige rij van de toetsenbord-matrix uit. Poort C van de Parallele periferie-interface wordt gelezen, en teruggeschreven, met nu op de plaats van de vier bits die de r'ij bepalen, het nummer van de te lezen r'ij. Dan wordt poort B van de Parallele periferie-interface in register A gelezen, waardoor de toestand van de acht kolommen bekend is. De vier andere controle-outputs van poort C van de Parallele periferie-interface worden door deze routine niet gewijzigd.

adres 145FH
 naam ISFLIO
 in n.v.t.
 uit vlag NZ indien een In/uit-file actief is
 wijzigt AF

Deze standaardroutine controleert of de BASIC Interpreter op dat ogenblik input of output via een in/uit-bufter leidt. Daartoe wordt PIRFIL bekeken. Normaal is die 0, maar wanneer statements zoals "PRINT#1", "INPUT#1" en dergelijke door de Interpreter verwerkt worden, staat daar een bufferadres van een controlblok voor dat bepaald kanaal (*).

adres 146AH
 naam DCONPR
 in HL, DE
 uit NC als HL>DE, Z als HL=DE, C als HL<DE
 wijzigt AF

De BASIC Interpreter gebruikt deze standaardroutine om de waarden die in HL en DE staan, te vergelijken.

1410
1497M

naam GETVCP
in A - kanaal-nummer
uit HL - adres in kanaalbuffer
wijzigt AF, HL

naam GETV2
in L - nummer van een byte (0 tot 36)
uit HL - adres in kanaalbuffer
wijzigt AF, HL

naam PNYDIO
in n.v.t.
uit n.v.t.
wijzigt niets

naam FORPOT
in n.v.t.
uit n.v.t.
wijzigt niets

naam PUTD
in A - nummer van een rij, E - databyte
uit vlag Z indien rij vol is
wijzigt AF, BC, HL

naam 1492K
in A - kanaal-nummer
uit vlag Z indien rij vol is
wijzigt AF, BC, HL

naam 149DK
in A - kanaal-nummer
uit vlag Z indien rij vol is
wijzigt AF, BC, HL

naam 149DK
in A - kanaal-nummer
uit vlag Z indien rij vol is
wijzigt AF, BC, HL

naam LFTD
in A - nummer van een rij
uit HL - de nog vrije ruimte in die rij
wijzigt AF, BC, HL

naam 149BK
in A - kanaal-nummer
uit vlag Z indien rij vol is
wijzigt AF, BC, HL

naam 149BK
in A - kanaal-nummer
uit vlag Z indien rij vol is
wijzigt AF, BC, HL

naam 150YH
in A - kanaal-nummer
uit vlag Z indien rij vol is
wijzigt AF, BC, HL

```

adres 1510H
naam GRPPT
in R Karaktercode
uit n.v.t.
wijzigt EI

```

Deze standaardroutine zet een karakter op het scherm in grafische- en veelkleurenmode. Functioneel doet ze hetzelfde als de standaardroutine CHPUT.

Vooraf wordt via de standaardroutine CHVCHR gecontroleerd of het om een grafisch karakter gaat. Gaat het om een grafische header (01H) dan stopt de routine zonder meer. Gaat het om een omgezet grafisch karakter, dan wordt het gedeelte waar de controlecodes worden gedecodeerd, overgeplaat. Zoniet, wordt gecontroleerd of het om een controlecode gaat. Enkel een "CR" wordt herkend (157EH), codes lager dan 20H worden genegeerd.

In de veronderstelling dat het karakter schrijfbaar is, wordt het pixelpatroon ervan uit de ROM karakterset gecopieerd naar de PATWRK-buffert (0752H) en FORCLR gecopieerd naar ATRBYT om de kleur ervan in te vullen. Uit GRPACX en GRPACV worden vervolgens de huidige grafische coördinaten gehaald waaruit het adres van de pixel links bovenaan het karakter-veld berekend wordt via de standaardroutines SCALXY en HAPXYC.

In het acht bytes tellende patroon in PATWRK wordt één byte per keer verwerkt. Bij het begin van elk byte wordt het adres van de huidige pixel berekend via de standaardroutine FETICH, en weggezet. Dan worden de acht bits na elkaar bekeken. Is een bit 1, dan wordt de overeenkomstige pixel gezet door de standaardroutine SETC. Is het bit nul, dan gebeurt er niets. Na elk bit wordt de positie van de pixel op het scherm één punt naar rechts gezet (SRCH). Is het byte afgewerkt, of wordt de rechterrand van het scherm bereikt, dan wordt de oorspronkelijke pixel-positie teruggehaald en één rij omhoog gebracht door de standaardroutine TDDWNC.

Is het patroon afgewerkt, of wordt de onderkant van het scherm bereikt, dan wordt GRPACX aangepast. In grafische mode wil dit zeggen: vermeerderd met acht; in veelkleurenmode, met 32. Is GRPACX daarna hoger dan 255 (de rechterkant van het scherm) dan wordt een "CR" uitgevoerd (157EH).

```

adres 157EH

```

Deze routine verzorgt de "CR"-beweging voor de standaardroutine GRPPT. Ze bestaat uit een combinatie van "CR" + "Line Feed". GRPACV wordt op 0 gezet, en bij GRPACV wordt 8 of 32 opgeteld, afhankelijk van de scher-mode. In GRPACV daardoor hoger dan 191 (de onderkant van het scherm) dan wordt deze 0 gemaakt.

Een toepassingsprogramma kan GRPACX en GRPACV rechtstreeks bewerken, ter compensatie van het beperkte aantal controlefuncties die ter beschikking staan.

```

adres 1599H
naam SCALXY
in BC = X-coördinaat, DE = Y-coördinaat
uit vlieg NC indien aangepast
wijzigt AF

```

Deze standaardroutine beperkt, indien nodig, grafische coördinaten. De BASIC Interpreter kan coördinaten berekenen tussen -32768 en +32767, hoewel dit ruimschoots buiten het schermbereik ligt. Deze routine wijzigt coördinaten die te groot zijn, zodanig dat ze binnen het fysisch bereikbare gebied vallen. Is X groter dan 255, dan wordt hij 255 gemaakt. Is Y groter dan 191, dan wordt hij 191. Is één van beide negatief (d.w.z. hoger dan 7FFFH), dan wordt hij 0. Tenslotte worden beide coördinaten, indien het scherm in de veelkleurenmode staat, door vier gedeeld, zoals verast door de standaardroutine HAPXYC.

```

adres 15D9H

```

Het deze routine ken de huidige scher-mode onderzocht worden. Wanneer het scherm in grafische mode staat, wordt de Z vlieg gezet.

```

adres 15D9H
naam HAPXYC
in BC = X-coördinaat, DE = Y-coördinaat
uit n.v.t.
wijzigt AF,D,H

```

Het deze standaardroutine wordt een grafisch coördinatenpaar omgezet in een pixelpositie. De plaats in de karaktertabel waar het byte staat, waarin de pixel zich bevindt, wordt in CLOC gezet. Het bitmasker, waardoor de pixel binnen het byte geïdentificeerd kan worden, wordt in CHMSK gezet. Voor grafische mode wordt een iets verschillende opzetting gebruikt dan voor veelkleurenmode. De gelijklopende BASIC programma's zien er zo uit :

Grafische Mode

```

10 INPUT "X,Y Coördinaten";X,Y
20 A=(Y/8)*255+(Y AND 7)*X AND &HFB
30 PRINT"ADRES=";HEX$(BASE(12)+A);";H=";
40 RESTORE 100
50 FOR N=0 TO CX AND 7: READ H$: NEXT N
60 PRINT"MSK=";H$
70 GOTO 10
110 DATA 10000000
120 DATA 01000000
130 DATA 00100000
140 DATA 00010000
150 DATA 00001000
160 DATA 00000100
170 DATA 00000010

```

VeeIkLeuren Mode

```
10 INPUT "X,Y Coördinaten":X,Y
20 X=X\4:Y=Y\4
30 A=(Y\8)*256+(Y AND 7)*(X\4 AND 8)&FB)
40 PRINT"ADRES=";HEX$(BASE(17)+A); "H ";
50 IF X MOD 2=0 THEN HS="11110000" ELSE HS="00001111"
60 PRINT"MASK=";HS
70 GOTO 10
```

De toegestane waarden bij beide programma's zijn voor X: 0 tot 255 en voor Y: 0 tot 191. De data-regels in het programma voor de grafische mode komen overeen met de masker-tabel van acht bytes lang, op adres 160BH in de MSX-ROM. Regal 20 in het veeIkLeuren-programma komt overeen met de daling door vier die gebeurt in SCALXY. Die regel staat in het programma, om het coördinatenstelsel voor beide programma's gelijk te doen lopen.

```
adres 1639H
naam FEICHC
in n.v.t.
uit A - CHASK, HL - CLOC
wijzigt A, HL
```

Deze standaardroutine haalt het adres op van de huidige pixel ; registerpaar HL neemt de waarde uit CLOC en register A die uit CHASK.

```
adres 1640H
naam STOREC
in A - CHASK, HL - CLOC
uit n.v.t.
wijzigt niets
```

Deze standaardroutine zet in CLOC de inhoud van registerpaar HL (pixeladres) en in CHASK de inhoud van register A.

```
adres 1647H
naam READC
in n.v.t.
uit A - Kleurcode van huidige pixel
wijzigt AF, EI
```

Deze standaardroutine berekent de kleur van een pixel. Vooraft wordt het adres in URH berekend via de standaardroutine FEICHC. Staat het scherm in grafische mode, dan wordt het byte waarnaar CLOC verwijst, uit de Karaktertabel gelezen via de standaardroutine RDUH. De benodigde bit wordt dan, met behulp van CHASK, geïsoleerd, en gebruikt om in de overeenkomstige locatie van de Kleurentabel de bovenste of onderste vier bits te lezen.

Staat het scherm in veeIkLeurenmode, dan wordt het byte waarnaar CLOC verwijst, uit de Karaktertabel gelezen via de standaardroutine RDUH. Dan worden met CHASK de hoogste of laagste vier bits geïsoleerd. In elk van beide gevallen zal een waarde berekend worden die een normale kleur voor de video

Display Processor aangeeft, die ligt tussen nul en vijftien.

```
adres 1676H
naam SEIATR
in A - Kleurcode
uit Vlag C indien code illegaal was
wijzigt de vlaggen
```

Deze standaardroutine vult de inkkleur in voor grafisch gebruik, die door de standaardroutine SEIC en SEICX gebruikt wordt. De kleurcode, tussen nul en vijftien, wordt gewoon in AIRBYI gezet.

```
adres 167EH
naam SEIC
in n.v.t.
uit n.v.t.
wijzigt AF, EI
```

Deze standaardroutine zet de laatste geschreven pixel in een bepaalde kleur. De kleurcode wordt uit AIRBYI gehaald. Vooraft wordt het adres van de pixel berekend met de standaardroutine FEICHC. In grafische mode wordt zowel de Karaktertabel als de Kleurentabel gewijzigd.

In veeIkLeurenmode wordt, middels de standaardroutine RDUH, het byte waar CLOC naar verwijst uit de Karaktertabel gelezen. Vervolgens wordt de inhoud van AIRBYI in de hoogste of laagste vier bits gezet, volgens CHASK, en het byte wordt via de standaardroutine URH teruggeschreven.

```
adres 168CH
naam RIGITC
in n.v.t.
uit n.v.t.
wijzigt AF
```

Deze routine beweegt de scherm-positie van de laatste pixel één punt naar rechts. Lig dit buiten de rand van het scherm, dan wordt de positie niet gewijzigd, en de routine stopt met vlag C. In grafische mode wordt CHASK eerst een bit naar rechts geschoven. Lig de pixel dan nog binnen hetzelfde byte, dan stopt de routine hier. Verwijst CLOC naar de uiterst rechts karaktercel (LSB - FBH tot FFH) dan stopt de routine met vlag C (175AH). Zoniet wordt CHASK op 80H gezet (de uiterst linkse pixel) en wordt er bij CLOC 8 opgeteld.

In veeIkLeurenmode wordt de controle overgedragen aan een afzonderlijke routine (1779H).

```
adres 16C5H
naam RIGITC
in n.v.t.
uit n.v.t.
wijzigt AF
```

Deze standaardroutine verzet de schermpositie van de laatste pixel een punt naar rechts. In grafische mode wordt CHASK eerst een bit naar rechts geschoven. Lig de pixel dan nog binnen hetzelfde byte, dan stopt de routine. Zoniet wordt CHASK op 80H gezet (de uiterst linkse pixel van een byte) en bij CLOC wordt 8

opgesteld. Let er op, dat de berekende positie onjuist is, als verder gegaan wordt dan de rechterrind van het scherm!

In veelkleurenmode wordt de controle overgenomen door een afzonderlijke routine (178BH).

adres 15DBH

Deze routine verzet de laatste pixel één positie naar links. Ligt hij daardoor buiten de linkerind van het scherm, dan wordt het adres niet gewijzigd en stopt de routine met vlag C. In grafische mode wordt vooraf CHSK één bit naar links geschoven. Ligt de pixel dan nog binnen hetzelfde byte, dan stopt de routine hier. Als CLOC naar de uiterst linkse karaktercol verwijst (LSB = 00H tot 07H), dan stopt de routine met vlag C (175AH). Is dat niet zo, wordt CHSK op 01H gezet (uiterst rechtse pixel) en van CLOC wordt 8 afgetrokken.

In veelkleurenmode neemt een afzonderlijke routine de controle over (179CH).

adres 16E8H

naam LEFTC
in n.v.t.
uit n.v.t.
wijzigd AF

Deze standaardroutine verzet de schermpositie van de laatste pixel een punt naar links. In grafische mode wordt eerst CHSK één bit naar links geschoven. Ligt de pixel nog binnen hetzelfde byte, dan stopt de routine hier. Zoniet, wordt CHSK op 01H gezet (de uiterst rechtse pixel) en van CLOC wordt 8 afgetrokken. Let er op dat, indien de linkerind van het scherm werd overschreden, de berekende positie onjuist is.

In veelkleurenmode neemt een afzonderlijke routine de controle over (17ACH).

adres 170AH

naam TDDWNC
in n.v.t.
uit vlag C indien buiten scherm
wijzigd AF

Deze standaardroutine verzet de laatste pixel een positie omhoog. Ligt die positie daardoor buiten het scherm, dan wordt het adres niet gewijzigd en stopt de routine met vlag C. In grafische mode wordt eerst CLOC met 1 vermeerderd. Valt het resultaat dan nog binnen een acht-byte-grens, dan stopt de routine. Als CLOC binnen de laatste regel (karakters) lag (CLOC >= 1700H) dan stopt de routine met vlag C (1759H). Zoniet wordt bij CLOC 00F8H opgeteld.

In veelkleurenmode wordt de controle overgedragen aan een afzonderlijke routine (17CBH).

adres 172AH
naam DOWNC
in n.v.t.
uit n.v.t.
wijzigd AF

Deze standaardroutine zet de laatste pixel één positie lager. In grafische mode wordt CLOC eerst met 1 vermeerderd. Valt het dan nog binnen een acht-byte grens, dan eindigt de routine. Zoniet wordt 00F8H bij CLOC opgeteld. Let er op dat, indien de onderkant van het scherm wordt overschreden, de berekende positie onjuist is!

In veelkleurenmode wordt de controle overgedragen aan een afzonderlijke routine (17DCH).

adres 173CH

naam IUPC
in n.v.t.
uit vlag C indien buiten scherm
wijzigd AF

Deze standaardroutine verzet de laatste pixel één positie omhoog. Wordt de boverrind van het scherm overschreden, dan blijft de positie wat ze was en eindigt de routine met vlag C. In grafische mode wordt eerst 1 afgetrokken van CLOC; ligt dat dan nog binnen een acht-byte grens, dan eindigt de routine. Lag CLOC op de bovenste regel (karakters) (CLOC < 0100H) dan eindigt de routine met vlag C, zoniet wordt 00F8H van CLOC afgetrokken.

In veelkleurenmode wordt de controle overgedragen aan een afzonderlijke routine (17E3H).

adres 175DH

naam UPC
in n.v.t.
uit n.v.t.
wijzigd AF

Standaardroutine om de laatste pixel een positie hoger te zetten. In grafische mode wordt eerst CLOC met 1 verlaagd. Ligt het dan nog binnen een acht-byte grens, dan eindigt de routine. Zoniet wordt van CLOC 00F8H afgetrokken. Let er op dat, indien de boverrind van het scherm wordt overschreden, de berekende positie niet juist is.

In veelkleurenmode wordt de controle overgedragen aan een afzonderlijke routine (17FBH).

adres 1779H

Deze routine is de veelkleuren-versie van de routine op 16ACH. Ze is identiek aan deze laatste, behalve dat hier CHSK vier keer naar rechts geschoven wordt en F0H wordt indien de grens van een karaktercol wordt overschreden.

adres 178BH

Deze routine is de veelkleuren-versie van RIGHTC (16GSH). Ze is identiek aan deze laatste, behalve dat hier CHASK vier keer naar rechts geschoven wordt en F0H wordt indien de grens van een karaktercel wordt overschreden.

adres 179CH

Deze routine is de veelkleuren-versie van de routine op 16DBH. Ze is identiek aan deze laatste, behalve dat hier CHASK vier keer naar links geschoven wordt en 0FH wordt indien de grens van een karaktercel wordt overschreden.

adres 17ACH

Deze routine is de veelkleuren-versie van LEFTC. Ze is identiek aan deze laatste, behalve dat hier CHASK vier keer naar links geschoven wordt en 0FH wordt indien de grens van een karaktercel wordt overschreden.

adres 17C6H

Dit is de veelkleuren-versie van IDOWNC (170AH). Ze is identiek aan deze laatste, behalve dat hier het grensadres voor de onderkant van het scherm 0500H is in plaats van 1700H. In deze routine zit een foutje, waardoor ze onvoorspelbare dingen gaat doen indien H1TCGP het startadres van de Karaktertabel op een andere waarde dan nul, zijn normale waarde, gezet wordt. Dit adres 17CEH hoort een EX DE:HL-instructie tussengevoegd te worden! Indien het startadres van de Karaktertabel verhoogd wordt, dan zal de routine reageren alsof de bodem van het scherm bereikt werd, hoewel dat niet zo is. Deze routine wordt door het PAINT-statement gebruikt. Het volgende programma illustreert de fout:

```
10 BASE(17)-$H1000
20 SCREEN 3
30 PSET(200,0)
40 DRAW:DI80,100:U100R100*
50 PAINT(150,90)
60 GOTO 60
```

adres 17DCK

Dit is de veelkleuren-versie van DOWNC (172AH). Ze is identiek aan de routine voor de grafische mode.

adres 17E3H

Dit is de veelkleuren-versie van TUPC (173CH). Ze is identiek aan de versie voor de grafische mode, behalve dat er ook een foutje in zit. Op adres 17EBH hoort een EX DE:HL-instructie tussengevoegd te worden! Indien het startadres van de Karaktertabel verhoogd wordt, zal de routine reageren alsof ze nog binnen de tabel werkt, terwijl ze eigenlijk al boven de rand van het scherm werkt. Het effect kan bereken worden door in regel 40 van het vorige programma, het "R100"-gedeelte weg te halen.

84

adres 17FBH

Dit is de veelkleuren-versie van UPC (175DH). Ze is identiek aan de versie voor de grafische mode.

adres 1809H

```
naam NSETCX
in HL - aantal te bewerken pixels
uit n.v.t.
wijzigt AF,BC,DE,HL,EI
```

Deze standaardroutine kleurt een reeks pixels, vanaf de laatste geplote pixel, horizontaal en naar rechts toe. Ze kan nageboot worden door de standaardroutine SPIC en RIGHTC, maar dat zou de werking ervan aanzienlijk vertragen. Het aantal pixels dat in registerpaar HL aangegeven wordt, moet zo berekend worden dat de rechter-rend van het scherm niet overschreden wordt. Dit zou tot zeer vreemde effecten leiden. Deze routine behoudt de coördinaten van de laatste geplote pixel (voordat de routine zelf begon te werken).

In grafische mode wordt eerst met behulp van CHASK berekend hoeveel pixels naar rechts nog in de huidige karaktercel liggen. In de veronderstelling dat de inhoud van registerpaar HL voldoende groot is, worden die dan allemaal gezet. Het overblijvende aantal wordt door B gedeeld, om het aantal volledige karaktercellen te bepalen. Opvolgende bytes in de karaktertabel worden vervolgens 0 gemaakt, en de overeenkomstige bytes in de kleurentabel met de inhoud van AIBBYI gevuld, om deze cellen te "vullen". Wat dan nog HL wordt in het oorspronkelijke aantal in registerpaar HL, wordt in een bitmasker omgezet, met behulp van een zeven bytes lange tabel op adres 185DH. Deze pixels worden daarna gezet (186CH).

In veelkleurenmode wordt de controle overgedragen aan een afzonderlijke routine (188BH).

adres 186CH

Deze routine zet tot maximaal acht pixels binnen een karaktercel in een bepaalde kleur, in grafische mode. De kleurcode wordt in AIBBYI gehaald, registerpaar HL bevat het adres van het overeenkomstige byte in de karaktertabel, en register A bevat een bitmasker (bv. 11100000) waarin elke 1 een bit voorstelt dat ingekleurd moet worden.

Indien AIBBYI overeenkomt met de bestaande kleur van een gezette pixel in het overeenkomstige byte in de kleurentabel, dan wordt elk gespecificeerd bit in het byte in de karaktertabel gezet. Indien AIBBYI overeenkomt met de bestaande kleur van een gezette pixel in het overeenkomstige byte in de kleurentabel, dan wordt elk gespecificeerd bit in het byte in de karaktertabel gezet.

Indien AIBBYI met geen van beide bestaande kleuren in het byte in de kleurentabel overeenkomt, dan wordt normaal elk gespecificeerd bit in het byte in de karaktertabel gezet, en de kleur van een gezette pixel in het byte in de kleurentabel wordt gewijzigd. Indien dit evenwel tot gevolg zou hebben dat alle

85

bits in het byte in de Karaktertabel gezet zouden worden, dan wordt elk gespecificeerd bit gezet en de kleur van een gesetste pixel wordt in het byte in de Kleurentabel gewijzigd.

adres 18B8H

Dit is de veelkleurenversie van de standaardroutine NSEIX, de standaardroutine SEIC en RIGHT worden zo vaak aangeroepen tot het opgegeven aantal pixels gekleurd zijn. De uitvoeringssnelheid is in veelkleurenmode niet zo belangrijk, omwille van de lagere scherme-resolutie en het daaruit volgende lagere aantal noodzakelijke bewerkingen.

adres 18C7H

naam GTRASPC
in n.v.t.
uit DE - ASPCTI, HL - ASPCT2
wijzig DE, HL

Deze standaardroutine berekent de verhoudingen bij een "CIRCLE"-statement, indien er geen worden gespecificeerd.

adres 18CFH

naam PNTINI
in A - grenskleur (0 tot 15)
uit Vleg C indien niet toegelaten kleur
wijzig AF

Deze standaardroutine bepaalt de grenskleur voor het "PAINI"-statement. In veelkleurenmode wordt de opgegeven kleur in BDRATR gezet. In grafische mode wordt BDRATR uit KIRBYT gecopieerd, omdat het niet mogelijk is in die mode een kleur te hebben voor de grenslijn die verschilt van de kleur voor het opgevulde vlak.

adres 18E4H

naam SCANR
in B - "fill"-schakelaar, DE - hoeveel overslaan
uit DE - hoeveel nog overslaan, HL - aantal pixels
wijzig AF, BC, DE, HL, EI

Deze standaardroutine wordt door de "PAINI"-statement verwerkt gebruikt om naar rechts toe te zoeken, vanaf de laatste pixel, tot de kleurcode van BDRATR wordt gevonden, of de rand van het scherm werd bereikt. De eindpositie wordt nu het adres van de laatste pixel, en de startpositie wordt in CSARFA en CSARFB gezet. De lengte van het doortrokken gebied wordt in doorlopen gebied ingekleurd, maar dit kan vermindert worden, maar alleen in grafische mode, door in register B nul te zetten bij het begin van de routine. Hoeveel pixels in de opgegeven kleur overgeslagen mogen worden, vanaf de oorspronkelijke uitgangspositie, wordt in registerpaar DE van tevoren bepaald. Deze voorziening wordt door de "PAINI"-statement verwerkt gebruikt, bij het zoeken naar geten in een horizontale grens, die normaal verhindert dat naar boven toe gezocht wordt.

adres 197AH

naam SCANL
in n.v.t.
uit HL - aantal pixels
wijzig AF, BC, DE, HL, EI

Deze standaardroutine zoekt naar links toe, vanaf het adres van de laatste pixel, tot de kleurcode in BDRATR gevonden wordt of de rand van het scherm wordt bereikt. De eindpositie wordt het nieuwe adres van de laatste pixel, en de lengte van het doorlopen gebied wordt in registerpaar HL gezet. Het doorlopen gebied wordt altijd ingekleurd.

adres 19C7H

Deze routine wordt door SCANL en SCANR gebruikt, om de kleur van de laatste pixel te vergelijken met de grenskleur in BDRATR.

adres 19DDH

naam TAPDOF
in n.v.t.
uit 0.v.t.
wijzig EI

Deze standaardroutine zet de motor van de cassette-recorder uit, nadat er data op de cassette werden geschreven. Na 550 msec. (op een machine met een "wait"-toestand) gaat de routine over in de standaardroutine TAPIOF.

adres 19E9H

naam TAPIOF
in n.v.t.
uit n.v.t.
wijzig EI

Deze standaardroutine zet de motor van de cassette-recorder uit, nadat er data van de band werden gelezen. De nodopoort van de parallelle periferie-interface zorgt ervoor dat het relais geopend wordt. Let er op dat de interrupt op het einde van deze routine terug in werking worden gesteld: tijdens de overdracht van data van en naar cassette dienen die afgezet te worden, omdat de timing nauw juistert.

adres 19F1H

naam TAPDND
in A - bepaald voor lengte van de header
uit Vleg C indien onderbroken door "CTRL/STOP"
wijzig AF, BC, HL, DI

Deze standaardroutine zet de motor van de cassette-recorder aan, wacht 550 msec om de band de tijd te geven op snelheid te komen, en schrijft dan een "header" naar de cassette. Een header is een groep van digitaal hoge signalen, die voor elk datablok geschreven worden, om de bandreits te kunnen bepalen bij het teruglezen van dat blok.

De lengte van de header wordt bepaald door register A : 00H geeft een korte, iets anders geeft een lange header. De BASIC statements "SAVE", "CSAVE" en "BSAVE" genereren een lange header bij het begin van een file, en naelden korte headers tussen elke word datablokken. Het aantal cijfci in de header wordt ook beïnvloed door de ingestelde baudrate, om de tijdsduur ervan konstant te houden :

```
1200 baud kort ... 3640 cijfci ... 1,5 seconde
1200 baud lang ... 15360 cijfci ... 6,1 seconde
2400 baud kort ... 7936 cijfci ... 1,6 seconde
2400 baud lang ... 31744 cijfci ... 6,3 seconde
```

De motor wordt aangezet, en er wordt even gewacht. Dan wordt de inhoud van HEADER met 256 vermenigvuldigd en indien register A een van nul verschillende waarde bevat, nog eens met vier, om het nodige aantal cijfci te verkrijgen. vervolgens wordt het berekende aantal hogegende pulsen geproduceerd, en de controle gaat over naar de standaardroutine BREAK. Rangziken de CTRL/STOP--toetsen pas op het einde van de routine worden afgetest, kan ze niet halfweg worden onderbroken.

```
adres 1A13H
-----
naam TAPUI
in A - databyte
uit Vlag C indien onderbroken door "CTRL/STOP"
wizigt AF, B, HL
```

Deze standaardroutine schrijft één byte naar cassette. De NSX ROM maakt gebruik van de FSK-methode (Frequency Shift Keyed : gecodeerd door frequentie-verschuiving) om informatie op een cassette te bewaren. Bij 1200 baud is die identiek aan de Kansas City Standard, die door de BBC wordt gebruikt voor het overbrengen van Basiccode-programma's.

Bij 1200 baud wordt elk "0"-bit geschreven als een volledige LAG-cijclus van 1200 Hz, en elk "1"-bit als twee volledige HD06-cijci van 2400 Hz. Zo wordt de datasnelheid konstant gehouden, doordat de "0" en "1"-bits even lang duren, werd voor 2400 baud gekozen, dan worden de frequenties respectievelijk 2400 en 4800 Hz, maar de verhoudingen blijven dezelfde.

Een databyte wordt geschreven, voorafgestaan door een startbit (0) (1A50H), dan acht databits, met bit 0 eerst, en gevolgd door twee stop-bits (1) (1A40H). Bij 1200 baud duurt één byte dus normaal 119833 jasc - 9,2 msec. Na het schrijven van de stop-bits controleert de standaardroutine BREAK of "CTRL/STOP" werd ingedrukt. Als voorbeeld geven we hieronder weer hoe het byte 43H naar cassette geschreven wordt :



Het is van belang, geen te lange tijd te laten tussen twee bytes, wanneer data worden weggeschreven, omdat daardoor de kans

op fouten vergroot. Een pauze van 80 jasc tussen twee bytes, bijvoorbeeld, geeft bij teruglezen ongeveer twaalf procent fouten. Als tussen twee bytes in nogal veel moet gebeuren, dan kan het beste gebufferd worden, om de data als blokken, met een header, te verzamelen. Dat gebeurt ook bij het "SAVE"-commando.

```
adres 1A39H
-----
Deze routine schrijft één LAG-cijclus van ongeveer 816 jasc naar cassette. De lengte van elke helft van de cijclus wordt uit LDW gelezen. Nadien gaat de controle over naar de algemene routine die cijci produceert (1A50H).
```

```
adres 1A40H
-----
Deze routine schrijft twee HD06-cijci van ongeveer 366 jasc naar cassette. De lengte van elke helft van de cijclus wordt uit HIGH gelezen. Nadien gaat de controle over naar de algemene routine die cijci produceert (1A50H).
```

```
adres 1A50H
-----
Deze routine schrijft één cijclus naar cassette. De lengte van de eerste helft van de cijclus wordt in register L gezet, en die van de tweede helft in register H. De eerste lengte wordt afgeteld, en dan wordt het "Car Out"-bit via de Hodepoort van de Parallele periferie-interface gezet. Nu wordt de tweede lengte afgeteld, en het bit wordt gereset.
```

Bij alle NSX-machines heeft de 280 een klokfrequentie van 3,579545 MHz (280 nsec) met één "wait"-status tijdens de HI-cijclus. Deze routine telt 16 T-tijden als eenheid, zodat elke tel met 4,47 jasc overeenkomt. Er wordt ook onverschillig om welke lengte het gaat, een extra 20,7 jasc vast bijgeteld.

```
adres 1A63H
-----
naam TAPIDN
in n.v.t.
uit Vlag C indien onderbroken door "CTRL/STOP"
wizigt AF, BC, DE, HL, DI
```

Deze standaardroutine zet de motor van de cassette-recorder aan, en leest de cassette tot een header gevonden wordt; dan wordt de baudrate bepaald. Opervolgende cijci worden ingelezen, en de lengte van elke cijclus wordt gemeten (1B34H). Wanneer er 1.111 cijci gevonden worden, met minder dan 35 jasc verschil in lengte, dan gaat het om een header.

Nu worden de volgende 256 cijci gelezen (1B34H), en er wordt een gemiddelde van gemaakt, om de lengte van een HD06-cijclus op die cassette te bepalen. Dit getal wordt met 1,5 vermenigvuldigd en in LDWLIH gezet : daardoor wordt de minimaal aanvaardbare lengte van een startbit (0) bepaald. De lengte van een HD06-cijclus wordt in WINDID gezet. Daar wordt het opgehaald, om het verschil te maken tussen LAGS- en HD06-cijci.

adres 1ABCH
 naam TAPIN
 in n.v.t.
 uit A - gelezen byte; vlag C indien CTRL/STOP of I/O Error
 wijzigt AF,BC,DE,L

Deze standaardroutine leest een databyte van cassette. Eerst wordt de cassette continu gelezen, tot een startbit gevonden wordt. Dit gebeurt door te zoeken naar een laaggaand signaal, de lengte van de volgende cyclus te meten (1B1FH) en die te vergelijken met LOWLIN, om te zien of ze groter is.

De acht databits worden dan ingelezen, door het aantal hoog/laag-overgangen te tellen binnen een bepaalde periode (1B03H). Is dat aantal 0 of 1, dan gaat het om een "0"-bit. Zijn er twee of drie overgangen, dan gaat het om een "1"-bit. Worden er meer dan drie overgangen geteld, dan stopt de routine, worden vlag C gezet; er wordt aangenomen dat het in dat geval om een of andere fout in de hardware gaat. Na de bepaling van de waarde van elk bit, worden er nog een of twee overgangen gelezen, om de synchronisatie te behouden. Wordt er voordien een oneven aantal geteld, dan wordt er nog één gelezen, in het andere geval nog twee.

adres 1B03H

Deze routine wordt door de standaardroutine TAPIN gebruikt om het aantal hoog/laag-overgangen te tellen binnen een bepaalde periode. De lengte van die periode staat in WINWID en is ongeveer 1,5 keer een H00G-cyclus, zoals de figuur hieronder toont:

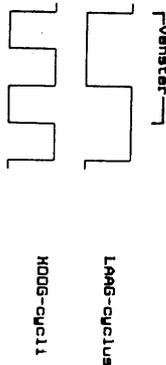


Fig. 40 : meetperiode voor overgangen

Het "Cas Input"-bit wordt voortdurend via register 14 van de Programmeerbare geluidsgenerator bekeken, en vergeleken met de voorgaande uitlesing, in register E. Elke keer als een overgang vastgesteld wordt, verhoogt register C met 1. Dit controlis gebeurt om de 17,3 µsec. Dat wil zeggen dat de waarde in WINWID, die door de standaardroutine TAPIN werd vastgesteld, met een tel-periode van 11,45 µsec, inderdaad met 1,5 vermenigvuldigd wordt.

adres 1B1FH

Deze routine meet hoe lang het duurt tot aan de volgende overgang van een cassette-input. Het "Cas Input"-bit wordt voortdurend gecontroleerd, via register 14 van de Programmeerbare geluidsgenerator, tot het een andere waarde krijgt dan in register E wordt aangegeven. Dan wordt de overeenkomstige vlag omgekeerd. De tijdsluur staat in register C, waarbij elke eenheid 11,45 µsec voorstelt.

adres 1B34H

Deze routine meet de lengte van een complete cyclus op cassette, tussen twee hoog/laag-overgangen. Via register 14 van de Programmeerbare geluidsgenerator wordt het "Cas Input"-bit bekeken, tot het nul wordt. De overgangsvlag in register E wordt op 0 gezet, en de tijd tot aan de volgende laag/hoog-overgang gemeten (1B23H). Vervolgens wordt de tijd gemeten tot aan de volgende hoog/laag-overgang (1B25H), en het totaal komt in register C.

adres 1B45H

naam OUTDLO
 in A - karakter dat verwerkt moet worden
 uit n.v.t.
 wijzigt EI

Deze standaardroutine wordt door de BASIC Interpreter gebruikt om een karakter naar het op dat moment actieve randapparaat te sturen. Voortaf wordt met de standaardroutine ISFLIO gecontroleerd of de output naar een I/O-buifter wordt geleid. Is dat zo, dan gaat de controle over naar de sekventiële output-verwerker (BCYBH), via de standaardroutine CALBAS. Indien PRIFLG nul is, wordt de controle overgedragen aan de standaardroutine CHPUF, om het karakter op het scherm te zetten. Als de printer actief is, wordt RAWPRT bekeken. Staat op die plaats een van nul verschillende waarde, dan wordt het karakter direct naar de printer gestuurd (1B4BH), zoiets wordt verdergegaan met de standaardroutine OUTDLP.

adres 1B53H

naam OUTDLP
 in A - karakter dat verwerkt moet worden
 uit n.v.t.
 wijzigt EI

Deze standaardroutine stuurt een karakter naar de printer. Is het karakter een IAB-code (05H) dan worden spaties gestuurd tot LPTPOS een vastvoud van 8 is (0,8,16, enz.). Is het karakter een CR-code (0DH) dan wordt LPTPOS nul gemaakt. Een andere controlecode leest LPTPOS omgekeerd, was het karakter een gewone schrijfbaar karakter, dan wordt LPTPOS opgehoogd.

Als NTRXSP nul is - wat betekent dat het een printer met NSK-specificaties is - dan wordt het karakter zonder meer aan de printer doorgegeven (1B4BH). Indien een gewone printer aangeloten is, wordt via de standaardroutine CNVCHR naar grafische karakters gezocht. Is het karakter een header (01H)

dan doet de routine verder niets meer. Is het een omgezet grafisch karakter, dan wordt dit door een spatie vervangen. Alle andere karakters worden naar de printer gestuurd (18MCH).

adres 1897H TABEL

Deze tabel van twintig bytes wordt door de toetsenbord-decoder gebruikt om bij een bepaald toets-nummer de bijpassende routine op te zoeken.

TOETSNUMMER	ADRES	FUNCTIE
00H tot 0FH	0FB3H	Rijden 0 tot 5
30H tot 32H	0F10H	SHIFT, CTRL, GRAPH
33H	0F36H	CAP
34H	0F10H	CODE
35H tot 39H	0FC3H	F1 tot F5
3AH tot 3BH	0F10H	ESC, TAB
3CH	0F46H	STOP
3DH tot 40H	0F10H	BS, CR, SEL, SPACE
41H	0F06H	HOME
42H tot 57H	0F10H	INS, DEL, CURSOR

adres 18ABH

Deze routine wordt door de standaardroutine OUTDLP gebruikt om een karakter naar de printer te sturen. Dit gebeurt via de standaardroutine LPTOUT. Als na LPTOUT vlag C gezet is, wordt de controle overgedragen aan de Device I/O error-generator (73BE2) via de standaardroutine CALBS.

adres 18BFH TABEL

De volgende 2 K bevat de karakterzet die de computer gebruikt bij het aanzetten. De eerste acht bytes bevatten het bitpatroon voor karaktercode 00H, de volgende acht het patroon voor karaktercode 01H en zo verder tot en met karaktercode FFH.

adres	23BFH
naam	PINLIN
in	n.v.t.
uit	HL-start van tekst, vlag C indien "CTRL/STOP"
wijzig	AF, BC, DE, HL, EI

Deze standaardroutine wordt door de Hoofdruis van de BASIC Interpreter gebruikt om een logische tekstregel die werd ingetypt, te lezen. De controle gaat over naar de standaardroutine INLIN na het punt waar de vorige lijn werd afgebroken (23E0H).

adres	23CCH
naam	0INLIN
in	n.v.t.
uit	HL-start van tekst, vlag C indien "CTRL/STOP"
wijzig	AF, BC, DE, HL, EI

Deze standaardroutine wordt door het "INPUT"-statement gebruikt, om een logische tekstregel van het scherm te lezen. De karakters

"7" worden via de standaardroutine OUTD0 op het scherm gezet, en de routine loopt door in de standaardroutine INLIN.

adres 23D5H

naam	INLIN
in	n.v.t.
uit	HL-start van tekst, vlag C indien "CTRL/STOP"
wijzig	AF, BC, DE, HL, EI

Deze standaardroutine wordt door het "LINE INPUT"-statement gebruikt om een logische tekstregel van het scherm te lezen. De karakters worden van het toetsenbord uitgelezen tot ofwel "CR" ofwel "CTRL/STOP" wordt ingedrukt. Vervolgens wordt de logische regel karakter per karakter van het scherm gelezen, en in de tekstbuffer BUF in het werkgebied gezet.

De laatste schermcodinaten worden vooraf uit CSRX en CSRY gehaald en in FSTPDS gezet. Het byte in LINITB dat overeenkomt met de regel boven de gelezen regel, wordt eerst niet-nul gemaakt (0C29H) om te voorkomen dat die regel logisch doorloopt tot op de uitgelezen regel.

Eik karakter dat wordt ingetypt en dat via de standaardroutine CHGET wordt uitgelezen, wordt (0919H) vergeleken met de tabel met edit-functies op 2439H. Dan wordt de controle aan een van de bijhorende routines overgedragen, ofwel aan de algemene verwerkingsroutine (23FFH), al naargelang, dit duurt het zo lang tot vlag C gezet wordt (door "CTRL/STOP" of "CR"). Dan wordt registerpaar HL geladen met het startadres van BUF en de routine stopt. Naar op dat vlag C gezet wordt, indien vlag Z gezet is; daarmee wordt een onderscheid gemaakt tussen een onderbreking door "CR" of een bevestigde "CTRL/STOP" en een gewone "CTRL/STOP".

adres 23FFH

Deze routine verwerkt alle karakters voor de standaardroutine INLIN op de editing-functies na. Is het karakter een "TAB"-code (09H) dan worden spaties gestuurd (23FFH) tot CSRX een vaalvoud van acht plus 1 is (kolommen 1,9,17,25,33). Wanneer het om een grafische header gaat, wordt die gewoon doorgegeven aan de standaardroutine OUTD0. Andere controlecodes onder 20H worden doorgegeven, waarna INSFLG en CSYLE op nul worden gezet. Gaat het om een schrijfbaar karakter, dan wordt eerst INSFLG gecontroleerd, en indien nodig wordt een spatie tussengevoegd (24F2H) voordat het karakter naar de standaardroutine OUTD0 gestuurd wordt.

adres 2439H TABEL

Deze tabel bevat de edit-codes, die door de standaardroutine INLIN herkend worden, gekoppeld aan het adres van een routine die ze verwerkt:

CODE NAAR FUNCTIE

 08H 2561H BS, 1 positie terug
 12H 24E5H INS, schakel INSERT in of uit
 18H 24FEH ESC, doet niets
 02H 260EH CTRL-B, naar einde van vorige woord
 06H 25FBH CTRL-F, naar begin van volgende woord
 05H 25D7H CTRL-N, naar eind van logische regel
 05H 25B9H CTRL-E, wis tot eind van de regel
 00H 24C5H CTRL-STOP, stop uitvoering
 15H 256EH CR, stop uitvoering
 7FH 2550H CTRL-U, wis regel
 7FH 2550H DEL, wis karakter

 adres 245AH
 Deze routine voert de "CR"-bewerking uit voor de standaardroutine INLIN. De startcoördinaten van de logische regel worden opgezocht (266CH) en de cursor van het scherm gehaald (082EH). Tot maximaal 254 karakters worden dan uit URAN gelezen (080BH) en in BUF gezet. Eventuele nul-codes (00H) worden genegeerd, en alle karakters met een code lager dan 20H worden vervangen door de combinatie van een grafische header (01H) en de code + 40H. Op het einde van elke schermregel wordt in LINITB gekeken (0C1DH) of de logische regel verder gaat op de volgende schermregel. Eventuele achteropkomende spaties worden uit BUF gehaald, en op het einde wordt een nul gezet, als markteken. De cursor wordt terug op het scherm gezet (0851H) en de cursorcoördinaten worden ingewild met de waarde voor de laatste schermregel die deel uitmaakt van de logische regel; dit gebeurt via de standaardroutine POSIT. Vervolgens wordt een "LF" naar de standaardroutine DUIDD gestuurd, wordt INSFLG op nul gezet en de routine stopt met een "CR" in register A (00H), en vlaggen NZ en C. Deze "CR"-code wordt door de hoofdlus van de standaardroutine INLIN naar het scherm gezet net voor het einde ervan.

 adres 245AH
 Deze routine voert de "CTRL/STOP"-bewerking uit voor de standaardroutine INLIN. De laatste schermregel die nog bij de logische regel hoort, wordt opgezocht door in LINITB te kijken (0C1DH); CSTYLE wordt op 0 gezet, bij het begin van BUF wordt een nul gezet en alle muziek-variabelen worden gewist via de standaardroutine GICINI. Nu wordt gekeken in TRPBL of er een "ON STOP"-statement in werking is. Is dat zo, dan wordt de cursor gereset (246FH) en de routine stopt met vlaggen NZ en C. B65RDH wordt gecontroleerd op bevalligde ROM. Is die controle positief, dan wordt de cursor gereset (244FH) en de routine stopt met vlaggen NZ en C. Is dat niet zo, dan wordt de cursor gereset en de routine stopt met vlaggen Z en C.

 adres 24E5H
 Deze routine voert de "INS"-bewerking uit voor de standaardroutine INLIN. De toestand van INSFLG op dat ogenblik wordt omgekeerd en de controle gaat over op de standaardroutine CSTYLE (24E6H).

 adres 24F2H

 Deze routine wordt gebruikt door het stuk van de standaardroutine INLIN dat zich niet met speciale edit-toetsen ophoudt, om een spatie tussen te vogen. De cursor wordt weggehaald (0A2EH) en uit CSRY en CSRY worden de coördinaten gehaald. Het karakter dat op die plaats staat wordt uit URAN gehaald en door een spatie vervangen (085EH). De opeenvolgende karakters worden dan 1 kolom naar rechts gecopieerd, tot het einde van een scherm-regel.

 Op dit punt wordt LINITB gecontroleerd (0C1DH) om te zien of de logische regel verder gaat. Is dat zo, dan gaat de bewerking op de volgende regel verder. Zoniet, wordt het karakter dat op de laatste kolom stond, bekeken. Als het een spatie was, stopt de routine nadat ze de cursor opnieuw op het scherm heeft gezet (0851H). Was het een ander karakter, dan wordt de overeenkomstige plaats in LINITB op 0 gezet, om aan te geven dat die logische regel verder loopt. Het nummer van de volgende schermregel wordt vergeleken met het aantal regels op het scherm (0C32H). Is die regel de laatste, dan wordt het scherm 1 regel naar boven gescrolld (0A8BH). Was dat niet zo, dan wordt een blanco regel ingevoegd (0A87H) en het copieren gaat verder.

 adres 2550H
 Deze routine voert de "DEL"-bewerking uit voor de standaardroutine INLIN. Als de cursorpositie de meest rechts kolom aangeeft, en de logische regel niet verder loopt, dan wordt enkel een spatie naar URAN geschreven (2595H). Is dat niet zo, dan wordt een "RECHTS"-code (1CH) naar de standaardroutine DUIDD gestuurd en de controle gaat naar de "BS"-routine.

 adres 2561H
 Deze routine voert de "BS"-bewerking uit voor de standaardroutine INLIN. Eerst wordt de cursor weggehaald (0A2EH) en de kolomcoördinaat van de cursor met 1 verlaagd, tenzij hij uiterst links stond, en de vorige regel niet doorliep. Vervolgens worden karakters uit URAN gelezen (080BH) en aan positie naar links teruggeschreven (085EH) tot op het einde van de logische regel. Dan wordt een spatie naar URAN geschreven (085EH) en de cursor wordt terug op het scherm gezet (0851H).

 adres 25A6H
 Deze routine voert de "CTRL--U"-bewerking uit voor de standaardroutine INLINE. De cursor wordt weggehaald (0A2EH) en de start van de logische regel wordt opgezocht (266CH). Die coördinaten worden in CSRY en CSRY gezet, en de hele logische regel wordt gewist (25BEH).

 adres 25B9H
 Deze routine voert de "CTRL--E"-bewerking uit voor de standaardroutine INLINE. De cursor wordt weggehaald (0A2EH) en de rest van de schermregel gewist (085EH). Dit gebeurt zowaar in LINITB gecontroleerd (0C1DH). Daarna wordt de cursor teruggezet (09E1H), INSFLG wordt op 0 gezet en in CSTYLE wordt

de code voor "blokvormige cursor" gezet (242DH).

adres 25F8H

Deze routine voert de "CTRL"-F"-bewerking uit voor de standaardroutine INLINE. De cursor wordt weggehaald (0A2EH) en niet zo vaak naar rechts bewogen (2624H) tot een niet alfanumeriek karakter gevonden wordt. Daarna wordt hij weer naar rechts verzet (2624H) tot er een alfanumeriek karakter gevonden wordt. Nu wordt de cursor terug op het scherm gezet (25C0H) en de routine stopt.

adres 260EH

Deze routine voert de "CTRL"-B"-bewerking uit voor de standaardroutine INLINE. De cursor wordt weggehaald (0A2EH) en daarna zo vaak naar links bewogen (2534H) tot een alfanumeriek karakter gevonden wordt. Daarna wordt hij verder naar rechts geschoven tot een niet alfanumeriek karakter gevonden is, op den 1 positie naar rechts (0A5BH). De cursor wordt weer op het scherm gezet (25C0H) en de routine stopt.

adres 2624H

Deze routine verzet de cursor één positie naar rechts (0A5BH), leedt dan register D met het nummer van de uiterst rechtse kolom en register E met het nummer van de onderste regel op het scherm, en controleert dan of op de cursor-positie een alfanumeriek karakter staat (263DH).

adres 2634H

Deze routine verzet de cursor één positie naar links (0A4CH), leedt register D met het nummer van de uiterst linkse kolom en register E met het nummer van de bovenste regel op het scherm. De coördinaten van de cursor worden met deze waarden vergeleken en de routine stopt met vlag Z indien ze gelijk zijn. In het andere geval wordt het karakter op die plaats uit URAH gelazen (0BDBH) en er wordt gecontroleerd of het een alfanumeriek karakter is. Zo ja, worden de vlaggen NZ en C gezet; zo nee, worden de vlaggen NZ en NC gezet en de routine stopt.

Alfanumerieke karakters zijn: de cijfers "0" tot "9", de letters "A" tot "Z" en de letters "a" tot "z". Daar worden ook de grafische karakters BGH tot 9FH en AGH tot FXH bijgenomen: dat waren oorspronkelijk Japanse lettertekens. Die hadden eigenlijk uitgesloten moeten worden, toen de ROM voor Europe werd aangepast.

adres 266CH

Deze routine zoekt de start op van een logische regel, en zet de scherm-coördinaten ervan in registerpaar HL. Elke scherregel boven de laatste wordt via LINITB gecontroleerd (0C1DH) tot een regel wordt gevonden die logisch op die scherregel eindigt. De regel die op het scherm onmiddellijk daaronder ligt, is het begin van de logische regel, en het regelnummer ervan wordt in register L gezet. Dit nummer wordt vergeleken met de inhoud van FSIF05, waar het regelnummer in staat op het openlijk dat de standaardroutine INLIN begon te werken, om te zien of de cursor

nog op dezelfde regel staat. Als dat zo is, wordt de kolom-coördinaat in register H op de oorspronkelijke positie (uit FSIF05) gezet. Was het niet dezelfde regel, dan wordt register H ingevuld met de uiterst linkse positie, zodat de hele regel gelazen kan worden.

adres 2680H

Op dit adres staat een sprong-instructie naar de Initialisatie-routine bij het inschakelen van de machine (7C76H).

adres 2683H

Op dit adres staat een sprong-instructie naar de standaard-routine SWCHR (558CH).

adres 2686H

Op dit adres staat een sprong-instructie naar de standaard-routine CHRSTR (4665H).

adres 2689H

Op dit adres staat een sprong-instructie naar de standaard-routine GETYPR (5597H).