BEELDOPBOUW BIJ DE GRAFISCHE SCHERMEN.
Adressen en VPOKE-instructies.

INLEIDING.

Vooral voor hen die wel eens 'VPOKEn' of gebruik maken van machinetaal is het noodzakelijk om te weten hoe de VDP funktioneert en welke adressen we waar kunnen vinden. Daarom wil ik deze keer eens met U gaan snuffelen in de VDP en daarbij dan speciale aandacht schenken aan de geheugenstructuur bij de grafische schermen. We zullen daarbij dan kijken waar we alle tabellen kunnen vinden, hoe uiteindelijk het beeld tot stand komt en hoe we dat kunnen beinvloeden met VPOKE-instructies.

Tabel 1. MSX 2 VDP ADRESSEN.

SCREEN:	2	4	5	6	7	8
Aant.kleuren:	16	16	16	4	16	256
Beeldlijnen:	192	192	212	212	212	212
Pixels/lijn:	256	256	256	512	512	256
NAAMTABEL						
BASE:	10	20	25	30	35	40
Van adres:	&H1800	&H1800	&H0000	&H0000	&H0000	&H0000
tot adres:	&H1B00	&H1B00	&H6A00	&H6A00	&HD400	&HD400
Aantal bytes:	24*32	24*32	212*128	212*128	212*256	212*256
PATROONTABEL						
BASE:	12	22				
Van adres:	&H0000	&H0000				
tot adres:	&H1800	&H0000				
Aantal bytes:	6142	6142				
KLEURTABEL						
BASE:	11	21				
Van adres:	&H2000	&H2000				
tot adres:	&H3800	&H3800				
Aantal bytes:	6142	6142				
SPRITE ATTR.						
BASE:	13	23	28	33	38	43
Van adres:	&H1B00	&H1E00	&H7600	&H7600	&HFA00	&HFA00
tot adres:	&H1B80	&H1E80	&H7680	&H7680	&HFA80	&HFA80
Aantal bytes:	32*4	32*4	32*4	32*4	32*4	32*4
SPRITE PATR.						
BASE:	14	24	29	34	39	44
Van adres:	&H3800	&H3800	&H7800	&H7800	&HF000	&HF000
tot adres:	&H4000	&H4000	&H8000	&H8000	&HF800	&HF800
Aantal bytes:	256*8	256*8	256*8	256*8	256*8	256*8
of:	64*32	64*32	64*32	64*32	64*32	64*32
SPRITE COLOR						
Van adres:		&H1C00	&H7400	&H7400	&HF800	&HF800
tot adres:		&H1E00	&H7600	&H7600	&HFA00	&HFA00
Aantal bytes:		32*16	32*16	32*16	32*16	32*16
COLOR INTENS.						
Page:	-	-	0	0	0	0
Van adres:	&H1B80	&H1B80	&H7680	&H7680	&HFA80	&HFA80
tot adres:	&H1BA0	&H1BA0	&H76A0	&H76A0	&HFAA0	&HFAA0
Aantal bytes:	16*2	16*2	16*2	16*2	16*2	16*2

In tabel 1 is alvast een overzicht gegeven van de adressen van de diverse tabellen welke gebruikt worden bij SCREEN 2-8 met uitzondering van screen 3. Dit laatste scherm vindt weinig toepassing en zo het wel wordt toegepast, is er daarbij weinig behoefte aan VPOKE-instructies of de toepassing van machinetaalroutines.

SCREEN 2 en 4 (MSX 1 en 2)

Bij screen 2 wordt het beeld op dezelfde manier opgebouwd als bij screen 4. Het enige verschil is dat bij screen 4 nu 8 sprites naast elkaar kunnen worden geplaatst tegen slechts 4 bij screen 2. Wat ik nu over screen 2 ga vertellen geldt dus ook voor screen 4. De MSX 2-ers mogen daarom alvast meelezen.
De NAAMTABEL bestaat uit 768 bytes welke zijn onderverdeeld in 3 blokken. Per blok hebben de bytes een inhoud welke oploopt van 0 tot 255, welke laatste dan de maximale waarde is welke een byte kan bevatten. Vandaar dan ook de onderverdeling in 3 blokken (3 maal 256 maakt 768). De plaats van een byte in een blok komt overeen met de plaats op het scherm. Normaal komt trouwens bij de initialisatie van screen 2 ook de inhoud van de bytes overeen met de plaats op het scherm, te weten 0 - 31 voor de bovenste rij, 32 - 63 voor de tweede rij enz. Alleen met een VPOKE-instructie kunnen we deze volgorde veranderen en dat kan wel eens handig zijn, maar daarover later. Elke byte (naam) in de naamtabel correspondeert weer met een set van 8 bytes in de PATROONTABEL waarin, zoals de naam al aangeeft, het beeldpatroon is vastgelegd. Tevens correspondeert die byte dan ook nog met een set van 8 bytes in de KLEURTABEL waarin de kleuren van het patroon zijn vastgelegd (zie ook fig.1).

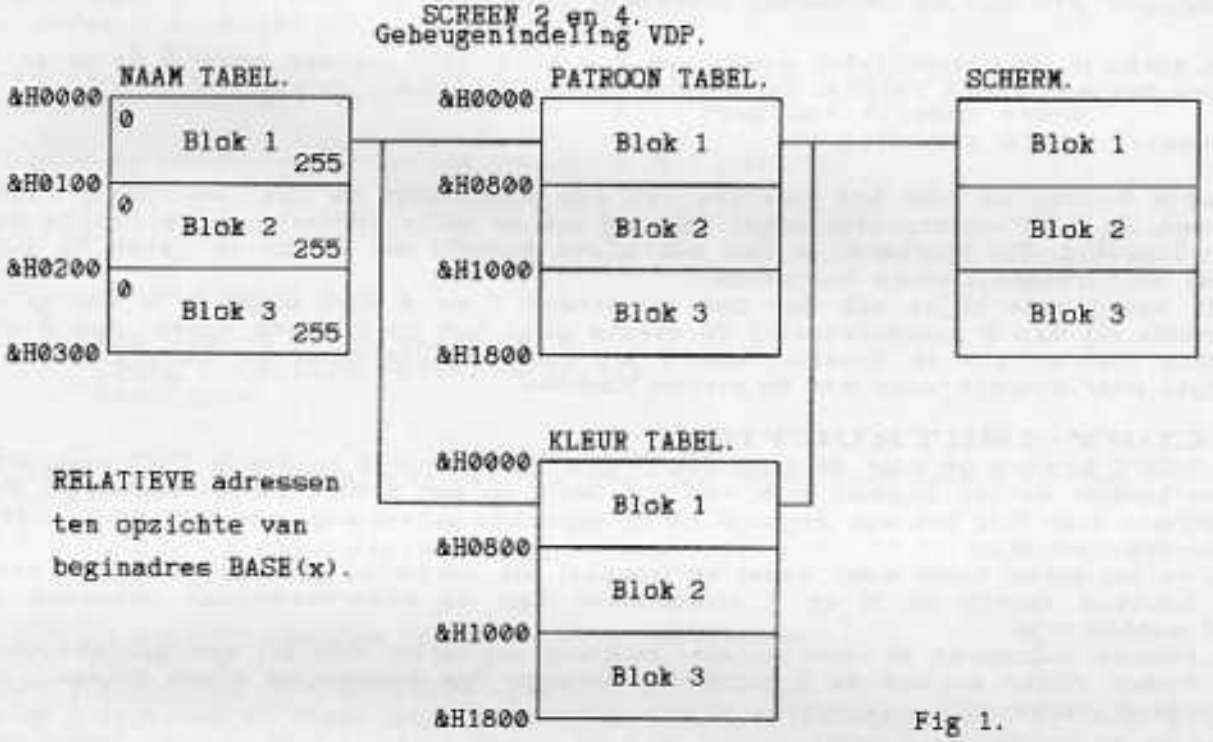


Fig 1.

BEGINADRESSEN:

	NAAMTABEL	PATROONTABEL	KLEURTABEL
SCREEN 2:	BASE(10)	BASE(12)	BASE(11)
SCREEN 4:	BASE(20)	BASE(22)	BASE(21)

Het patroon in elk van de 768 vakjes op het beeldscherm is dus opgebouwd uit 8 bytes, b.v.:

```
byte 0:  0 0 0 1 1 1 0 0
byte 1:  0 0 1 0 0 0 1 0
byte 2:  0 1 0 0 0 0 0 1  enz.
```

Elke bit (0 of 1) komt nu overeen met een PIXEL op het beeldscherm. Een 1 wil zeggen dat de pixel de VOORGRONDkleur krijgt en een 0 geeft aan dat die pixel de ACHTERGRONDkleur krijgt.
In de kleurtabel wordt dan in hetzelfde relatieve adres (= volgnr. van het adres t.o.v. het beginadres van de tabel) per byte de voor- en achtergrondkleur vastgelegd. De inhoud van zo'n kleurbyte is samengesteld uit:

16 * het voorgrond kleurnr. + achtergrond kleurnr.

Als het patroon rood is (kleur 9) bij een zwarte (kleur 1) achtergrond dan wordt de kleurbyte dus: 16 * 9 + 1 = 145. Handiger is het om hierbij met de HEX-code te werken. Bij de 1-byte HEX-code staat dan het eerste karakter voor de voorgrondkleur en het tweede voor de achtergrondkleur. Bij hetzelfde kleurvoorbeeld wordt dat dan &H91.

ADRES BEREKENING.

Als we willen weten op welke adressen de gegevens voor een bepaalde pixel zijn opgeslagen (X-Y coordinaten gegeven) dan berekenen we eerst RA (relatieve adres) met de formule:

```
RA = (Y\8)*256 + (X\8)*8 + YMOD8
```

Bij screen 2 vinden we dan:

```
Adres in patroontabel  = BASE(12)+RA
Adres in kleurtabel    = BASE(11)+RA
```

De 8 bits van de byte uit de patroontabel staan voor een horizontale rij van 8 pixels. Wanneer we deze pixels nu een nummer geven overeenkomstig het bitnr.(7-6-5-4-3-2-1-0) dan zal onze pixel zich bevinden op positie 7-XMOD8.

REKENVOORBEELD.

Stel, we willen op een beeld met achtergrondkleur zwart (1) een witte punt (15) zetten op de coordinaten: X=100, Y=50.
Het relatieve adres wordt dan:

```
RA = (50\8)*256+(x\8)*8+YMOD8 = &H600
```

Het patroon bevindt zich dan op adres:

```
BASE(12)+RA = &H0000 + &H600 = &H600
```

Met de pixel op positie:

```
7-100MOD8 = 3
```

Op ons adres in de patroontabel moeten we dus &B00001000 inpoken ofwel 8 decimaal. We doen dat dan met VPOKE &H600,8. Vervolgens poken we de kleur in met:

```
VPOKE BASE(11)+RA,&HF1
```

ofwel:

```
VPOKE &H2600,&HF1
```

Natuurlijk hadden we voor het plaatsen van een punt beter de PSET-instructie kunnen toepassen, de BASIC-interpreter zoekt dan wel uit op welke adressen welke waarde moet worden ingevuld. Dit voorbeeld is dan ook alleen bedoeld om U aan te geven op welke wijze we de adressen kunnen berekenen.
Uit het voorgaande blijkt ook dat het bij screen 2 en 4 niet mogelijk is om op een horizontale rij van 8 pixels (waarbij de eerste pixel ligt op X = n*8) meer dan 2 verschillende kleuren toe te passen. Immers bij een volgende pixel op die rij wordt de kleurbyte weer overschreven met de nieuwe kleuren.

DE COPY-INSTRUCTIE.

Bij de MSX 2 kennen we voor de schermen 5 t/m 8 de handige en snelle COPY-instructie. Hiermee kunnen we een bepaald blok van het beeld op een andere plaats copieeren. Voor de schermen 2 en 4 is het ook mogelijk om op beperkte schaal snel een blok te copieeren. Deze beperkingen zijn:
1) Het te copieeren beeld moet zowel horizontaal als vertikaal uit een meervoud van 8 pixels bestaan terwijl de X en Y coordinaten van de linkerbovenhoek eveneens een 8-voud moeten zijn.
2) We kunnen onbeperkt in -horizontale- richting copieeren, dus bij een gelijkblijvende Y-coordinaat. Alleen als ook de Y-coordinaat wijzigt, dan kunnen we alleen binnen 1 van de 3 blokken copieeren.

EERSTE BLOK NAAMTABEL GEVULD MET OORSPRONKELIJKE NAMEN. SCREEN 2 en 4.

Y=	X= 0								64								136								200							
0	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F
8	20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F	30	31	32	33	34	35	36	37	38	39	3A	3B	3C	3D	3E	3F
16	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	51	52	53	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F
24	60	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F	70	71	72	73	74	75	76	77	78	79	7A	7B	7C	7D	7E	7F
32	80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F	90	91	92	93	94	95	96	97	98	99	9A	9B	9C	9D	9E	9F
40	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	AA	AB	AC	AD	AE	AF	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	BA	BB	BC	BD	BE	BF
48	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	DA	DB	DC	DD	DE	DF
56	E0	E1	E2	E3	E4	E5	E6	E7	E8	E9	EA	EB	EC	ED	EE	EF	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	FA	FB	FC	FD	FE	FF
64																																

EERSTE BLOK NAAMTABEL NA COPY-INSTRUCTIE: (0,0)-(63,39) TO (0,136)

Y=	X= 0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	64 00	01	02	03	04	05	06	07	136 19	1A	1B	1C	1D	1E	1F	200
0	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	00	01	02	03	04	05	06	07	19	1A	1B	1C	1D	1E	1F	
8	20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F	30	20	21	22	23	24	25	26	27	39	3A	3B	3C	3D	3E	3F	
16	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	40	41	42	43	44	45	46	47	59	5A	5B	5C	5D	5E	5F	
24	60	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F	70	60	61	62	63	64	65	66	67	79	7A	7B	7C	7D	7E	7F	
32	80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F	90	80	81	82	83	84	85	86	87	99	9A	9B	9C	9D	9E	9F	
40	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	AA	AB	AC	AD	AE	AF	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	BA	BB	BC	BD	BE	BF	
48	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF	D0	01	D2	D3	D4	D5	D6	D7	D8	D9	DA	DB	DC	DD	DE	DF	
56	E0	E1	E2	E3	E4	E5	E6	E7	E8	E9	EA	EB	EC	ED	EE	EF	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	FA	FB	FC	FD	FE	FF	
64																																	

Fig.2

Stel dat we bij SCREEN 2 een blok dat wordt begrenst door X1=0, Y1=0, X2=63, Y2=39 willen copieeren naar X3=136, Y3=0 (zie ook fig.2). Allereerst berekenen we dan het 'van' en het 'naar' adres in de NAAMTABEL:

```
VA-adres = BASE(10)+(Y0\8)*32+X0\8 = &H1800+&H0 = &H1800
NA-adres = BASE(10)+(Y3\8)*32+X3\8 = &h1800+&HF1 = &H1817
```

Vervolgens berekenen we het aantal stappen in X-Y richting:

```
DX = (X2-X1)\8 = (63-0)\8 = 7
DY = (Y2-Y1)\8 = (39-0)\8 = 4
```

De COPY-instructie wordt dan:

```
FOR Y=0 TO 4:FOR X=0 TO 7
VPOKE NA+32*Y+X,VPEEK(VA+32*Y+X)
NEXT:NEXT
```

Met onderstaande instructies kan het hele beeld N*8 pixels naar rechts worden verschoven:

```
FOR X=31 TO N STEP-1:FOR Y=0 TO 23
VPOKE &H1800+Y*32+X,VPEEK(&H1800+Y*32+X-N)
NEXT:NEXT
```

Voor SCREEN 4 gelden dezelfde berekeningen en instructies.

SPRITES in SCREEN 2.

In tabel 1 zijn ook adressen gegeven voor de COLOR INTENSiteit. Deze toevoeging geldt echter alleen voor de MSX 2. Later komen we hier nog op terug wanneer we de schermen 4 t/m 8 gaan behandelen.
Bij de MSX 1 zijn alleen een SPRITE ATTRIBUTEN-TABEL en de SPRITE PATROON-TABEL aanwezig.

SPRITE ATTRIBUTEN-TABEL.

In deze tabel worden de attributen opgeslagen voor 32 sprites (is maximum aantal dat kan worden geplaatst) met de rangnummers 0 t/m 31. Per sprite worden dan de volgende attributen vastgelegd:

```
Relatieve adres 0: Y-coordinaat.
Relatieve adres 1: X-coordinaat.
Relatieve adres 2: Patroonnummer van de sprite.
Relatieve adres 3: kleurnummer van de sprite.
```

Willen we b.v. met een VPOKE-instructie de X-coordinaat van sprite N wijzigen, dan vinden we het betreffende adres uit:

```
ADRES = BASE(13) + N * RA    waarin RA = relatieve adres.
```

SPRITE PATROONTABEL.

In deze tabel van 1248 bytes kunnen de patronen worden vastgelegd van 256 kleine sprites (8*8 pixels) of 64 grote sprites (16*16) pixels. Bij de SCREEN-instructie wordt reeds de grootte van de sprites gedefinieerd: SCREEN 2,0 kl. sprites, SCREEN 2,1 kl. sprites vergroot, SCREEN 2,2 gr. sprites en SCREEN 2,3 gr. sprites vergroot. Bij de uitvoering van deze instructie wordt de patroontabel dan verdeeld in 256 blokken van 8 bytes ofwel 64 blokken van 32 bytes.
Het eerste adres van patroon P vinden we met:

```
ADRES = BASE(14) + P * 8     voor kleine sprites of
ADRES = BASE(14) + P * 32    voor grote sprites.
```

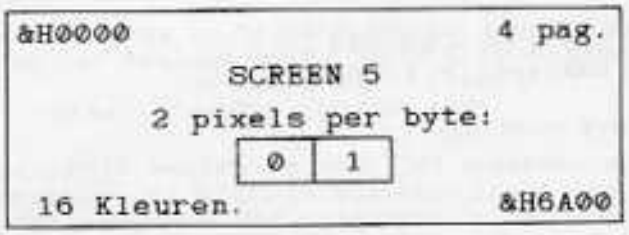
SCREEN 5 t/m 8 (MSX 2)

VOORNAAMSTE KENMERKEN VAN DEZE SCHERMEN (zie ook fig. 3):
Beeldhoogte 212 pixels i.p.v. 192./ Per pixel andere kleur mogelijk./ Meerdere pagina's te programmeren.
Alle gegevens voor de beeldopbouw zijn in de NAAM-tabel samengevat, dus geen aparte patroon- en kleurtabel.(zie ook fig. 3). Het beeld wordt nu ook niet meer blok voor blok opgebouwd, doch regel (beeldlijn) voor regel.
Per byte wordt, afhankelijk van het schermtype, aan 1 of meerdere pixels een bepaalde kleur toegekend. Deze manier van beeldopbouw maakt het onder andere mogelijk om bij de MSX 2 de snelle COPY-instructie toe te passen.
Voorts zijn er dan nog de verruimde COLOR- en SPRITE- instructies.

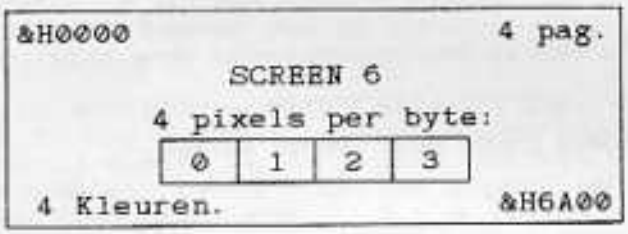
AANTAL PAGINA'S.

Bij deze schermen zijn meerdere pagina's mogelijk. In tabel 1 zijn de tabellen gegeven voor pagina 0. De andere pagina's beschikken over exact dezelfde tabellen en hebben dezelfde adressen. Met de SET PAGE-instructie bepalen we naar welke pagina wordt geschreven (actieve pagina) en welke pagina in beeld is.

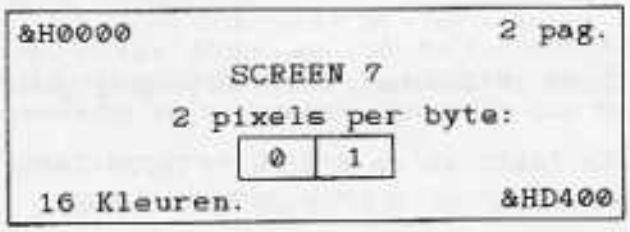
Fig 3. NAAMTABEL SCREEN 5 t.m.8.



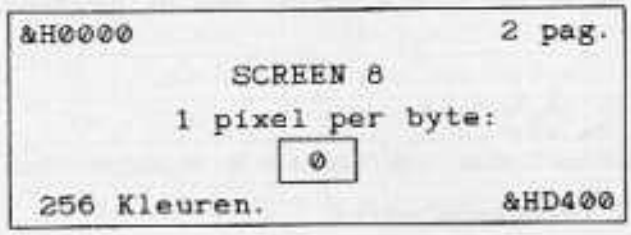
Breed: 256 pixels / 128 bytes
Hoog : 212 pixels / 212 bytes
Naamtabel 27136 bytes lang
Adresberekening:
ADR = Y*128 + X\2
Pixelpositie: Xmod2



Breed: 512 pixels / 128 bytes
Hoog : 212 pixels / 212 bytes
Naamtabel 27136 bytes lang
Adresberekening:
ADR = Y*128 + X\4
Pixelpositie: Xmod4



Breed: 512 pixels / 256 bytes
Hoog : 212 pixels / 212 bytes
Naamtabel 54272 bytes lang
Adresberekening:
ADR = Y*128 + X\2
Pixelpositie: Xmod2



Breed: 256 pixels / 256 bytes
Hoog : 212 pixels / 212 bytes
Naamtabel 54272 bytes lang
Adresberekening:
ADR = Y*256 + X
Pixelpositie = adres.

SPRITES ALGEMEEN.

Voor alle schermen 5 t/m 8 gelden dezelfde condities voor wat betreft aantal sprites, indeling van de attributen- en patroontabellen, sprite-INSTRUCTIES e.d. Alleen is er wat verschil in de kleuren bij de diverse schermen. Per schermtype zullen we daar dan apart aandacht aan schenken.
Het berekenen van de adressen in de ATTRIBUTEN- en PATROON tabellen werd al eerder behandeld bij screen 2. Alleen het BASE adres moet daarin dan worden aangepast. Deze adressen kunt U vinden in tabel 1.

Omdat het bij de schermen 4 t/m 8 mogelijk is om een sprite lijn voor lijn een andere kleur te geven (SPRITE COLOR$- instructie), is er nu voor de sprites een aparte kleurtabel aanwezig, welke we de SPRITE COLOR-tabel noemen. De kleurcode op het vierde adres in de ATTRIBUTENtabel heeft daardoor geen functie meer (deze bytes hebben nu allen 15 als inhoud. Inpoken van een andere waarde heeft geen enkele invloed).

De SPRITE COLOR-tabellen hebben geen nummer zodat het beginadres niet is op te vragen met de instructie PRINT BASE(X). In tabel 1 zijn de adressen opgenomen, zoals die gelden voor de Philips VG 8235; hopelijk zitten ze op Uw computer op dezelfde plaats.
De tabel bestaat uit 32 blokken van 16 bytes. In elk blok wordt lijn voor lijn aangegeven welke kleur de sprite moet hebben. Bij kleine sprites worden alleen de eerste 8 bytes per blok benut. bij grote sprites (16*16 pixels) worden alle 16 bytes gebruikt. Wanneer we het beginadres van deze tabel BASE(C) noemen. dan vinden we het eerste adres van een blok uit:

Beginadres = BASE(C) + 32*n

waarin 'n' dan het spritenummer is. Let wel, deze colortabel is dus gebonden aan het spritenummer en niet aan het sprite patroon!
Met de instructie: COLOR SPRITE$(X) = X$ kunnen we de diverse beeldlijnen van de sprite een andere kleur geven. Die X$ is dan een string met 8 of 16 kleurnummers in de vorm van b.v.: CHR$(2) + CHR$(5) + CHR$(9) enz. Voor een grote sprite wordt dat dan een hele verzameling. Willen we slechts 1 lijn wijzigen, dan is het eenvoudiger om dat te doen met de VPOKE-instructie.
Om een sprite bij screen 5 t/m 8 buiten beeld te krijgen, moet Y liggen tussen 211 en 223. Echter niet op 216, want deze waarde is gereserveerd voor de 'verdwijntruuk'. Bij Y=216 verdwijnen namelijk alle sprites met hoger rangnummer ook uit het beeld. Bij screen 2 en 4 is dat het geval bij Y=208.

SPOOKRIJDER.

Onder bepaalde omstandigheden kan het gebeuren dat er naast de geprogrammeerde sprite nog een tweede sprite in beeld verschijnt. Deze sprite bestaat dan uit een wit vierkant en ligt 32 pixels naar links. Om dit te voorkomen dienen in het begin van het programma eerst alle sprites per pagina te worden gewist. We kunnen dat doen via de BIOS routine op adres &H69, hiertoe nemen we in het begin van het programma dan de volgende regels op:

```
DEFUSR0=&H69
FOR I=0 TO n: SET PAGE 0, n: R=USR0(0): CLS: NEXT
```

Hierin is 'n' dan het aantal pagina's van het betreffende scherm.

COLOR INTENSITEIT.

Bij de MSX 2 is voor alle schermen ook een tabel toegevoegd met de colorintensiteit voor de 16 kleurpalet nummers. Deze tabel omvat 16*2 bytes. Per 2 bytes wordt daar de intensiteit van de drie basiskleuren opgeslagen (zie ook tabel 2).

Tabel 2.

COLOR INTENSITEITS TABEL. SCREEN 0 t.m. 8						AFWIJKENDE SPRITE COLOR in in SCREEN 8				
Kleur nr.	Intens. R	G	B	Inhoud byte: 1	2	Kleurnr. 1)	2)	Intens. R	G	B	Kleur:
0	0	0	0	&H00	&H00	0	0	0	0	0	transparant
1	0	0	0	&H00	&H00	1	1	0	0	2	d.blauw
2	1	6	1	&H11	&H01	2	12	3	0	0	d.rood
3	3	7	3	&H33	&H07	3	13	3	0	2	d.paars
4	1	1	7	&H17	&H01	4	96	0	3	0	d.groen
5	2	3	7	&H27	&H03	5	97	0	3	2	d.cyaan
6	5	1	1	&H51	&H01	6	108	3	3	0	d.geel
7	2	6	7	&H27	&H06	7	109	3	3	2	grijs
8	7	1	1	&H71	&H01	8	157	7	4	2	oranje/geel
9	7	3	3	&H73	&H03	9	3	0	0	6	blauw
10	6	6	1	&H61	&H06	10	28	7	0	0	rood
11	6	6	4	&H64	&H06	11	31	7	0	6	paars
12	1	4	1	&H11	&H04	12	224	0	7	0	groen
13	6	2	5	&H65	&H02	13	227	0	7	6	cyaan
14	5	5	5	&H55	&H05	14	252	7	7	0	geel
15	7	7	7	&H77	&H07	15	255	7	7	6	wit

1) Kleurnummer voor de sprites.
2) Werkelijke kleur volges de kleurcode: 4*R + 32*G + B\2.

Deze tabel maakt het mogelijk om na een kleurwijziging met de COLOR=(c,R,G,B) instructie ook deze aangepaste kleuren mee te nemen wanneer we een bepaald beeld willen saven.
#) Zouden we dat niet doen, dan krijgt dat beeld later na het weer inladen de normale standaardkleuren.
Na het inladen van de COLOR INTENS tabel moet dan nog wel de instructie: COLOR=RESTORE worden gegeven, want pas dan neemt de processor deze waarden over. Ook als we in deze tabel een kleur veranderen met een VPOKE instructie, moeten we daarna de COLOR=RESTORE instructie geven. Alleen als we de kleur veranderen met de COLOR=(c,R,G,B) instructie, dan vindt deze restore automatisch plaats.
#) We kunnen alleen de -actieve- pagina of een deel daarvan saven met de instructie:
BSAVE"<dev>:naam",<beginadres>,<eindadres>,S

SCREEN 4 (1 pag.).

BEELDOPBOUW.
Bij dit scherm met 16 verschillende kleuren, vindt de opbouw van het beeld op dezelfde wijze plaats als bij het eerder behandelde SCREEN 2. Alleen biedt dit scherm nu wat meer mogelijkheden v.w.b. de COLOR en SPRITE instructies, zoals die hierboven reeds werden besproken.

SPRITES.
Dezelfde condities als bij screen 2, doch nu met de mogelijkheid om 8 sprites in 1 horizontale lijn te plaatsen.

TOEPASSING.
Voornamelijk computerspelletjes. Daar we echter naast screen 4 ook altijd over screen 5 beschikken,zal dit laatste scherm veelal de voorkeur verdienen gezien de vele extra mogelijkheden.

SCREEN 5 (4 pag.).

BEELDOPBOUW.
Bij dit scherm zijn 16 verschillende kleuren mogelijk. Per byte kunnen 2 pixels worden ingekleurd. Bij de HEX-code geeft dan het eerste karakter de kleur van de eerste (linkse) pixel en het tweede karakter de kleur van de tweede (rechtse) pixel aan, b.v.: &HF4 : pixel 0 is wit, pixel 1 is donkerblauw. In fig.3 kunt U zien op welk adres welke pixel is te vinden.
Met de COLOR = (c,R,G,B)-instructie kan uiteraard elke kleur weer naar believen worden gewijzigd.

SPRITES.
Zie SPRITES ALGEMEEN.

TOEPASSING.
Speciaal geschikt voor computerspelletjes met daarbij de mogelijkheid van een snelle beeldwisseling (4 pag.).

SCREEN 6 (4 pag.).

BEELDOPBOUW.
Bij dit scherm is de schermbreedte 512 pixels, dus dubbel zoveel als bij screen 5. Om nu toch binnen de beschikbare geheugenruimte ook hierbij over 4 pagina's te kunnen beschikken, is er wat op de kleuren bezuinigd. Daardoor zijn hier maar 4 verschillende kleuren mogelijk, namelijk de nummers 0, 1, 2 en 3.
Om hier de opbouw te begrijpen, moeten we de inhoud van een byte in BIN-code geven, verdeeld in 4 groepen b.v.:

```
&B 1 0   1 1   0 0   0 1
```

Deze groepen staan hier dan voor achtereenvolgens de kleuren 2, 3, 0 en 1. In HEX-code wordt dit dan &HB1. Voor hen die liever met decimale getallen werken:
De inhoud van een byte is gelijk aan 64 maal de eerste kleur + 16 maal de tweede kleur + 4 maal de derde kleur + de laatste kleur.

Nu zijn we bij de MSX2 gelukkig niet aan de standaardkleur gebonden en kunnen we met de instructie COLOR=(c,R,G,B) elke gewenste kleur aan de nummers 0 t/m 3 toekennen. We mogen ook zonder bezwaar de kleur een nummer geven, dat hoger is dan 3; de uiteindelijke kleur wordt dan toch MOD4 van dat nummer, dus ligt dan weer in de range 0-3. Daar kleur 0 een transparante kleur is, kunnen we daar moeilijk mee uit de voeten en houden we in principe maar 3 'echte' kleuren over.
Met de volgende ingreep kunnen we daar echter verandering in brengen:

```
VDP(9)  =  VDP(9) OR &H20
```

Kleur 0 is nu met de COLOR=(c,R,G,B) instructie op elke gewenste kleur te brengen. Deze ingreep kan later weer ongedaan worden gemaakt met de instructie:

```
VPD(9)  =  VDP(9) AND &HDF
```

of door het resetten van de computer.
Bij dit scherm zijn dus alleen de eerste 8 bytes (= 4 kleuren) van de COLOR INTENS-tabel van belang.

SPRITES.

Bij de sprites kunnen we wel beschikken over 16 verschillende kleuren, doch deze wijken sterk af van de standaardkleuren en zijn afhankelijk van de kleuren welke we aanvankelijk aan de nummers 0 t/m 3 hebben toegekend.
De sprite-kleuren 0,5,10 en 15 zijn nu gelijk aan de kleuren, welke we aan respectievelijk de nummers 0 t/m 3 hebben toegekend. De overige spritekleuren zijn dan mengkleuren, waarvan sommigen met streepjespatroon.
Daar deze spritekleuren dus geheel afhankelijk zijn van de 4 keuzekleuren is er derhalve geen tabel te geven met deze spritekleuren. Wel kan ik U een kort programma aan de hand doen waarmee U vrij snel alle sprites in beeld kunt brengen:

```
10 SCREEN 6,1,0:COLOR 2,2,2:CLS
20 VDP(9)=VDP(9) OR &H20
30 COLOR=(0,7,0,0):COLOR=(1,0,7,0):'rood/groen
40 COLOR=(2,0,0,7):COLOR=(3,7,7,7):'blauw/wit
50 SPRITE$(0)=STRING$(8,255)
60 FOR Y=0 to 1:FOR X=0 TO 7
70 PUT SPRITE YX8+X,(XX20+10,YX20),YX8+X,0
80 NEXT:NEXT
90 GOTO 90
```

De keuzekleuren in regels 30 en 40 kunt U uiteraard naar eigen smaak aanpassen.

TOEPASSING.

Hoewel we maar over een paar verschillende kleuren kunnen beschikken, heeft dit scherm het voordeel van 4 pagina's met de mogelijkheid van 64 karakters per regel. Daardoor leent dit scherm zich goed voor het maken van (blok-) grafieken, waarbij veel tekst vereist is.

SCREEN 7 (2 pag.).

BEELDOPBOUW.

Ook hier is evenals bij screen 6 het scherm 512 pixels breed, doch nu met de mogelijkheid van 16 verschillende kleuren. Het systeem is hier gelijk aan dat bij screen 5, te weten 2 pixels per byte. De inhoud van een byte is dus weer:

```
          16*kleurnr.1e pixel + kleurnr. 2e pixel.
Of bij HEX-code: 1e karakter = kleurnr. 1e pixel.
                 2e karakter = kleurnr. 2e pixel.
```

SPRITES.

Zie SPRITES ALGEMEEN.

TOEPASSING.

Voor spelletjes met een grotere figuur-en tekstdichtheid, gecombineerd met de mogelijkheid van 16 kleuren. Ook zeer geschikt voor grafieken of diagrammen. Nadeel is alleen dat hier slechts twee pagina's beschikbaar zijn.

SCREEN 8 (2 pag.).

BEELDOPBOUW.

Op dit scherm kunnen we beelden met 256 verschillende kleuren opbouwen. Dit kost echter wel 1 byte per pixel, waardoor de naamtabel bij 256 pixels per beeldlijn toch nog 54272 bytes aan geheugenruimte vergt.
Het kleurnummer en daarmee ook de inhoud van een byte volgt uit: `4*R + 32*G + B\2`, waarin R,G en B dan de intensiteiten zijn van de drie basis kleuren.

SPRITES.

Voor de sprites hebben we maar 16 kleuren ter beschikking. Meer zou ook niet kunnen, omdat de sprite instructies zijn gebaseerd op max. 16 kleuren. De spritekleuren wijken wel af van de normale standaardkleuren en zijn niet te wijzigen. In tabel 2 zijn de 16 kleuren gegeven met daarbij het overeenkomstige kleurnummer uit de 256-range.
De COLOR INTENSITEIT tabel heeft bij dit scherm geen functie omdat hier de kleuren niet kunnen worden gewijzigd. Waarom deze tabel hier dan toch is opgenomen is mij niet duidelijk.

TOEPASSING.

Met dit scherm kunnen, dankzij de vele kleurnuances beelden worden gekreeerd met 'dieptewerking'. Het scherm is daardoor behalve voor spelletjes ook bijzonder bruikbaar voor de 'beeldende kunstenaars' onder de MSXers.

Ger.