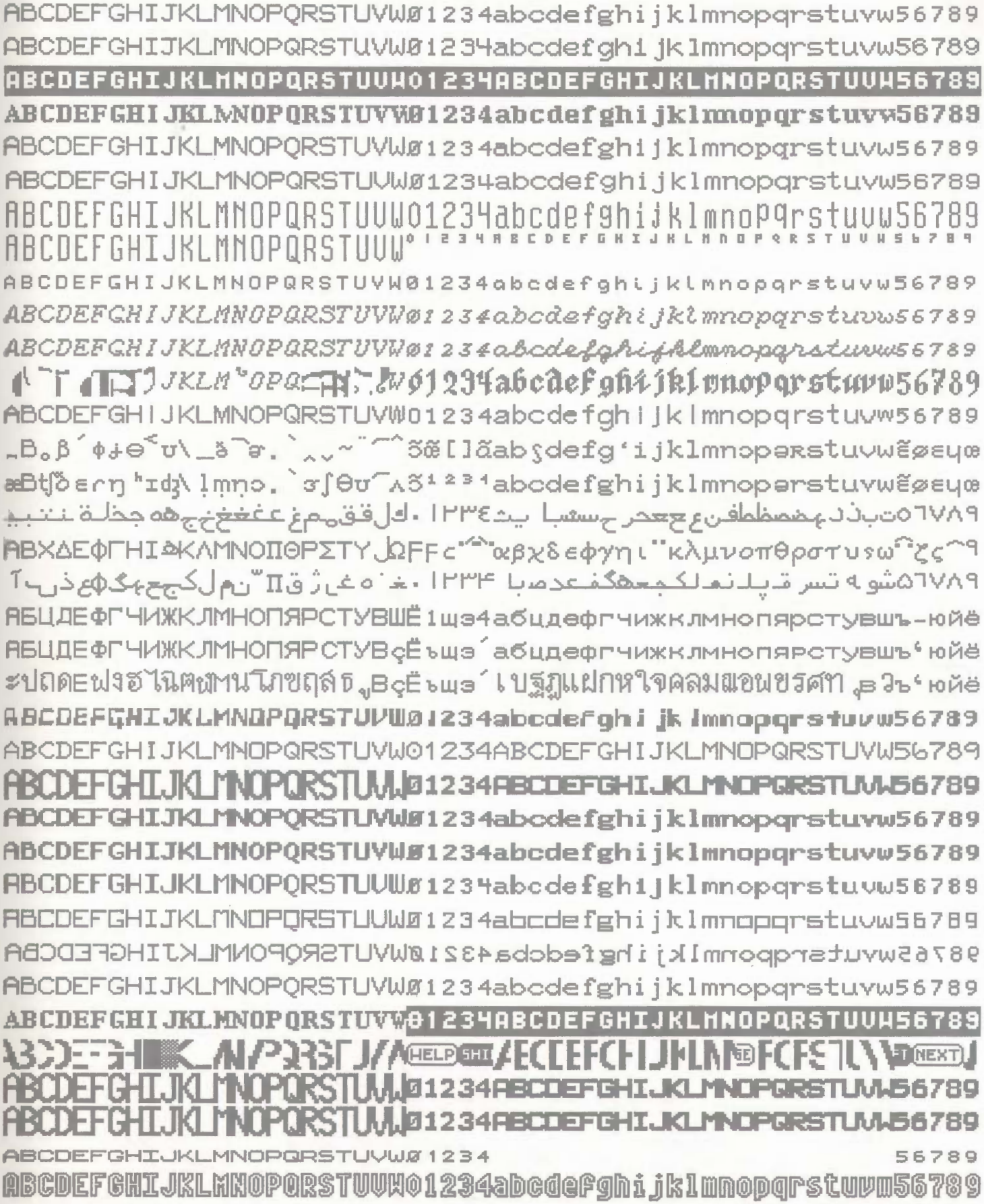
Verkoopprijs:
900 Bfr./Hfl. 49,-

MSX WORKSHOP 88

MSX WORKSHOP 88

SUPERFONT



MSX WORKSHOP 88

Beste lezer,

WORKSHOP 88 is de vierde compilatie-uitgave van MSX CLUB MAGAZINE. Bij de vorige uitgaven konden we ons beperken tot een schijfje en een begeleidende brochure. Mede door het enorme succes van ons disk-abonnement hebben we onze lezers steeds meer materiaal kunnen bieden, op schijf - klaar voor gebruik. Vandaar dat WORKSHOP 88 dit lijvige boekje met 3 schijven is geworden.

Gebruik

Afgezien van de cursussen, zijn de programma's in willekeurige volgorde te gebruiken.
U kan op twee manieren "aanvallen":

- *U kiest een programma uit het boek. In een kadertje vindt U dan de juiste file-naam en op welke schijf U het programma terugvindt.*
- *U kiest een programma van schijf. In de inhoudstafel vindt U dan de pagina waar U een aangepaste toelichting kan vinden.*

We wensen U veel MSX-plezier met WORKSHOP 88,

redactie MSX CLUB MAGAZINE

WORKSHOP 88 INHOUD

#A	001	D2	23	#B-00I2	BAS	D2	42	DECRUNCH	COM	D2	67
#A	002	D2	24	#B-00J1	BAS	D2	42	DIAMOND	FNT	D3	72
#A	008	D2	28	#B-00J2	BAS	D2	42	DIEFSTAL	BAS	D1	91
#A	009	D2	29	#B-00K1	BAS	D2	43	DRAAK	DES	D2	70
#A	03A	D2	26	#B-00K2	BAS	D2	43	DRAAK	MSX	D1	20
#A	03B	D2	26	#B-00L1	BAS	D2	43	DRAAKKRT		D1	20
#A	04A	D2	26	#B-00L2	BAS	D2	43	DRAWDEMO	D1	D1	4
#A	04B	D2	26	#B-00M1	BAS	D2	44	DRAWDEMO	D2	D1	5
#A	05A	D2	27	#B-00M2	BAS	D2	44	DRAWDEMO	D3	D1	6
#A	05B	D2	27	ANIMAL1	BAS	D2	88	DRAWDEMO	D4	D1	6
#A	06A	D2	27	BANNERB	BAS	D1	81	DRAWDEMO	D5	D1	6
#A	06B	D2	27	BELGIANS	BAS	D1	81	DRAWDEMO	D6	D1	7
#A	07A	D2	27	BELGIE	BAS	D1	91	DRAWING	BAS	D1	91
#A	07B	D2	27	BIGFULL	FNT	D3	75	ETIKET	BAS	D2	80
#A	07C	D2	28	BMH1KRT		D1	20	ETIKET	LDR	D2	80
#A	07D	D2	28	BMH2KRT		D1	20	ETIKET	PRG	D2	80
#A-010A	BAS	D2	31	BMH2SNEL		D1	20	ETIKET2	PRG	D2	80
#A-010B	BAS	D2	31	BOL1	BAS	D1	87	FANTASIE	M16	D1	95
#A-011A	BAS	D2	31	BOL2	BAS	D1	87	FATFACE	FNT	D3	75
#A-011B	BAS	D2	31	BOOMH1	MSX	D1	20	FILES	BAS	D2	112
#A-012A	BAS	D2	32	BOOMH2	MSX	D1	20	FINE1	FNT	D3	74
#A-012B	BAS	D2	32	BOS	BAS	D1	11	FINE2	FNT	D3	74
#A-013	BAS	D2	32	BOXBOLD	FNT	D3	74	FOUT	BAS	D1	71
#A-014ML	BAS	D2	34	BROADWAY	FNT	D3	72	GARNET	FNT	D3	72
#A-015ML	BAS	D2	37	BUBBLE	FNT	D3	72	GARNET 2	FNT	D3	72
#B-00A	BAS	D2	39	C41GROOT	SCD	D2	70	GOTH1	FNT	D3	73
#B-00A1	BAS	D2	41	C41KLEIN	SCD	D2	70	HAALUIT	DES	D2	70
#B-00A2	BAS	D2	41	CHESS	BAS	D1	71	HANDSCHR	FNT	D3	74
#B-00B	BAS	D2	39	CILIND1	BAS	D1	87	HAVANAGI	M16	D1	35
#B-00B1	BAS	D2	41	CILIND2	BAS	D1	87	HELI1	BIN	D1	91
#B-00B2	BAS	D2	41	CKSOFT1	FNT	D3	74	HITHOT	BAS	D2	61
#B-00C	BAS	D2	39	CKSOFT2	FNT	D3	74	HUIZNRIJ	BAS	D1	9
#B-00C1	BAS	D2	41	CLAES	STP	D3	73	INST-HEX	BAS	D2	80
#B-00C2	BAS	D2	41	CLS-ROTR	BAS	D2	38	ITALIC	FNT	D3	73
#B-00D1	BAS	D2	41	COENS	STP	D3	73	KADOPAPR	BAS	D1	11
#B-00F1	BAS	D2	41	COLSCA	BAS	D2	61	KAM	MSX	D1	20
#B-00F2	BAS	D2	41	CRUNCH	COM	D2	67	KAMKRT		D1	20
#B-00G1	BAS	D2	42	CUBIC	FNT	D3	75	KAR-TEST	BAS	D2	39
#B-00G2	BAS	D2	42	DALLAS	M16	D1	35	KARTEST	BAS	D2	40
#B-00H1	BAS	D2	42	DARTS	BAS	D1	91	KERSTSPE		D2	56
#B-00H2	BAS	D2	42	DATACREA	BAS	D2	70	KLEUR7	BAS	D2	70
#B-00I1	BAS	D2	42	DATATIME	BAS	D1	87	KLEUR8	BAS	D2	70

KOCH	MSX	D1	20
KOCHKRT		D1	20
KOKO	PCT	D3	68
KOKO1	PCT	D3	69
KRUIMEL	BAS	D1	29
LAATZIEN	DES	D2	70
LAYOUT	STP	D3	72
LAYOUT2	STP	D3	73
LIJNEN	BAS	D1	87
LIST1	BAS	D1	14
LIST2	BAS	D1	14
LIST3	BAS	D1	15
LIST4	BAS	D1	15
LIST5	BAS	D1	15
LIST6	BAS	D1	16
LIST7	BAS	D1	16
LIST8	BAS	D1	16
LIST9	BAS	D1	16
LITALIC	FNT	D3	73
LITELINE	FNT	D3	75
LOGO16		D2	46
LOGO18		D2	49
LOGSPIRA	MSX	D1	20
LOGSPKRT		D1	20
M20LOGO		D2	51
MASTERMI	M16	D1	81
MEERMIN	BAS	D1	29
MEERMIN	DES	D1	35
MILITARY	FNT	D3	74
MISIRLOU	M16	D1	35
MODERN	FNT	D3	74
MONDRIAN	MSX	D1	20
MONDRKRT		D1	20
MOTOR	BAS	D1	71
MSXCURST	BAS	D2	64
MUSICMOD	BAS	D2	112
NOTENLES	BAS	D2	82
OPAL	FNT	D3	72
PACMAN	HOF	D1	71
PALET1	BAS	D1	11
PALET2	BAS	D1	11
PIEN	STP	D3	73
PIJLEN	BAS	D2	60
PINBALL	FNT	D3	75
POPJE	DES	D2	70
PRINT	BIN	D2	80
PROG1	GEN	D2	90
PROG1	MTP	D2	90
PYTB3KRT		D1	20
PYTHB3	MSX	D1	20
RACER	BAS	D1	29
REAKTIE	BAS	D1	87
REISBAG	BAS	D1	91
RELBEW	BAS	D1	11
RENBAC	M16	D1	35
RESUME	M16	D1	35
ROMAN	FNT	D3	75
RUBY12	FNT	D3	72
RUBY8	FNT	D3	72
RUSSIANS	FNT	D3	74
SAFFIER	FNT	D3	75
SCHEMA	PCT	D1	115
SCHRIJF	FNT	D3	73
SCHUIF	BAS	D1	91
SCRIPT	FNT	D3	73
SFEDIT	BAS	D2	79
SHADOW	FNT	D3	73
SHOW	DES	D2	70
SHOWPIC	COM	D2	67
SLANGEN	BAS	D1	91
SMALL	FNT	D3	72
STANDA 2	FNT	D3	73
STANDARD	FNT	D3	72
STENCIL	FNT	D3	73
STER	MSX	D1	20
STERKRT		D1	20
STERREN	BAS	D1	9
STREEP	FNT	D3	74
STYLING1	FNT	D3	74
STYLING2	FNT	D3	74
SUPERF	BAS	D2	55
SUPERF	PIC	D2	55
TBEXT	PAS	D2	90
TERMINUS	BAS	D1	29
TERUG1	BAS	D1	10
TERUG2	BAS	D1	10
TERUG3	BAS	D1	10
TESTDEM2	BAS	D2	35
TESTDEM3	BAS	D2	39
TESTDEMO		D2	23
TESTGR3	COM	D2	87
TESTGR3	PAS	D2	87
UURROOS	BAS	D1	78
VANGNET	BAS	D1	87
VERKIEZ	BAS	D1	71
VETCURSP	FNT	D3	74
VLO	BAS	D1	71
VRDWIJNP	BAS	D1	11
WIKKEL	MSX	D1	20
WIKKLKRT		D1	20
WO3	BAS	D1	81
WOESTIJN	MX1	D2	61
WOESTIJN	MX2	D2	61
YESTERD	BAS	D1	29
Z80	PCT	D3	45

Electronicaprojecten voor MSX computers 76
Edit mogelijkheden 83
Super Impose & Video 92
Conversie MSX-BASIC IBM/GWBASIC 94
Tekenplank 99
Printer selector interface 100
MSX LOGO - LCN LOGO 105
Datacommunicatie 108

De files met de extensie .FNT zijn fonts voor DYNAMIC PUBLISHER.
De files met de extensie .PCT zijn schermen voor DYNAMIC PUBLISHER.
De files met de extensie .STP zijn stempels voor DYNAMIC PUBLISHER.

DRAW : GRAPHIC MACRO LANGUAGE

In deze aflevering wil ik de grafische mogelijkheden van onze MSX nogmaals wat nader bekijken. Met name wil ik de DRAW instructie grondig behandelen. Al testend kwam ik er al snel achter dat deze instructie zeer veel mogelijkheden biedt. Jammer genoeg kan de DRAW ook veel problemen geven voor de programmeur, die niet goed voorbereid is. Mijn oorspronkelijke plan om een mogelijkheid te maken om dezelfde gegevens zowel naar het scherm als naar de plotter te sturen moest ik (voorlopig) opgeven. Het leek mij een goed idee om een programma te hebben, waarmee mensen, die al een programma hadden voor een tekening op de plotter (bv de Sony PRN-041) datzelfde programma konden gebruiken als invoergegeven voor mijn nieuwe programma, dat dan een soortgelijk plaatje op het scherm zet.
Ook het omgekeerde kan erg handig zijn; omdat de plotter vrij traag is en de pennetjes vrij duur, kan uittesten op het scherm handig zijn. Maar misschien lukt dat een volgende aflevering wel. Wil iemand anders dit idee uitwerken, mag dat gerust van mij, maar laat het dan even weten, zodat dubbel werk wordt vermeden.

DRAW

De DRAW is eigenlijk geen instructie zoals de andere in MSX-BASIC. Het is veel meer een aanroep van een andere programmeertaal waarin kan geprogrammeerd worden door de opdrachten in die andere taal als parameter mee te geven aan de DRAW. MSX-BASIC kent nog zo'n instructie namelijk de PLAY die voor geluid is wat de DRAW voor tekeningen is.

Graphics Macro Language

De taal waarin we programmeren door gebruik van de DRAW is de Graphics Macro Language of kortweg GML. Deze taal kent maar een zeer bescheiden aantal (dertien) instructies en kan daarom ook niet zelfstandig alles doen wat je van een programmeertaal mag verwachten. Voordat we aan de slag kunnen met GML moet er eerst wat voorbereid worden. GML is een macrotaal en dat wil zeggen dat er alleen bepaalde routines aangeroepen kunnen worden om de verschillende zaken te laten uitvoeren. GML voert, zoals uit de naam reeds mocht worden opgemaakt, alleen grafische instructies uit.

Voorbereiding

We beginnen met de meest simpele vorm. We dienen voor het aanroepen van de DRAW alleen maar een SCREEN commando te geven (met een grafisch scherm natuurlijk) en we kunnen aan de slag met DRAW. Als we echter klaar zijn met tekenen of beter gezegd als de MSX alle DRAW-instructies heeft uitgevoerd, zal het scherm terugspringen naar het laatste tekstscherm en met de prompt aangeven de volgende opdracht te willen krijgen.
Onze al dan niet geslaagde tekening zijn we dan direct kwijt en dat zal wel niet de bedoeling geweest zijn. Om dit te voorkomen moeten we het programma na de DRAW's laten eindigen met bijvoorbeeld de regel : 999 goto 999. Maar ook andere methodes kunnen aangewend worden. Om te beginnen wil ik eerst een minimaal oefen programma laten zien dat de werking van de meeste opdrachten al toont.

```
10 REM draw-ing demo 1 / F.H. Druijff
- 2/88
20 SCREEN 2
30 DRAW "r60"
40 DRAW "d50"
50 DRAW "e30"
60 DRAW "f20"
70 DRAW "g50"
80 DRAW "l30"
90 DRAW "h20"
100 DRAW "u20"
110 GOTO 110
```

DRAWDEMO D1 D1

programma 1

We zien in dit programma de minimale voorbereidingen die getroffen moeten worden voordat DRAW gebruikt kan worden. In regel 30 beginnen we dan voor het eerst de DRAW te gebruiken en wel door de tekenopdracht een lijn van vermoedelijk lengte 60 naar rechts te trekken.
Na zeven andere lijnen (lijnstukken ware beter geformuleerd) is onze tekening af. Regel 110 zorgt ervoor dat we de tekening blijven zien. We onderbreken het programma nu door de CTRL en STOP toets gelijktijdig in te drukken en krijgen de mededeling dat het programma in regel 110 werd onderbroken. We laten het programma gelijk weer lopen en geloven onze ogen niet ! Nog maar eens en ja hoor we hadden het de eerste keer goed gezien. We krijgen iets anders als de eerste keer. De kronkellijn verhuist langzaam maar zeker naar rechts en naar beneden. De MSX onthoudt blijkbaar waar we net geëindigd zijn en begint bij een nieuwe run op de plaats waar de vorige eindigde.
Dit betekent dat we bij gebruik van DRAW altijd voor een beginpunt moeten zorgen als we willen dat elke keer hetzelfde resultaat verkregen wordt.

Hebben we de tekening na een aantal malen zover naar beneden gekregen dat we de onderlijn van het scherm gaan passeren,dan blijkt dat de figuur ook verandert van vorm.

De beste manier om te begrijpen hoe de tekening wordt 'aangepast' is door je voor te stellen dat de punten die onder het scherm zakken allemaal naar boven worden geveegd tot ze op de onderrand liggen. De lijnen worden dan naar die randpunten getrokken. De wiskundige spreekt van het projecteren van de punten onder de onderlijn op de onderlijn. Gaan we alsmaar door zal onze tekening uiteindelijk alleen een punt rechtsonder op het scherm worden. Zelfs als alle punten buiten het scherm liggen !

Beginpunt

Om een vast beginpunt te krijgen kunnen we dit in of buiten GML vastleggen. Elke 'normale' grafische instructie heeft een eindpunt. Dit eindpunt is daarna het beginpunt voor DRAW als die daar op volgt.
De meest eenvoudige instructie om altijd op punt (0,0) te beginnen is natuurlijk PSET (0,0) of PRESET (0,0) Voor de algemeenheid gaat de voorkeur naar de PRESET. Bij PSET wordt namelijk al direct een punt in de voorgrondkleur neergezet en als de DRAW niet ook dit punt tekent staat er een punt op een niet gewenste plaats. Met PRESET zetten we een punt in de achtergrondkleur en lijkt het probleem weg. Echter met PRESET wordt ook die kleur als tekenkleur gekozen en zien we zonder kleurcommando in de DRAW niets van de tekening. De tekening wordt wel neergezet, maar is niet zichtbaar. Dus zullen we toch voor PSET kiezen met eventueel de kleur erbij.
De andere oplossing is het beginpunt in GML te kiezen door de DRAW "bm0,0". De uitleg van deze instructie volgt verderop.

Eén letter instructies

Zoals we al zagen in het voorbeeldprogramma wordt de GML-taak aan DRAW gegeven of als dat logischer klinkt geeft DRAW aan GML de instructie tussen aanhalingstekens (double quotes) en bestaat de GML-instructie uit een letter die vaak gevolgd wordt door bijvoorbeeld een of meer getallen. In het eerste voorbeeld werden de volgende instructies gebruikt : r,l,u en d en verder e,f,g en h. Zij worden alle gevolgd door een getal dat een maat is die gebruikt wordt bij de lengtebepaling van de lijn die getrokken gaat worden. Als de grootteinstelling niet verandert, hebben de instructies de volgende betekenis :

r60	Trek een lijn met lengte 60 naar rechts
l30	Trek een lijn met lengte 30 naar links
u20	Trek een lijn met lengte 20 naar boven
d50	Trek een lijn met lengte 50 naar beneden

Met de engelse benamingen in het achterhoofd is dit simpel te onthouden : R(ight) = rechts, L(eft) = links, U(p) = op,omhoog en D(own) = beneden,omlaag. Voor de diagonale richtingen was het moeilijker voor de ontwerpers. Logische benamingen zoals noordoost kunnen niet omdat de afkortingen dan minstens twee letters moeten zijn.
Men koos er voor om met de wijzers van de klok mee vanaf 12 uur de diagonale richtingen aan te geven met e,f,g en h. Erg logisch is dit niet en nogal inconsequent ten opzichte van de horizontale en vertikale keuze. Maar goed, de keuze is nu eenmaal zo gedaan en we kunnen beter deze afspraak ons eigen maken dan een discussie houden over hoe het ook had gekund.

Overzicht

Maar er zijn meer positieve en negatieve aspecten aan de DRAW instructie.

Tekststring

In het eerste voorbeeld zagen we reeds dat de GML- instructies gegeven werden als tekststring. En gebruik aub in mijn nabijheid niet het woord karakterketting, al is dit goed; ik krijg er koude rillingen van. Maar als het als tekststring is te geven, valt het misschien ook als variabele tekststring op te geven. De listing hierna laat dit zien. In regel 40 wordt A$ gedefinieerd en in regel 50 bij DRAW gebruikt. Ik heb niet alle mogelijkheden geprobeerd maar ik heb sterk de indruk dat de tekststring na DRAW net als bij PRINT of LPRINT is te construeren.

DRAWDEMO D2 D1

```
10 REM draw-ing demo 2 / F.H. Druijff
- 1/88
20 SCREEN 2
30 PSET (50,140)
40 A$="s4r20u20l20d20e20h10g10f20"
50 DRAW A$
60 H$=INKEY$:IF H$<>"s" AND H$<>"S" GO
TO 60
70 SCREEN 0:LIST
```

programma 2

Tevens zien we dat in dit programma ook de PSET gebruikt worden om het huisje met kruis op de gewenste plaats te krijgen. In regel 60 en 70 zien we ook nog wat handigheidjes staan. Het is te begrijpen dat ik in de testfase ter voorbereiding van een artikel vele veranderingen in mijn testprogramma's aanbreng niet alleen om het programma goed te maken maar vaak ook om expres fouten te maken om te zien hoe er op gereageerd wordt.
Om nu elk programma steeds te moeten onderbreken met CTRL en STOP toets om dan de listing op te vragen is soms wat lastig. Het kan immers slimmer zie de listing hiervoor. Aan het eind van de tekening vraagt de computer steeds om de S-toets wordt deze met of zonder SHIFT ingedrukt zal het scherm tekstscherm 0 worden en zal de listing ter verbetering, verandering worden getoond.
Ik heb deze studiestop in het programma laten staan omdat de geïnteresseerde lezer daar ook zijn gemak van kan hebben. De cassette / diskette abonnementhouders zullen deze listings dan ook zo terug vinden op bandje of schijf.

Schuine lijnen

Aan de hand van de listing hiervoor kunnen we zien dat we, om bijvoorbeeld 40 naar rechts te gaan, dat kunnen doen door r40 te geven. Maar ook e20f20 en f20e20 komen op hetzelfde punt uit.
Bij de diagonale lijnen wordt dus niet de lengte van de lijn opgegeven maar de lengte van de projectie van die lijn. e10 betekent dus niet : trek een lijn naar rechtsboven met een

basislengte van 10 maar trek een lijn naar rechtsboven die zowel in horizontale als vertikale richting een geprojecteerde lengte van 10 heeft.

Tekenmaat

Rest nog in dit voorbeeldprogramma 2 de s4 aan het begin van A$. Het is misschien al opgevallen hoe voorzichtig ik omging met de term lengte. Dit komt omdat bij DRAW met behulp van de s-instructie de maat veranderd kan worden. Als op een schone machine begonnen wordt, is zonder verandering de s-maat 4. Dit betekent dat alle opgegeven getallen ook de aantallen pixels zijn die getekend worden.
Geven we echter s de waarde 8 mee zal alles tweemaal zo groot worden op het scherm. Met de waarde 2 bij s zal alles tweemaal zo klein worden.
Tot zover geen probleem maar als even de moeite wordt genomen in voorbeeld programma 2 een waarde die oneven is voor s te proberen, zal men snel inzien dat niet altijd dat verkregen wordt wat gewenst werd. Om problemen te voorkomen raad ik een ieder aan alleen viervouden te gebruiken in de echte tekeninstructie en dan kan men elke maat kiezen die gewenst wordt of men wil juist in die tekeninstructie vrij zijn, maar dan raad ik aan voor de s een viervoud te kiezen. Beide even gekozen is ook goed. Doet men dit niet krijgt men regelmatig last van niet goed aansluitende lijnen.
Bij de DRAW kunnen we trouwens alleen gehele getallen gebruiken, andere worden omgewerkt.

```
10 REM draw-ing demo 3 / F.H. Druijff
- 1/88
20 SCREEN 2
30 DRAW "s5bm50,140"
40 A$="r20u20l20d20e20h10g10f20"
50 DRAW A$
60 H$=INKEY$:IF H$<>"s" AND H$<>"S" GO
TO 60
70 SCREEN 0:LIST
```

DRAWDEMO D3 D1

prog3

Voorgaande listing verschilt niet veel van de vorige. Het verschil zit in regel 30. Daar wordt de s al opgegeven maar interessanter is het vervolg. Onze tekenpen wordt daar verplaatst naar punt (50,140).
De instructie m komt van move = beweeg maar omdat dan een lijn getrokken zou worden van het laatste punt naar dit punt (50,140) staat er een b voor zodat de verplaatsing onzichtbaar is. b komt van blank.

Duidelijkheid

Ik vind het altijd van groot belang dat programma's leesbaar zijn. en ik bedoel daar niet alleen mee dat het programma netjes in elkaar moet steken. Maar ook dat de listing o.a. voldoende spaties bevat. Daarom adviseer ik iedereen ook om de instructies in GML in kleine letters op te geven omdat letters en cijfers dan gemakkelijker in groepjes uiteen vallen. Om de leesbaarheid nog verder op te voeren vroeg ik mij af of het toegestaan was in de A$ spaties te zetten.

In de volgende listing staat zo'n voorbeeld en het bleek toegestaan met spaties. Maar andere tekens werden geweigerd. Jammer.

DRAWDEMO D4 D1

```
10 REM draw-ing demo 4 / F.H. Druijff
- 1/88
15 ON ERROR GOTO  70
20 SCREEN 2
30 DRAW "s4 bm50,140"
40 A$="r23 u23 l23 d23 e23 h11 l1 g11
f23"
50 DRAW A$
60 H$=INKEY$:IF H$<>"s" AND H$<>"S" GO
TO 60
70 SCREEN 0:LIST
```

prog 4

Een ander handigheidje, dat ook in de voorgaande listing staat maar nu bij correcte listings eigenlijk overbodig is geworden, is de volgende techniek :

15 ON ERROR GOTO 70

Als nu bij het aanbrengen van een verandering een programmeerfout gemaakt wordt, zal het programma zichzelf automatisch listen. Indien gewenst kunnen we in die regel 70 (of daarna) ook het regelnummer van de regel waar het fout ging laten afdrukken, het foutnummer en eventueel de foutmelding. Dit moet dan wel VOOR de LIST omdat het programma na LIST automatisch stopt.

Ik kan deze techniek elke serieuze programmeur aanbevelen.

Omdat ik het plan had een combinatie programma te maken voor plotter en scherm leek het mij verstandig in ieder geval in het begin de tekenopdrachten in DATA regels op te slaan. Ik zou daarvan in ieder geval een schermroutine met GML en een plotterroutine kunnen maken.

Om dat althans voor GML te oefenen de volgende listing:

DRAWDEMO D5 D1

```
10 REM draw-ing demo 5 / F.H. Druijff
- 1/88
15 ON ERROR GOTO 60
20 SCREEN 2
30 DRAW "s4 bm100,140"
40 READ A$:IF A$<>"einde" THEN DRAW A$
:GOTO 40
50 H$=INKEY$:IF H$<>"s" AND H$<>"S" GO
TO 50
60 SCREEN 0:LIST
70 DATA r40 u40 l40 d40
80 DATA "bm+20,+10"
90 DATA e30 h30 g30 f30
100 DATA einde
```

prog 5

We zien in dit programma weer veel bekende zaken, maar toch ook een paar nieuwe. We starten vanuit punt (100,140) en tekenen daar rechtsboven een vierkant. Ik wil daar een tweede vierkant van gelijke grootte schuinoverheen leggen. Ik kan nu na het tekenen van het eerste vierkant de pen verplaatsen naar punt (120,150) en van daaruit het tweede vierkant tekenen. Ik wil echter gebruik maken van de mogelijkheid relatief, dat wil zeggen ten opzichte van het punt waar we ons bevinden, te verplaatsen. We doen dit door de gewenste verplaatsing aan te geven met teken (+ of -).
De gewenste verplaatsing is 20 naar rechts en 10 naar beneden en het vermoeden is dat dit gebeurd door m+20,-10. Maar het MSX-scherm speelt ons parten; op deze manier gaan we niet 10 naar beneden maar juist 10 naar boven.
De y=coördinaat wordt immers met 10 verminderd ! Of deze ellende al niet genoeg is, komt er nog een probleem om de hoek kijken. Bij de m instructie worden twee getallen meegegeven, gescheiden door komma's. Maar in onze DATA notatie zijn dit dan twee velden.
Door de gekozen leesroutinevorm kan de tekst die ingelezen wordt niet verwerkt worden. We lossen het op door het geheel tussen aanhalingstekens te plaatsen. (regel 80)
Een andere oplossing had ook gekund

80 DATA br20,bd10

of zonder komma, want dat maakt nu niet uit.

Draaien

Met de instructie A gevolgd door 0,1,2 of 3 kunnen we alles 0, 90, 180 of 270 graden draaien. Ik heb daar nog geen voorbeelden van gemaakt maar, net zoals streepjeslijnen, zal ik de volgende keer daar iets van laten zien.

Snelheid

Waarom zou iemand nu met DRAW en GML willen programmeren in plaats van LINE, PSET en dergelijke ? Het gemak van programmeren kan een rol spelen maar ook best een kwestie van gewenning zijn. Mogelijkerwijs zou de snelheid van het programmma een argument kunnen zijn. Programmavoorbeeld 6 tracht dit eventuele snelheidsverschil zichtbaar te maken.
Uit dit programma bleken mij geen spectaculaire verschillen tussen DRAW en de andere mogelijkheden.

```
10 REM draw-ing demo 6 / F.H. Druijff
- 2/88
15 ON ERROR GOTO 110
20 SCREEN 2
25 TIME=0:'routine 1
30 DRAW "bm130,180"
40 FOR S=1 TO 60
50 DRAW "s=s;r8 u8 l8 d8"
60 NEXT
65 A=TIME:'routine 2
70 FOR S=1 TO 60
80 LINE (8,160)-(8+S+S,160-S-S),,B
90 NEXT
95 B=TIME
100 H$=INKEY$:IF H$<>"s" AND H$<>"S" G
OTO 100
110 SCREEN 0:PRINT "routine 1 :"A/50;"
sec."
120 PRINT "routine 2 :"(B-A)/50;"sec."
:PRINT:LIST
```

prog 6

DRAWDEMO D6 D1

Na de initialisatie wordt de tijd op nul gezet (regel 25) Vervolgens wordt met DRAW een zestigtal rechthoeken neergezet met gelijke linkeronderhoeken. De tijd wordt genoteerd (regel 65) en we gaan andermaal maar nu met LINE een zestigtal rechthoeken neerzetten. Wederom wordt de tijd genoteerd (regel 95) en zodra we dit wensen wordt de tijd afgedrukt. Zoals al eerder werd gezegd: doe dat voor de LIST.

Variabele waarden

In regel 50 zien we echter nog iets vreemds staan. s=s; in plaats van de gebruikelijke s8. Met deze constructie is het mogelijk een variabele waarde op een aangename manier aan GML door te geven. We hadden natuurlijk met de STR$ functie de waarden in karaktervorm kunnen krijgen en met eventueel nog een RIGHT$ om de leidende spatie kwijt te raken, maar gemakkelijk is anders. In GML kunnen we gewoon na de instructie (hier dus de s) een 'is gelijk aan'-teken (=) zetten en daarachter de variabelenaam waarin de waarde staat. Ik koos hier voor de variabelenaam s omdat ik dan gelijk weet waar het bij hoort. Omdat ik niet de enige ben die zulke variabelenamen kiest, moet de gebruiker zijn variabelenaam besluiten met een puntkommma (;).

Instructie als variabele

Tot besluit de x-instructie. Net zoals het gemakkelijk is variabele waarden mee te geven, zo kan het ook handig zijn een aantal instructies als variabele mee te geven. We denken nog even aan de A$ uit de eerste voorbeelden. Willen we deze A$ dan meermalen als onderdeel van een grotere opdracht hebben, kunnen we dat als volgt doen :

40 DRAW "r40xA$;r30xA$;r20xA$;r10"

Nu wordt A$ driemaal ingevoegd in de DRAW. Ik wil hier nog een aantal vooorbeelden van laten zien, maar door ruimtegebrek moet dat wachten tot de volgende keer.

Veel programmeerplezier
Frank H. Druijff

DRAW : verder uitgediept...

In deze aflevering wil ik verder gaan met nog zaken te bespreken die te maken hebben met de DRAW-instructie. In nummer 16 was ik hier reeds mee begonnen maar er lagen nog wat losse opmerkingen, die toen in dat artikel niet aan bod gekomen zijn. Omdat we een nummer overgeslagen hebben is het wellicht verstandig de hoofdzaken voor DRAW nog even te herhalen.

Graphics Macro Language

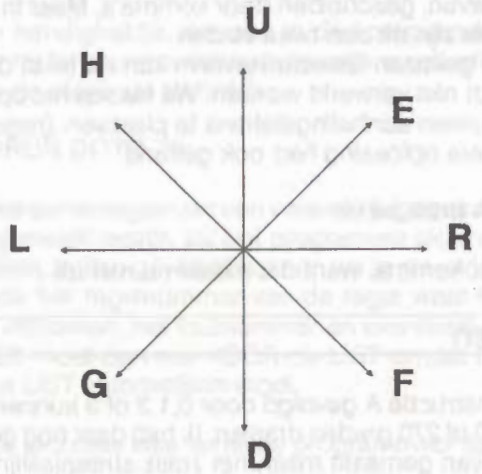
Met DRAW roepen we als het ware een miniprogrammeertaal aan. De taal waarin we programmeren door gebruik van de DRAW is GML. De graphics macro language is, zoals de naam doet vermoeden, een taal die via macro's grafische opdrachten verwerkt. In de letterlijke betekenis zou dat betekenen dat de instructies vertaald zouden worden in de andere basicinstructies zoals PSET en LINE maar dit is toch niet zo. We kunnen de DRAW beter zien als een in BASIC ingebouwde runtime compiler. De enige instructie die een soortgelijke benaderingswijze kent is de PLAY die voor geluid doet wat de DRAW voor grafisch werk doet.
De taal kent instructies van een letter. We kunnen naar eigen voorkeur hoofdletters of kleine letters gebruiken. Mijn voorkeur gaat uit naar kleine letters omdat die beter opvallen tussen de cijfers en de omliggende BASIC-instructies. Binnen redelijke grenzen kunnen ook spaties ingepast worden naar eigen inzicht. Ik kies voor een spatie tussen elke instructie. Gelukkig besloot mijn collega redacteur Clark ook voor het plaatsen van spaties tussen de instructies hetgeen de leesbaarheid van de opdrachten aanzienlijk vergroot.
Ik geef nu eerst een lijst van alle letters / instructies die gegeven kunnen worden in GML.

Instructieset

a of A	voor hoek	(eng = Angle)
b of B	voor blanco	(eng = Blank)
c of C	voor kleur	(eng = Colour)
d of D	voor omlaag	(eng = Down)
e of E	voor naar rechtsboven	)
f of F	voor naar rechtsbeneden	) vier letters
g of G	voor naar linksbeneden	) op volgorde
h of H	voor naar linksboven	)
l of L	voor naar links	(eng = Left)
m of M	voor verplaatsen	(eng = Move)
n of N	voor terugkeer	(eng = No)
r of R	voor naar rechts	(eng = Right)
s of S	voor grootte	(eng = Size)
u of U	voor omhoog	(eng = Up)
x of X	voor doe / voer uit	(eng = eXecute)

Voor de acht richtingen die we aan kunnen geven is de volgende 'windroos' een handig hulpmiddel.

Overzicht



Aan de meeste instructies wordt een waarde meegegeven die aangeeft hoeveel de verplaatsing bedraagt. Andere instructies vragen natuurlijk andere waarden dan die voor verplaatsing. De b/B en n/N hebben zelfs geen getalwaarde maar een andere instructie als argument.
We laten ze even de revu passeren :

Windroosverplaatsingen

Voor de instructies uit de windroos dus u,d,l,r en e,f,g,h geldt dat de opgegeven waarde alleen aan de getalgrenzen moet voldoen die binnen MSX bestaan. Wordt het product van schaalfactor en de lengtefactor echter meer dan 32767 (de maximale positieve integer) zal de opdracht niet uitgevoerd worden. Het programma geeft geen foutmelding en zal gewoon doorgaan met de volgende instructie. Ik heb niet onderzocht of de 'foutieve' waarde leidt tot afbreken van de gehele DRAW of alleen deze tekenopdracht niet uitvoert. In alle gevallen is de instructie fout omdat geen enkel scherm deze waarde ook maar bij benadering kan uitvoeren. De reden voor mij om het hier te vermelden is dat een aantal programmeurs misschien onverklaarbare fouten krijgen en niet weten in welke hoek zij de fout moeten zoeken.

Move

De normale m/M instructie kan vanzelfsprekend naar elk punt van het scherm verplaatsen. De opgegeven getallen moeten wel op het gekozen scherm liggen om zinnig te zijn. Net zoals bij bijvoorbeeld LINE protesteert de MSX echter niet op te grote waarden maar legt die dan wel op de rand met alle meestal niet gewenste gevolgen van dien.
Zoals in het vorige artikel reeds genoemd kunnen we de verplaatsopdracht ook relatief opgeven door het plaatsen van plus / min tekens voor de waarde. Er schuilt hier echter een levensgrote adder onder het gras. De waarde die zo relatief wordt opgegeven wordt voor uitvoer wel eerst bewerkt met de schaalfactor. Ik ben geen tegenstander van deze aanpak maar wil wel een ieder waarschuwen dit terdege in de gaten te houden bij en andere schaalfactor dan vier.

Hoek

Hier kunnen we erg kort over zijn. Er zijn slechts vier hoeken mogelijk namelijk 0, 90, 180 en 270 graden. 360 graden is weer hetzelfde als 0 graden en bij de a/A geven we dus alleen 0,1,2 of 3 op om de respectievelijke hoeken aan te geven.

Kleur

De kleur die opgegeven kan worden hangt alleen af van het gekozen scherm om zinnig te zijn. DRAW accepteert alles van 0 tot 255.

Schaal

Ook bij de schaal geldt dat waarden van boven de 255 uit den boze zijn. Denk echter aan de eerder beschreven beperkingen. Waarde 0 voor de schaal mag zeer beslist niet en leidt tot foutmeldingen. (Jawel de MSX controleert hier eens !)
De laatste instructies kennen geen getalargumenten.

10 REM Huizenrij / F.H. Druijff - 4/87

```
20 SCREEN 2
30 A$="r8 u10 l5 d10 r17 u20 l20 e10 f10 l20 d20"
40 FOR I=10 TO 240 STEP 22
50 DRAW "bm=i;,50"+A$
60 NEXT
70 GOTO 70
```

HUIZNRIJ BAS D1

Blank

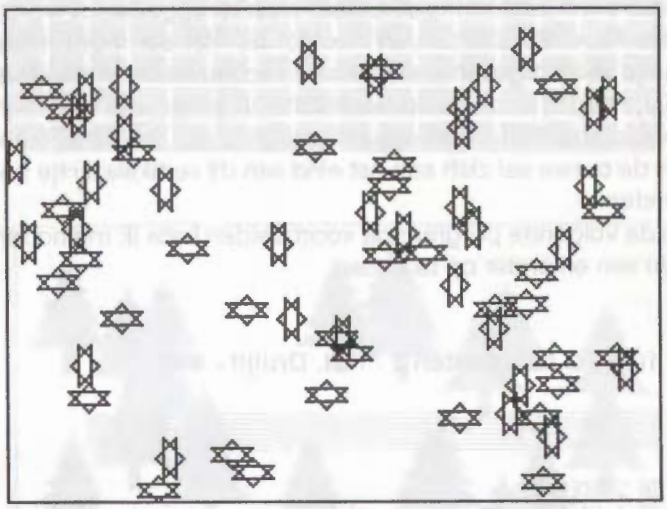
Als voor een der tekeninstructies een b/B gezet wordt zal de uitvoering van die instructie niet op het beeldscherm zichtbaar worden.
Ik raad iedereen ook ten sterkste aan om ten einde de grafische cursor op een bepaalde plaats bijvoorbeeld (50,70) te krijgen de instructie
DRAW "bm50,70"
te gebruiken en niet
PRESET (50,70)
omdat de tekenkleur dan nog wel eens anders blijkt te zijn of worden dan men zich wenste.
Noot : altijd goed is : PSET(X,Y),POINT(X,Y)

Bij de andere instructies mag hij wel staan maar is dan wel zinloos.

10 REM sterren / F.H. Druijff - 4/87

```
20 SCREEN 2
30 A$="r16 g8 h8 bm+8,-3 f8 l16 e8"
40 J=1-J:DRAW "a=j;"
50 X=240*RND(1):Y=170*RND(1)+16
60 DRAW "bm=x;,=y;"
70 DRAW A$
80 IF INKEY$="" GOTO 40
90 CLS:GOTO 40
```

STERREN BAS D1



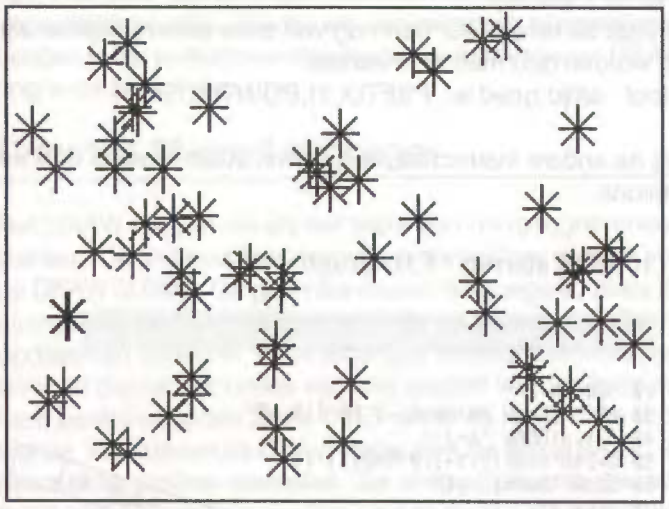
Terugkeer naar uitgangspunt

Door voor een van de tekeninstructies een n/N te zetten kunnen we er voor zorgen dat na het uitvoeren van de instructie die volgt de grafische cursor weer geplaatst wordt waar hij stond voor het uitvoeren van de laatste instructie.

1Ø REM terugzetten 1 / F.H. Druijff - 5/88

```
2Ø SCREEN 2
3Ø X=1Ø+236*RND(1)
4Ø Y=1Ø+172*RND(1)
5Ø K=1+15*RND(1)
6Ø DRAW "s4 bm=x;,=y; c=k;"
7Ø DRAW "nu9 ne6 nr7 nf6 nd9 ng6 nl7 nh6"
8Ø IF INKEY$<>" " GOTO 3Ø
9Ø SCREEN Ø:LIST
```

TERUG1 BAS D1



Lastig kan wel zijn met het - hierna te behandelen - invoegen van strings met opdrachten dat alleen na de eerste opdracht van deze string de cursor weer teruggeplaatst wordt. Dus A$="r20 u20":DRAW "nxA$;" zal niet een horizontaal lijntje te zien geven waar aan de rechterkant een vertikaal lijntje begint en de cursor weer in het uitgangspunt maar een horizontaal lijntje met een vertikaal lijntje aan de LINKERkant en de cursor zal zich aan hét eind van dit vertikale lijntje bevinden.
In de volgende programma voorbeelden toon ik methoden om een en ander op te lossen.

1Ø REM terugzetten 2 / F.H. Druijff - 5/88

```
2Ø SCREEN 2
3Ø X=1Ø+236*RND(1)
4Ø Y=1Ø+172*RND(1)
5Ø K=1+15*RND(1)
6Ø DRAW "s4 bm=x;,=y; c=k;"
7Ø FOR I=1 TO 8 : READ A$:DRAW "nxA$;":NEXT
8Ø RESTORE
9Ø IF INKEY$<>" " GOTO 3Ø
1ØØ SCREEN Ø:LIST
11Ø DATA u9,e6,r7,f6,d9,g6,l7,h6
```

TERUG2 BAS D1

1Ø REM terugzetten 3 / F.H. Druijff - 5/88

```
2Ø SCREEN 5
3Ø FOR I=1 TO 8 : READ H$ : A$=A$+" n"+H$ : NEXT
4Ø DATA u9,e6,r7,f6,d9,g6,l7,h6
5Ø X=7+242*RND(1)
6Ø Y=9+174*RND(1)
7Ø K=1+15*RND(1)
8Ø DRAW "s4 bm=x;,=y; c=k; xA$;"
9Ø IF INKEY$<>" " GOTO 4Ø
1ØØ SCREEN Ø:LIST
```

TERUG3 BAS D1

Een combinatie van b/B en n/N is toegestaan maar natuurlijk volkomen onzinnig. Zoiets als het aanwijzen van iets op het moment dat niemand kijkt.

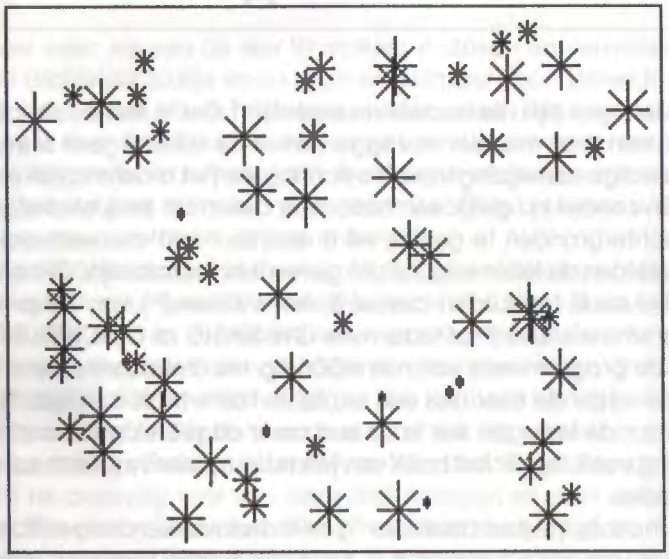
Invoegen

Nu het invoegen van variabelen voor waarden of zelfs volledige strings met opdrachten in andere opdrachtenseries. Bij verscheldene van de hiervoor geplaatste programma's zag de oplettende lezer al een '=' teken of x/X instructie staan. Ik behandel ze hier samen omdat er een aantal aardige overeenkomsten zijn.
Als we een getalswaarde van 20 aan bijvoorbeeld de r/R willen doorgeven kunnen we dat als volgt doen :
DRAW "r20"
maar het kan handig zijn als de waarde 20 in een variabele Q opgeborgen kan worden. De uitvoering wordt dan :
DRAW "r=Q;"
Op het moment dat de DRAW-interpreter dus een getalswaarde verwacht staat er een = om aan te geven dat hij de waarde in het geheugen dient op te zoeken. Erachter komt dan de naam van de variabele die de waarde bevat. Tot slot een kommapunt om het einde van deze instructie aan te geven. Vergelijk de kommapunt maar met de return aan het eind van een BASIC regel of de dubbele punt tussen de BASIC-instructies.
De x/X instructie doet iets soortgelijks maar is wezenlijk anders. De = kondigt een variabele aan die de gezochte waarde bevat . Het is zelf geen instructie, maar kondigt een argument aan. De x/X kondigt ook iets aan maar niet een argument voor bij een opdracht maar een hele serie opdrachten die in dit geval in een tekstvariabele zijn opgeslagen.

10 REM cadeaupapier / F.H. Druijff - 5/88

```
20 SCREEN 5
30 FOR I=1 TO 8 : READ H$ : A$=A$+" n"+H$ : NEXT
40 DATA u9,e6,r7,f6,d9,g6,l7,h6
50 X=7+242*RND(1)
60 Y=9+194*RND(1)
70 K=1+15*RND(1)
80 S=6*RND(1)
90 DRAW "s=S; bm=X;,=Y; c=K; xA$;"
100 IF INKEY$<>" " GOTO 40
110 SCREEN 0:LIST
```

KADOPAPR BAS D1



Ook hier weer de waarschuwing dat de DRAW instructie erg veel toelaat zonder foutmelding. De slordige programmeur zal dan ook zelden een foutmelding krijgen maar het opgegeven programma zal weinig of niets met de uitvoering te maken hebben. Zelfs al is het resultaat de gewenste tekening behoeft het programma nog niet goed te zijn.
DRAW "r20 xA$+A$;"
In deze opdracht zal niet de opdrachtenserie zoals vermeld in A$ tweemaal uitgevoerd worden, maar slechts eenmaal.

10 REM verdwijnpunt / F.H. Druijff - 4/87

```
20 SCREEN 2:I=9
30 A$="r8 u10 l5 d10 r17 u20 l20 e10 f10 l20 d20"
40 DRAW "bm0,170"
50 DRAW "s=I; xA$; bm+22,-10"
60 I=I-1
70 IF I>0 GOTO 50
80 IF INKEY$<>" " GOTO 80
```

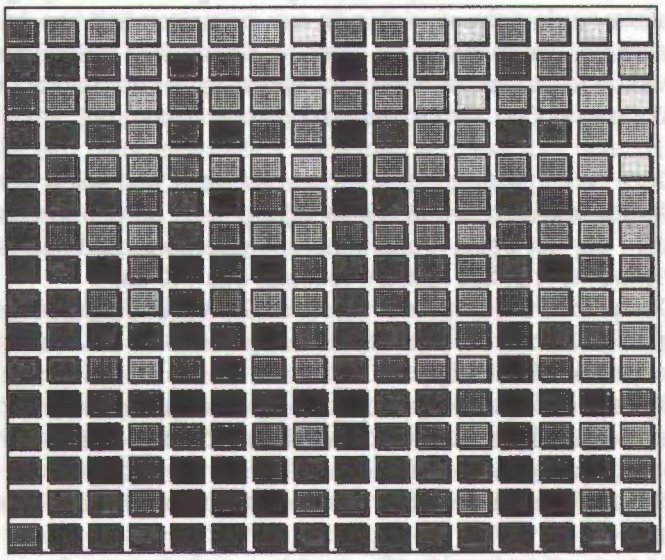
VRDWIJNP BAS D1

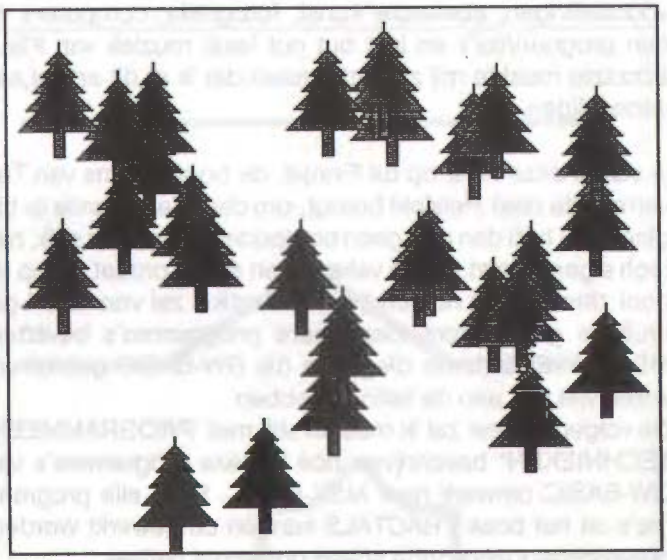
(c) MSX-CLUB

Er volgen nu nog enige voorbeeld programma's die als het goed is de lezer een aantal mogelijkheden van DRAW toepassingen laten zien. Rest mij nog een bijdrage van Wilfried v/d Eijnden bij de lezer aan te bevelen. Wilfried heeft een programma gemaakt dat de gebruiker in staat stelt direct met de cursor te tekenen. Het resultaat is niet alleen een aardige tekening (afhankelijk van de gebruiker) Maar ook de instructies om deze tekening te maken met behulp van de DRAW. Deze kunnen naar wens zowel op band als op flop weggeschreven worden en dan later in een programma worden opgenomen. Het programma is goed gestructureerd en kan door een iegelijk aan eigen behoefte worden aangepast.

PALET1 BAS D1

Veel programmeerplezier
Frank H. Druijff





BOS BAS D1

FRACTALS

Meetkundige figuren in eindeloze herhaling.

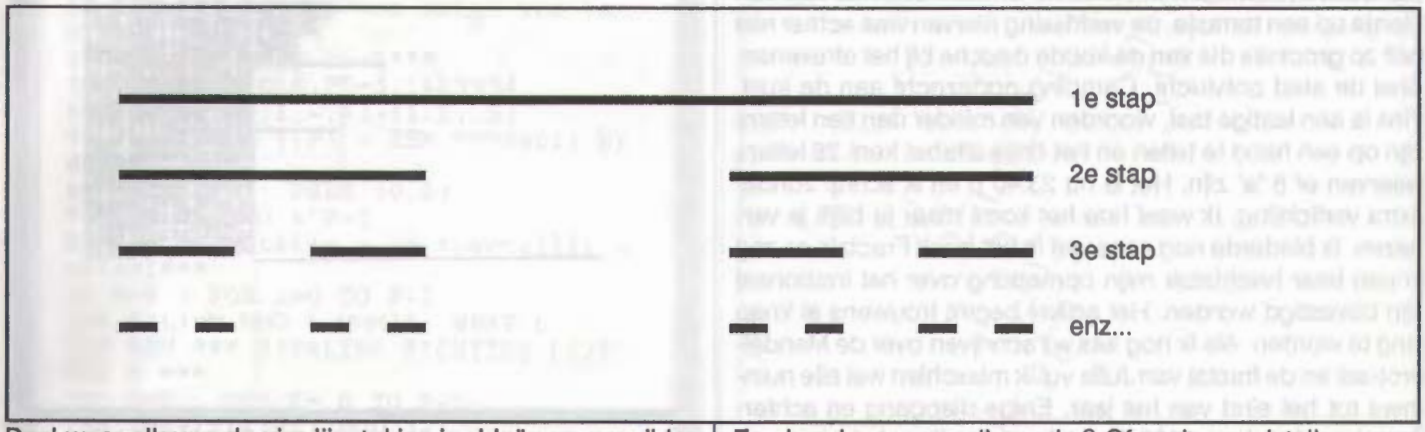
Deze titel en subtitel heb ik schaamteloos overgenomen van het boek van Hans Lauwerier. Hij is hoogleraar in de toegepaste en de zuivere wiskunde aan de universiteit van Amsterdam. Daarnaast is hij chef van de afdeling toegepaste wiskunde van het centrum voor wiskunde en informatica. De fractals hebben zijn bijzondere belangstelling.

Deze belangstelling delen wij in hoge mate. Met enige honderden anderen waren wij op vrijdag 1 juli op een symposium dat het Delfts studentencorps georganiseerd had ter viering van het 28e lustrum (bestaat dus 140 jaar).
Dit symposium ging over fractals. Er werden lezingen gehouden door voornoemde H.Lauwerier, B.B. Mandelbrot himself en H.O.Peitgen.
Bij deze lezingen enige computerdemonstraties, veel dia's en overhead displays en zelfs twee imposante - maar, jammer genoeg, korte - films.
Bij deze films muziek van - hoe kan het anders - Klaus Schulze.
De genoemde combinatie van zaken waar ik meer dan gemiddelde belangstelling voor heb (wiskunde, grafische voorstellingen, abstracte kunst, fotografie, computers en hun programma's en last but not least muziek van Klaus Schulze) maakte mij zo enthousiast dat ik er dit artikel aan wilde wijden.

Ik schrijf deze tekst op de Finnjet, de boot die ons van TravermÜnde naar Helsinki brengt, om daar de vakantie te beginnen. Ik heb dan ook geen computer bij me, snik snik, nee toch eigenlijk niet. Het is vakantie en alleen omdat we op de boot zitten kan ik nu schrijven. Dit artikel zal voor MSX-gebruikers geen direct toepasbare programma's bevatten. MS-DOSSERS (beter diegenen die GW-BASIC gebruiken) zullen wel wat aan de listings hebben.
De volgende keer zal ik meer in stijl met "PROGRAMMEERTECHNIEKEN" beschrijven hoe ik deze programma's van GW-BASIC omwerk naar MSX-BASIC. Bijna alle programma's uit het boek FRACTALS kunnen omgewerkt worden; alleen geheugengrootte speelt ons soms parten.

Maar wat zijn die fractals nu eigenlijk ? Dat is niet zo simpel in een paar woorden te zeggen, maar de subtitel geeft al een aardige aanwijzing in welke richting we het moeten zoeken. En zonder nu gelijk een historisch overzicht en wiskundige achtergronden te geven, wil ik aan de hand van wat voorbeelden de lezer enig inzicht geven wat fractals zijn. Om enthousiast te worden beveel ik het intikken(*) van de programma's aan.(* of laden met GW-BASIC of QUICKBASIC - de programma's staan in ASCII op het diskabonnement-, de volgende keer dus een aantal in 100% MSX-formaat. *)
Voor de lezer die wel voor een meer uitgebreide behandeling voelt, kan ik het boek van Hans Lauwerier van harte aanraden.
Ondanks (of juist daardoor ?) de indrukwekkende kwalificaties van Hans Lauwerier is het boek prettig leesbaar. Wiskundige kennis is vrijwel niet nodig op meer dan eenvoudig middelbaar onderwijs niveau. Onderdelen (vb. complexe getallen) die niet begrepen worden, kunnen worden overgeslagen. In het boek zelf staat de tip het net zo door te werken als men zelf leuk vindt. Wilt U gelijk zonder achtergrond achterin beginnen en programma's gaan intikken ? Best, doe dat en ga ermee spelen. D.w.z. verander eventueel wat getallen, maar kijk er niet gek van op dat programma's dan ineens niet meer kunnen werken in het beperkte computergeheugen. Of dat de tekening pas veel later klaar is en in sommige gevallen onveranderd t.o.v. de oorspronkelijke versie. Of het ergste : dat we, na bijvoorbeeld een dag rekenen om te bepalen of de punten van het beeldscherm wit of zwart moeten worden, er achter komen dat alle punten zwart zijn bij deze waarden.

Ik dwaal weer af. Ik begin met een oude fractal : de puntverzameling van Cantor (1870).
Neem een lijnstuk en deel dat in drie gelijke stukken. Stuf het middelste stuk weg. Verdeel nu het linker en rechter deel beide in drie gelijke stukken en laat het middelste stuk weer weg.



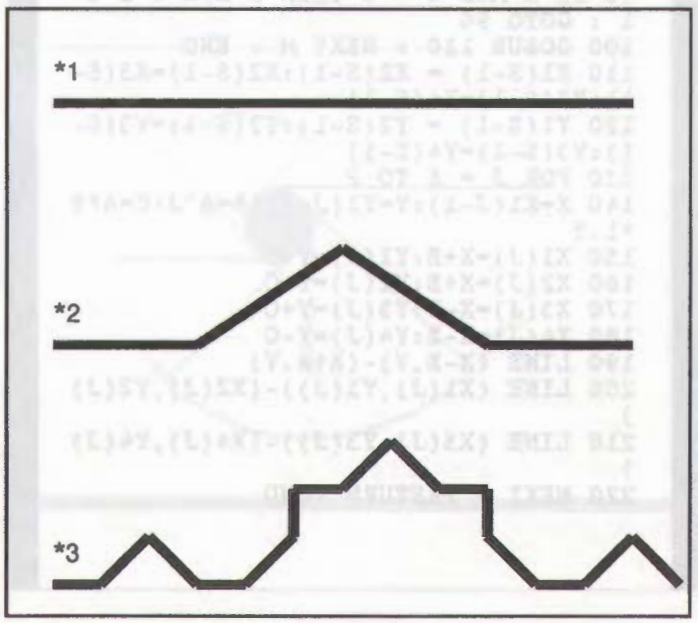
Deel weer elk van de vier lijnstukjes in drieën en verwijder het middelste stukje en zo gaan we alsmaar door. Uiteindelijk houden we alleen maar losse punten over. Deze opmerking is zeer moeilijk in te zien, want hoe kunnen we nu punten krijgen ? Het blijven toch **lijnstukjes** en geen punten ! Lauwerier gaat hier niet op in, maar laat ik dat nu eens wel doen zonder al te moeilijk te worden.
Ik ga beginnen met het lijnstuk van 0 tot 1 te laten lopen. Na de eerste stap behouden we dus het stuk 0 t/m 1/3 en het stuk 2/3 t/m 1 en verwijderd is het stuk tussen 1/3 en 2/3.
Als we afspreken de grenspunten te laten staan (heel belangrijk) zien we eenvoudig in dat het punt bij 1/3 in de verzameling zit. Na de tweede stap zit 1/3 nog steeds in de verzameling. Grenspunten blijven ! Rest nu nog in te zien dat na oneindig veel van deze drie-delingen er geen enkel getal te geven is dat met 1/3 op hetzelfde lijnstuk ligt.
Conclusie : als geen enkel punt met 1/3 samen op een lijnstuk ligt, is 1/3 een los punt.
Dezelfde beredenering gaat evenzeer voor alle andere eindpunten op. De verzameling bestaat dus uit alle punten die eindpunt zijn van een lijnstuk uit een van de stappen. We hebben dus te maken met een verzameling punten, dus dimensie 0.
Alternatieve benadering :
In de eerste stap verwijderen we 1/3, in de tweede stap 2 * 1/9, in de derde stap 4 * 1/27....
Als we deze allemaal optellen, benaderen we 1. De lengte die overblijft, wordt dus 0. Dit kan alleen als er niets of slechts losse punten overblijven.

Ik loop het risico de lezer in verwarring te brengen met de volgende opmerking : Er is binnen **elke** afstand van 1/3 altijd nog een oneindig grote hoeveelheid punten te vinden van de verzameling. Sterker : dit geldt voor elk punt van de verzameling en zo hebben we een eigenschap die aan een lijn doet denken. Dus toch geen los punt hoor ik al denken. Ja hoor, wees gerust : De verzameling is wel "vreemd", maar geen verzameling van lijnstukjes.

Een lijn of lijnstuk heeft dimensie één en met een geschikte definitie voor het begrip dimensie komen we tot dimensie 0, 6 3 0 9 ..... voor deze verzameling van Cantor.

Een breuk voor een dimensie ? Of zoals engelstaligen zeggen : een fractal dimension ? Ja, en daarom wordt deze verzameling een fractal genoemd. Een fractal is dus een verzameling die tot een niet-geheeltallige dimensie leidt. Ik vermoed trouwens sterk dat de dimensie van voornoemde verzameling geen breuk is, maar een irrationaal getal. Kolmogorov spreekt trouwens niet van dimensie maar van capaciteit. Aangezien we het hier hebben over verzamelingen met een niet-geheeltallige (dus fractal) capaciteit doet dit niets af aan de betekenis van het begrip fractal.
Het programma "KAM" maakt de verzameling van Cantor redelijk zichtbaar op het scherm.

Vele fractals zijn te herkennen aan het zichzelf verkleind repeteren van de basisfiguur. Bij de boom van Pythagoras splitst de stam zich in twee takken die elk ook weer een boom van Pythagoras zijn. Dus splitsen die zich elk in twee takken die ... enz. Om praktische redenen beperken we in computerprogramma's het aantal stappen. Bij Koch (een van de weinige vrouwelijke wiskundigen) wordt een lijnstuk (*1) als volgt bewerkt (*2)en dan elk van de vier lijnstukken in de nu ontstane figuur met hetzelfde algoritme bewerkt. (3*)

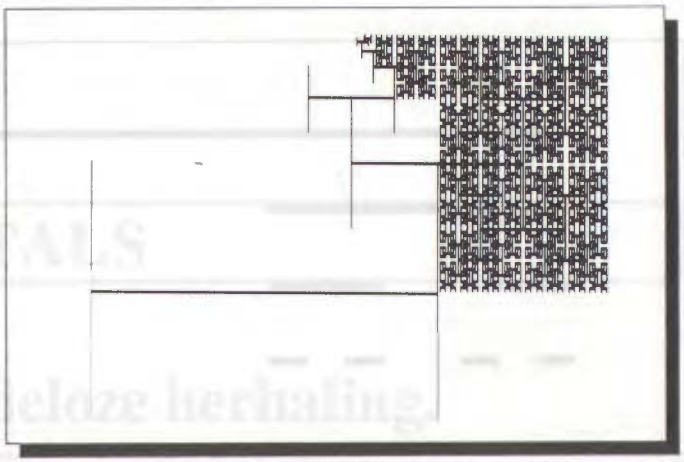


Zo, even ontschepen in Helsinki. Smoorheet, dus rap een biertje op een terrasje, de verfrissing hiervan was echter niet half zo groot als die van de koude douche bij het afrekenen. Snel de stad ontvlucht. Camping opgezocht aan de kust. Fins is een lastige taal, woorden van minder dan tien letters zijn op een hand te tellen en het finse alfabet kent 28 letters waarvan er 8 'a' zijn. Het is nu 23.40 u en ik schrijf zonder extra verlichting. Ik weet hoe het komt maar je blijft je verbazen. Ik bladerde nog eens wat in het boek Fractals en zag in een later hoofdstuk mijn opmerking over het irrationaal zijn bevestigd worden. Het artikel begint trouwens al knap lang te worden. Als ik nog iets wil schrijven over de Mandelbrot-set en de fractal van Julia vul ik misschien wel alle nummers tot het eind van het jaar. Enige diepgang en achtergrond en het jaar 2000 komt in zicht. Ik zal daarom nu volstaan met het geven van enige listings. Het zijn programma's waarvan ik weet dat ze na enige simpele ingrepen ook op MSX draaien. De volgende keer bespreek ik dan de opmerkmethode en geef de listings voor MSX. Delen meerderen in de vereniging mijn belangstelling voor fractals zal ik nog wat meer over dat onderwerp vertellen. Laat dit dus even horen.

LIST1 BAS D1

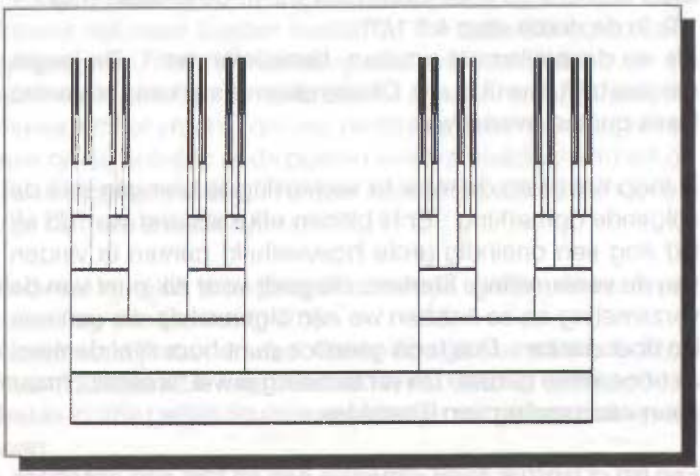
Filp

```
10 REM ***boomfractal op letter h ,
 backtrack methode***
20 REM ***naam:boomh2***
30 SCREEN 2 : CLS
40 WINDOW (-1.2,-.9)-(1.2,.9)
50 P=7 : DIM X1(P),X2(P),X3(P),X4(P
),Y1(P),Y2(P),Y3(P),Y4(P)
60 A=.5 : REM ***keuze verkortingsv
erhouding***
70 X1(0)=0 : Y1(0)=0 : S=1 : GOSUB
130
80 FOR M=1 TO 4^(P-1)-1 : N=M : S=P
90 IF N MOD 4 = 0 THEN N=N/4 : S=S-
1 : GOTO 90
100 GOSUB 110 : NEXT M : END
110 X1(S-1) = X2(S-1):X2(S-1)=X3(S-
1):X3(S-1)=X4(S-1)
120 Y1(S-1) = Y2(S-1):Y2(S-1)=Y3(S-
1):Y3(S-1)=Y4(S-1)
130 FOR J = S TO P
140 X=X1(J-1):Y=Y1(J-1):B=A^J:C=A*B
*1.5
150 X1(J)=X+B:Y1(J)=Y+C
160 X2(J)=X+B:Y2(J)=Y-C
170 X3(J)=X-B:Y3(J)=Y+C
180 X4(J)=X-B:Y4(J)=Y-C
190 LINE (X-B,Y)-(X+B,Y)
200 LINE (X1(J),Y1(J))-(X2(J),Y2(J)
)
210 LINE (X3(J),Y3(J))-(X4(J),Y4(J)
)
220 NEXT J :RETURN :END
```



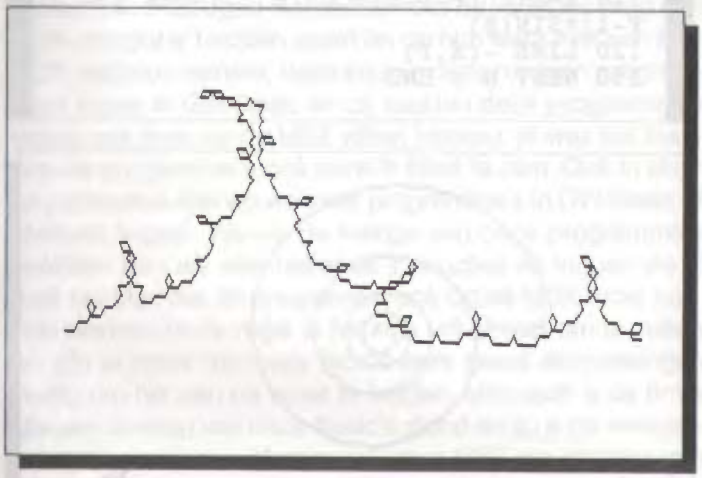
```
10 REM ***cantor kam***
20 REM ***naam:kam***
30 SCREEN 3     :CLS
40 WINDOW (-.3,-.4)-(1.3,.8)
50 DIM A(729),B(729):A(0)=0:A(1)=1
60 B=.1:LINE(0,0)-(1,0):LINE -(1,-B
):LINE -(0,-B):LINE -(0,0)
70 FOR P = 1  TO 6
80 FOR I=0 TO 2^P-1
90 B(I)=A(I)/3:B(I+2^P)=1-(1-A(I))/
3
100 NEXT I
110 FOR J= 1 TO 2 ^ (P+1)-1
120 A(J)=B(J):NEXT J
130 FOR K=0 TO 2^ (P+1)-1 STEP 2
140 LINE (A(K),B*P)-(A(K+1),B*P)
150 LINE (A(K),B*P)-(A(K),B*P-B):LI
NE (A(K+1),B*P)-(A(K+1),B*P-B)
160 NEXT K :NEXT P : END
```

LIST2 BAS D1



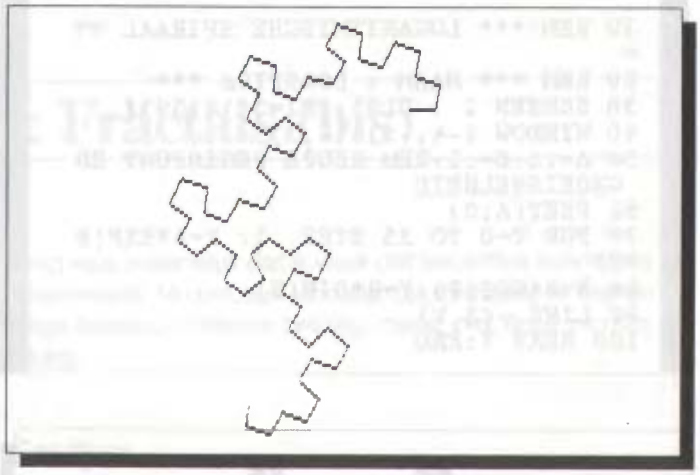
LIST3 BAS D1

```
10 REM *** kromme van helge von koc
h***
20 REM *** naam:koch***
30 SCREEN 3:CLS:PI=3.141593#
40 WINDOW (-.1,-.4)-(1.1,.5)
50 P=4 : DIM T(P) : REM ***keuze or
de***
60 H=3^(-P) : PSET (0,0)
70 FOR N=0 TO 4^P-1
80 REM ***notatie n in viertallig s
telsel***
90 M=N : FOR L=0 TO P-1
100 T(L)=M MOD 4:M=M/4: NEXT L
110 REM *** BEPALING RICHTING LIJNS
TUK N ***
120 S=0 : FOR K= 0 TO P-1
130 S=S+(T(K)+1) MOD 3-1
140 NEXT K
150 REM *** TEKENING LIJNSTUK N ***
160 X=X+COS(PI*S/3)*H
170 Y=Y+SIN(PI*S/3)*H
180 LINE -(X,Y)
190 NEXT N: BEEP: A$=INPUT$(1):END
```



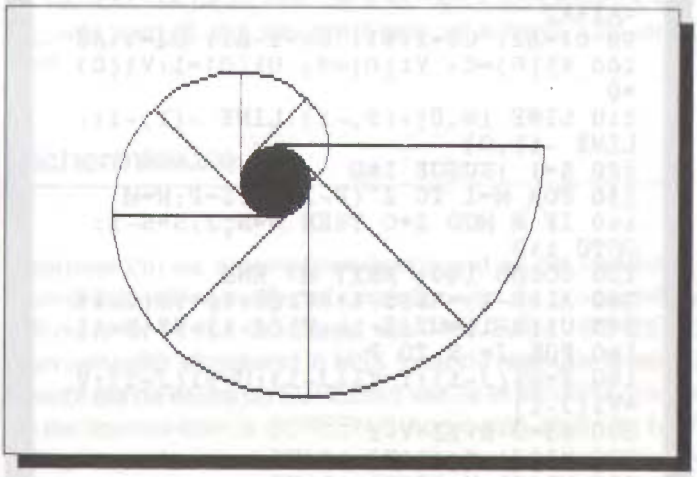
```
10 REM *** DRAAKKROMME MET GEGEVEN
HOEK ***
20 REM *** NAAM : DRAAK ***
30 SCREEN 2 : CLS :PI=3.141593#
40 WINDOW (-2.5,-2)-(1.5,1)
50 P=6 : REM *** KEUZE ORDE ***
60 H=2^(-P/2) : A=1.7453 : REM ***
KEUZE HOEK ***
70 B=PI-A:X=H:Y=0:LINE (0,0)-(H,0):
S=0
80 FOR N=1 TO 2^P-1: M = N
110 IF M MOD 2 = 0 THEN M=M/2 : GOT
O 110
120 IF M MOD 4 = 1 THEN D=1 ELSE D=
 -1
130 S=(S+D)
140 X=X+H*COS(S*B)
150 Y=Y+H*SIN(S*B)
160 LINE -(X,Y)
170 NEXT N : END
```

LIST4 BAS D1



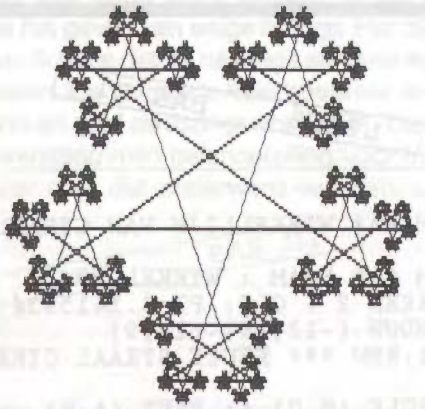
LIST5 BAS D1

```
10 REM *** WIKKELLIJN VAN CIRKEL **
*
20 REM *** NAAM : WIKKEL ***
30 SCREEN 2 : CLS: PI=3.141593#
40 WINDOW (-12,-9)-(12,9)
45 A=1:REM *** KEUZE STRAAL CIRKEL
***
50 CIRCLE (0,0),A: PSET (A,0)
60 FOR N= 0 TO 100: T=2*PI*N/80
70 X=A*(COS(T)+T*SIN(T))
80 Y=A*(SIN(T)-T*COS(T))
90 LINE -(X,Y)
100 IF N MOD 10 = 0 THEN LINE (COS(
T),SIN(T))-(X,Y)
110 NEXT N:PAINT (0,0):END
```



LIST6 BAS D1

```
10 REM *** LOGARITMISCHE SPIRAAL **
*
20 REM *** NAAM : LOGSPIRA ***
30 SCREEN 2 : CLS : PI=3.141593#
40 WINDOW (-4,-3)-(4,3)
50 A=.1:B=.1:REM KEUZE BEGINPUNT EN
 GROEISNELHEID
60 PSET(A,0)
70 FOR T=0 TO 35 STEP .1: R=A*EXP(B
*T)
80 X=R*COS(T):Y=R*SIN(T)
90 LINE -(X,Y)
100 NEXT T:END
```

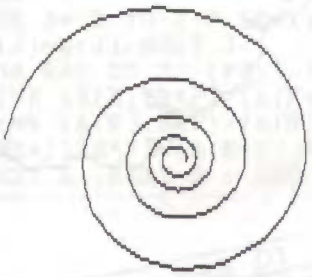


LIST7 BAS D1

```
10 REM *** PYTHAGORASBOOM, BACKTRAC
KMETHODE ***
20 REM *** NAAM :PYTHB3 ***
30 SCREEN 2 : CLS :PI=3.141593#
40 WINDOW (-5,-3)-(5,4.5)
50 P=12:DIM X1(P),Y1(P),X2(P),Y2(P)
,U1(P),V1(P),U2(P),V2(P)
60 REM *** KEUZE HOEK ***
70 F=PI/5 : C=COS(F) : S= SIN(F)
80 A1=-C*S : A2=C^2 : B1=A1+A2: B2=
-A1+A2
90 C1=B2: C2=1-B1: D1=1-A1: D2=1-A2
100 X1(0)=0: Y1(0)=0: U1(0)=1:V1(0)
=0
110 LINE (0,0)-(0,-1):LINE -(1,-1):
LINE -(1,0)
120 S=1 :GOSUB 180
130 FOR M=1 TO 2^(P-1)-1:S=P:N=M
140 IF N MOD 2=0 THEN N=Nç2:S=S-1:
GOTO 140
150 GOSUB 160: NEXT M: END
160 X1(S-1)=X2(S-1):Y1(S-1)=Y2(S-1)
170 U1(S-1)=U2(S-1):V1(S-1)=V2(S-1)
180 FOR J= S TO P
190 X=X1(J-1):Y=Y1(J-1):U=U1(J-1):V
=V1(J-1)
200 X3=U-X:Y3=V-Y
210 X1(J)=X+A1*X3-A2*Y3
220 Y1(J)=Y+A2*X3+A1*Y3
230 U1(J)=X+B1*X3-B2*Y3
240 V1(J)=Y+B2*X3+B1*Y3
250 X2(J)=X+C1*X3+C1*Y3
270 U2(J)=X+D1*X3-D2*Y3
280 V2(J)=Y+D2*X3+D1*Y3
290 LINE (X,Y)-(X1(J),Y1(J)):LINE -
(U1(J),V1(J)): LINE - (U,V)
300 LINE -(X,Y): LINE -(X2(J),Y2(J)
): LINE -(U2(J),V2(J)) : LINE -(U,V
)
310 NEXT J: RETURN : END
```

LIST8 BAS D1

```
10 REM *** STERFRACTAL ***
20 REM *** NAAM : STER ***
30 SCREEN 2 : CLS : PI=3.141593#
40 WINDOW (-.5,-.8)-(1.5,.7)
50 P=5 : V=4 : A=144:R=.35:A=A*PI/1
80
60 PSET (0,0) : X=0 : Y=0
70 FOR N= 0 TO (V+1)*V^(P-1)-1
80 M=N : B=N*A : F=0
90 IF M MOD V <>0 OR F>=P-1 THEN GO
TO 110
100 F=F+1: M=INT(M/V): GOTO 90
110 X=X+R^(P-F-1)*COS(B): Y=Y+R^(P-
F-1)*SIN(B)
120 LINE -(X,Y)
130 NEXT N : END
```



LIST9 BAS D1

```
10 REM *** MODERNE KUNST ***
20 REM *** NAAM :MONDRIAAN ***
30 SCREEN 2 : CLS : RANDOMIZE 1
40 H=.1: REM KEUZE SCHALINGSCONSTAN
TE
50 WINDOW (-.3,-.1)-(1.3,1.1)
60 LINE (0,0)-(1,0)
70 LINE -(1,1):LINE -(0,1):LINE-(0,
0)
80 FOR N=1 TO 100
90 X=.8*INT(100*RND)/100+.1
100 Y=.8*INT(100*RND)/100+.1
110 K=H*(1-SQR(RND))
120 IF RND>.5 THEN GOSUB 140 ELSE G
OSUB 150
130 NEXT N : END
140 LINE (X-K,Y)-(X+K,Y):RETURN
150 LINE (X,Y-K)-(X,Y+K): RETURN :
END
```

Programmeer Technieken : Fractals (bis)

Reacties

Ik heb nogal wat reacties gekregen op mijn vorige artikel over fractals. De aard van deze reacties was nogal uiteenlopend zodat ik er hier eerst nog eens op wil ingaan.

MSX en geen MS-DOS

Wij zijn een MSX-Club en geen MS-DOS club. Wij willen echter ook onze ogen niet sluiten voor de velen die thuis een MSX-computer hebben staan en op hun werk met een MS-DOS machine werken. Vaak komen deze mensen programma's tegen in GW-Basic en zij zouden deze programma's graag ook thuis op de MSX willen hebben. Al was het maar om die programma's ook eens in kleur te zien. Ook in allerlei publicaties komen we vaak programma's in GW-Basic of M-Basic tegen. Als we de listings van deze programma's bekijken zien we vele bekende instructies en krijgen we al snel het idee dat dit programma ook op de MSX moet kunnen werken. In de regel is het ook vrij simpel om te zetten en zijn er maar minimale (soms zelfs geen) aanpassingen nodig om het aan de praat te krijgen. Microsoft is de firma die aan de wieg van deze Basic's stond en zo is de verwantschap te verklaren. Komen wij voor MSX-ers interessante programma's tegen zullen wij die zeker publiceren. De programma's, die ik de vorige keer bij mijn artikel had staan, waren alle in GW-Basic en met toestemming van de auteur van 'Fractals', Hans Lauwerier, overgenomen uit diens boek. Voor de diskabonnementen waren de listings op schijf meegeleverd; al zullen maar weinigen er direct iets aan gehad hebben. De meeste MS-Dos machines hebben nog een vijf en een kwart inch drive en niet een drie en een half inch. Wij hebben het er echter toch opgezet om de meer actieve lezers de mogelijkheid te geven het zelf aan te passen.

Biertje

De tweede reactie was niet zo zeer een bezorgde maar eerder een nieuwsgierige. Vrijwel zonder uitzondering vroeg een ieder die ik sprak naar de prijs van dat biertje in Helsinki. Triest dat dat blijkbaar het meest intrigerende in mijn betoog was, maar leuk dat ik weet dat het artikel tenminste gelezen wordt. Nu om een eind aan de discussie te maken het pilsje kostte 21 (een en twintig) mark. Wel finse marken gelukkig.

Foutjes

In de listings waren jammer genoeg bij het intikken wat foutjes geslopen. In een aantal gevallen was er een integerdeling die als normale deling genoteerd stond.
Het betreft de programma's :

BOOMH2 in regel 90 (nr 19 blz 6 links)
KOCH in regel 100 (nr 19 blz 7 linksboven)
PYTHB3 in regel 140 (nr 19 blz 8 linksonder)

Vervang in deze regels de slash (streep van linksonder naar rechtsboven) door een backslash (streep van linksboven naar rechtsonder) en het probleem is weg. Bij PYTHB3 stond trouwens een c met cedille die nu backslash moet worden.
In regel 250 van PYTHB3 was een fout geslopen en regel 260 zelfs volledig weggevallen dus nu eerst de correcte regels :

```
250 X2(J)=X+C1*X3-C2*Y3
260 Y2(J)=Y+C2*X3+C1*Y3
```

Een ander klein foutje betrof de schermkeuze. Die moet in alle gevallen scherm 3 zijn. Controleer de regels 30 hierop. Ik vermoed dat de intikker zelf al aan MSX-aanpassingen bezig was toen hij zich een enkel keer tot scherm 2 liet verleiden.

Schermkeuze

Hiermee zijn we automatisch aangeland bij het onderwerp van deze keer namelijk het omzetten van de programma's uit nummer 19 van GW-Basic naar MSX-Basic. SCREEN 3 kan natuurlijk uitstekend in MSX maar de resolutie is dan zo slecht dat de keuze op SCREEN 2 valt. Is er een MSX-2 computer voorhanden is SCREEN 5 nog beter, maar dit heeft alleen voor de schermsnelheid en niet voor de resolutie gevolgen. Geef ik straks dus een aanpassing naar SCREEN 5 dan kunnen MSX-1'ers daar gewoon SCREEN 2 van maken

en omgekeerd kunnen MSX-2'ers de genoemde SCREEN 2 in SCREEN 5 of zelfs SCREEN 8 veranderen. Soms zal ik echter een hogere horizontale resolutie willen gebruiken en dan voor scherm 6 of 7 kiezen. In die gevallen kan alleen gekozen worden voor scherm 2,5 of 8 als we alle voorkomende x-coördinaten eerst halveren. Mijn keuze voor de hogere resolutie is echter niet voor niets gedaan en het resultaat zal meestal niet fraai zijn. Het zal nodig blijken te zijn om minstens een stap minder diep te gaan om het resultaat nog aangenaam voor het oog te houden.

Vastlopen

Velen die een diskabonnement hebben zullen vrijwel zeker getracht hebben de programma's te laten lopen op hun MSX. In vrijwel alle programma's staat echter al in regel 40 de instructie 'WINDOW' die MSX-Basic niet kent. We krijgen dus een foutmelding op regel 40 als we het programma toch starten. Verwijderen we 'WINDOW' in regel 40 door er bijvoorbeeld een ' voor te zetten zullen de meeste programma's wel werken, maar alleen na zeer lange tijd zullen slechts enige punten in de uiterste linkerbovenhoek gezet worden.

WINDOW

De instructie 'WINDOW' uit GW-Basic is een voor de programmeur zeer aangename. Je geeft er het kader mee aan waar binnen je wilt werken. De basic-interpreter zal er dan voor zorgen dat alles toch beeldvullend wordt. Ik zal het met een voorbeeld verduidelijken : stel je wilt een sinusfunctie zo groot mogelijk op het beeldscherm laten zien dus van -pi tot +pi op de horizontale as en van -1 tot +1 op de vertikale as. Om nog iets ruimte rondom over te hebben neem je voor pi niet een hele nauwkeurige waarde maar een ruime benadering als bijvoorbeeld 3,2 . Om nog enige teksten te plaatsen neem je ook niet precies 1 maar zeg 1,2 . Je bepaalt de te tekenen punten met hun juiste wiskundige waarden en zet die dan normaal op het scherm met een LINE instructie. Doen wij dit op een MSX-computer of beter gezegd op een computer die een MSX-basic interpreter gebruikt zullen alle punten/lijnen getekend worden binnen het vlak van (0,0) tot (3,1). Driekwart van de grafiek valt buiten het beeldscherm en de punten die het halen mogen vechten om de 8 (inderdaad acht) beschikbare punten.
In GW-Basic geven we gewoon aan het begin van ons programma op : WINDOW (-3.2,-1.2)-(3.2,1.2) en de interpreter zorgt er voor dat het punt met coördinaten (-3.2,-1.2) linksonder op het beeldscherm geplaatst wordt en het punt (3.2,1.2) rechtsboven.

In MSX-basic hebben wij deze instructie niet en zullen we de aanpassing dus zelf moet doen. Wat doet de interpreter nu precies ? Hij verschuift indien nodig het raam naar het midden van het beeldscherm en blaast het dan op tot de gewenste grootte. Vooral door dat laatste is de WINDOW ook weer niet zo prettig in onervaren handen. De vergroting gebeurt in horizontale en vertikale richting maar dat zal in het algemeen niet met dezelfde factor zijn. Opletten dus.

Aanpassen

We gaan uit van de WINDOW (-3.2,-1)-(3.2,1) en willen dit aanpassen voor scherm 2. We bepalen dan als eerste de horizontale maat van het window, dus de afstand van -3.2 tot 3.2 , hier is de waarde 6.4. We bepalen dan de vermenigvuldigingsfactor voor ons SCREEN 2. Op dit scherm hebben we horizontaal 256 (NIET 255) punten, de factor is dus 256 / 6.4 = 40. De vertikale maat is 2 en de hieruit volgende vermenigvuldigingsfactor is 192 / 2 = 96. De horizontale en de vertikale factor verschilt dus behoorlijk en we kunnen een grote vertekening verwachten. Willen we geen vertekening zullen we ons moeten houden aan de kleinste factor. Een deel van het scherm wordt dan niet gebruikt maar er is geen vertekening. Dit laatste afgezien van de standaard vertekening van MSX. De maat is nu aangepast aan het scherm maar de positie klopt nog niet. We verplaatsen de tekening nu door uit te rekenen waar het punt linksonder in de "window" komt en verplaatsen dat dan naar (0,0) Het window hier had als punt linksonder (-3.2,-1) na een vermenigvuldiging met 40 voor x en met 96 voor y wordt dit punt (-128,-96) en om dit punt op de MSX-oorsprong te krijgen moeten we de x met 128 en de y met 96 verhogen. Bevat het GW-basic programma de instructie LINE -(P,Q) moet dat in het aangepaste MSX-basic programma LINE (128+40*P,96+96*Q) worden. Willen we de tekening niet op zijn kop laten staan dan LINE (128+40*P,96-96*Q).

Geheugengrootte

Het volgende probleem kwam bij het eerste programma (BOOMH2) van de vorige keer. De MSX had normaal te weinig geheugen om alle gegevens op te slaan. Met enige trucs was daar misschien nog wel een mouw aan te passen maar dan draait het programma alleen op de 'grote' machines. We passen liever de diepgang van het programma aan. Bedenk dat we toch al altijd een beperkt deel uitrekenen bij fractals en of we nu tien of twaalf stappen doorgaan betekent meestal alleen maar een veel langer wachten tot het klaar is en dan als beloning voor dat langere wachten nog geeneens een fraaiere tekening. Door afrondingsfouten kunnen lijnen er lelijk uit gaan zien en kunnen patronen onge-

wenst dicht lopen. Voor het programma BOOMH2 betekende dit P = 6 in plaats van P = 7 in regel 50. Tot slot moesten dan nog de tekeninstructies aangepast te worden. Het betrof de regels 190,200 en 210 waar een LINE in stond. Voor de vermenigvuldigingsfactor heb ik hier gekozen voor 128. Dit is een macht van twee en dat is altijd aan te raden als vermenigvuldigingsfactor om afrondingsproblemen te voorkomen. Voor de horizontale verplaatsing nam ik 129 en voor de vertikale 96 en liet de tekening braaf op zijn kop staan omdat dat bij deze fractal niet erg is. (symmetrisch)

Het werkt wel maar hoe ?

We laten het programma na deze aanpassingen lopen en stellen vast dat het inderdaad werkt. Maar het duurt wel lang. Als we het programma nog eens bekijken zien we ook wel in dat we ons slaafje niets kunnen kwalijk nemen. We laten hem ook vreselijk veel doen. Als we alleen al kijken naar de regels 190,200 en 210 waar we door de raamlacune een aanpassing hebben moeten maken, die het nodig maakt voor elk lijntje dat getrokken gaat worden al acht berekeningen extra te maken. Dit een vertraging die we aan de aanpassing te danken hebben en ik wil die kwijt raken. We bestuderen het programma wat beter en zien in dat we natuurlijk de waarden die voor de coördinaten gebruikt gaan worden ook al van te voren op de juiste (voor MSX I) waarden in de arrays kunnen plaatsen. Het is daarvoor nodig de startwaarden in regel 70 anders te kiezen. Verder zien we dat de oorspronkelijke programmeur zich veel aangetrokken heeft van de voorwaarde voor algemeenheid. Voor een MS-DOScomputer ook zinnig omdat er straks misschien een uitbreiding komt om met een dubbele (of meer) beeldresolutie komt en een veel snellere processor waarmee een of meer stappen verder graven ineens wel kan en zinnig is. Voor MSX is dit argument wel geldig maar niet zo sterk. De snelheidswinst die nu geboekt kan worden is voor mij belangrijker en ik ga verder met aanpassen.

TWAALF MAAL S-1

in de regels 110 en 120 staat in totaal twaalf keer de berekening 'S-1'. Dit is weliswaar een hele eenvoudige berekening, maar eenmaal S = S-1 voor deze twaalf en dan nog eenmaal S = S + 1 scheelt tien berekeningen. Ik ga echter verder en kijk waar de variabele S nog meer gebruikt wordt. In principe is het mogelijk om op alle andere plaatsen waar in het programma S gebruikt wordt de waarde een kleiner te nemen zodat in de regels 110 en 120 we gewoon S in plaats van S-1 kunnen gebruiken. Het blijkt dat dit inderdaad de moeite loont.

Minder algemeen

Nu ga ik nog de algemeenheid van het programma aantasten door bepaalde waarden die uitgerekend worden als constanten in te voeren. De P = 6 bijvoorbeeld laat ik helemaal weg en vervang de P overal in het programma door 6, ook berekeningen waar deze 6 in voorkomt vereenvoudig ik zoveel mogelijk. Daarna moet ook de A = 0.5 er aan geloven en weer wordt het programma (iets) sneller maar ook minder algemeen.

In de marge

Er zijn nog wel meer zaken te vinden maar die vertegenwoordigen naar mijn idee alleen rommelen in de marge. Heb ik een echt grote over het hoofd gezien, laat mij dit dan weten. Ik zal er nog enige noemen : In regel 190 wordt als x-coördinaat X-B gebruikt terwijl in de regel erboven deze waarde net wordt uitgerekend en opgeslagen in X4(J). We zouden dus in regel 190 ook X4(J) kunnen gebruiken. Het opzoeken in een geïndiceerde array is echter trager dan het direct opzoeken van een variabele. Daar staat tegenover dat er geen berekening meer hoeft te worden gedaan. Aftrekken (en optellen) behoort echter tot de snelst uitgevoerde berekeningen. Het zal vermoedelijk iets sneller gaan maar of iemand het ziet? De berekening X + B wordt trouwens tweemaal vlak achter elkaar in de regels 150 en 160 uitgevoerd. We kunnen dus ook eerst XP = X + B geven en dan de waarde van de XP aan X1(J) en X2(J) toewijzen. Dit geldt evenzo voor X-B, Y + C en Y-C. De FOR-NEXT lussen zullen sneller doorlopen worden als we ze beëindigen met alleen NEXT zonder variabele. De winst hiervan is echter ook minimaal. Willen we het programma behoorlijk groter maken kan nog enige winst geboekt worden in het vermijden van de GOSUB en RETURN. Daartoe dienen we het stuk van regel 110 tot en met de NEXT van regel 220 in plaats van de 'GOSUB 110' van regel 100 te plaatsen. Het simpelst doen we dat door twee NEXT'n in regel 220 en laat regel 100 maar vervallen. Vergeet echter niet dat de FOR-NEXT ook moet komen in plaats van de GOSUB 130 uit regel 70. De snelheid van deze oplossing zal hoger zijn maar nu is de listing een stuk groter.

END en PI

Een slordigheidje in het programma is dat er twee 'END'-instructies in staan. Bij andere programma's zien we PI gedefinieerd worden terwijl die in het gehele programma niet gebruikt wordt. Ik vindt de algemeenheid in aanpak dan wel

enigszins te veel van het goede. Om PI te gebruiken zal ik zelf trouwens altijd PI = ATN(1)*4 programmeren en niet een afronding als PI = 3.141593. Zeker in MSX-basic is de PI zoals met de arctangens verkregen veel nauwkeuriger.

Extra's

Bij sommige programma's heb ik nog wat extra's toegevoegd. Kleur ! In sommige gevallen is de toevoeging van kleur een leuk extra. MSX-1ers moeten echter oppassen; in scherm 5 levert het geen problemen op maar in scherm 2 wel ! Kan het niet op scherm 5 gedraaid worden en moet het naar scherm 2, laat de kleur dan vallen. Draai het programma eenmaal met kleur en zie hoe triest. De keuze van voor- en achtergrondkleur is vanzelfsprekend wel vrij. De listing is soms ook zeer veel kleiner dan het origineel, maar bedenk dat dit ten koste ging van algemeenheid. In het blad zijn de korte MSX-listings opgenomen. Op de diskette staan ook de minimaal aangepaste versies. Alleen van het hier besproken programma BOOMH2 staan beide listings in het blad om het verhaal beter te kunnen volgen.

Uitsmijter

Ik wilde besluiten met een simpele manier om te zorgen dat de beeldverhoudingen redelijk kloppen met de werkelijkheid.
VDP(10) = VDP(10) XOR 2
zal het scherm in de vertikale richting zo uitrekken dat er vrijwel geen balken boven en onder in beeld staan. Om de normale situatie te herstellen moet dezelfde opdracht nogmaals gegeven worden.

Veel programmeer plezier
Frank H. Druijff

```
10 REM *** STERFRACTAL ***
20 REM *** NAAM : STER ***
30 SCREEN 5
50 P=5 : V=4 : A=144:R=.35:A=A*ATN(1)/
45
60 PSET (32,90) : X=0 : Y=0
70 FOR N=0 TO (V+1)*V^(P-1)-1
80 M=N : B=N*A : F=0
90 IF F<P-1 THEN IF M MOD V=0 THEN F=F
+1: M=M\V : GOTO 90
110 RP=R^(P-F-1) : X=X+RP*COS(B): Y=Y+
RP*SIN(B)
120 LINE -(32+212*X,90+212*Y) : NEXT
130 A$=INPUT$(1)
```

STERKRT

```
10 REM boomfractal op de letter H
20 REM naam :boomH1
30 DEFSNG A-Z:DIM X(2048),Y(2048)
40 SCREEN 5:CLS
50 P=9 : A=SQR(.5)
60 FOR M=0 TO P : AM=A^M : S=(M MOD2)*
AM : T=AM-S : FOR N=2^M TO N*2-1 : N2=
N+N
70 X(N2)=X(N)+T : Y(N2)=Y(N)+S : X(N2+
1)=X(N)-T : Y(N2+1)=Y(N)-S : NEXT : NE
XT
80 FOR N=2 TO 2^(P+2)-2 STEP 2
90 LINE (128+64*X(N),106+64*Y(N))-(128
+64*X(N+1),106+64*Y(N+1)):NEXT
100 GOTO 100
```

BMH1KRT

```
10 REM ***boomfractal op letter h , ba
cktrack methode***
20 REM ***naam:boomh2***
30 SCREEN 5 : CLS
40 DIM X1(6),X2(6),X3(6),X4(6),Y1(6),Y
2(6),Y3(6),Y4(6)
50 X1(0)=129 : Y1(0)=96 : S=0 : GOSUB
110
60 FOR M=1 TO 1023 : N=M : S=5
70 IF N MOD 4 = 0 THEN N=N\4 : S=S-1 :
 GOTO 70
80 GOSUB 90 : NEXT M : END
90 X1(S)=X2(S):X2(S)=X3(S):X3(S)=X4(S)
100 Y1(S)=Y2(S):Y2(S)=Y3(S):Y3(S)=Y4(S
)
110 FOR J=S+1 TO 6
120 X=X1(J-1):Y=Y1(J-1):B=2^(7-J):C=B*
.75
130 X1(J)=X+B:Y1(J)=Y+C
140 X2(J)=X+B:Y2(J)=Y-C
150 X3(J)=X-B:Y3(J)=Y+C
160 X4(J)=X-B:Y4(J)=Y-C
170 LINE (X-B,Y)-(X+B,Y)
180 LINE (X1(J),Y1(J))-(X2(J),Y2(J))
190 LINE (X3(J),Y3(J))-(X4(J),Y4(J))
200 NEXT J :RETURN :END
```

BMH2KRT

```
10 REM *** LOGARITMISCHE SPIRAAL ***
20 REM *** NAAM : LOGSPIRA ***
30 SCREEN 5 : PSET (130,95)
40 A=3.2:B=.1:REM KEUZE BEGINPUNT EN G
ROEISNELHEID
50 FOR T=0 TO 35 STEP .1: R=A*EXP(B*T)
60 LINE -(128+R*COS(T),96+R*SIN(T)) :
NEXT
70 A$=INPUT$(1)
```

LOGSPKRT

SCROLLING OP MSX

INLEIDING - MOTIVATIE

In dit en de komende artikelen, zal ik een aantal grafische- en geluidstechnieken uiteenzetten, die ik tijdens het programmeren op mijn TOSHIBA HX-10 (MSX-I) computer ontdekt heb. Ze zijn alle vanuit BASIC te realiseren, hoewel er ook in machinetaal routines voor te maken zijn. Daar ga ik echter nu niet verder op in. De routines en alle overige informatie zijn gericht op de MSX-I schermen zullen probleemloos op MSX-II werken als u op SCREEN 0 maar de 40-koloms mode aanhoudt.

Het is in geen geval mijn bedoeling hier een cursus BASIC van te maken, en ik vermoed ook heel sterk dat daar nauwelijks meer behoefte aan zal zijn vanwege het grote aanbod BASIC leerboeken en cursussen.
De in dit artikel besproken trucs zijn beslist niet noodzakelijk voor het schrijven van een goed programma maar wel handig als aanvulling op de programmeer-kennis van degenen die er wat aan zouden kunnen hebben.

Ik kan niet garanderen dat niets van de kern uit mijn artikelen staat vermeld in MSX-boeken; van de tijdschriften ben ik wat dit betreft wel redelijk zeker. In ieder geval heb ik zelf geen enkele kant en klare informatie uit ander werk gebruikt, maar berust dit werk geheel op eigen ervaringen.
De kennis van BASIC die ik bezit, heb ik mijzelf grotendeels aangeleerd, o.m. door listings te bestuderen, en ik heb dus nauwelijks BASIC-publikaties in boekvorm gelezen. Ik hoop dat u iets aan dit artikel heeft en wens u er in ieder geval veel sterkte mee.

INLEIDING OP DE SCROLLING OP MSX EN DE SCROLLING OP SCREEN 0

SCREEN 0 wordt niet of nauwelijks gebruikt voor grafische toepassingen (hooguit in BASICODE-II programma's), en dat is ook niet erg verwonderlijk. Er zijn uiteraard veel te veel beperkingen op dit scherm om van enig grafisch nut te kunnen zijn. Niettemin werken veel serieuzere programma's wel op dit scherm, omdat er een maximaal aantal (40 of 80 bij MSX-I resp. MSX-II) karakters (=lettertekens) goed leesbaar op het scherm gezet kunnen worden.
Wat de MSX-I schermen betreft: op SCREEN 2 is dit maximum van 40 ook wel haalbaar maar het gaat iets lastiger en vooral veel trager, zeker vanuit BASIC.

Tweekleurige tekeningen op SCREEN 0 zijn in principe wel mogelijk door de karakterset aan te passen, maar dit gaat op SCREEN 1 veel efficiënter met bovendien aanzienlijk meer kleur-mogelijkheden en dit zien we dan ook in veel spellen (deze SCREEN 1-spellen zijn herkenbaar doordat dan ook vertikaal de - horizontaal altijd geldende - beperking van twee kleuren per 8 pixels geldt, hier dus per 8 beeldlijnen).
Ondanks deze grafische beperkingen op SCREEN 0 is het niet zo heel erg moeilijk om programma's die op dit scherm werken, te verfraaien met een 'repeterende smooth scrolling'. Dit verdient naar ik aanneem wel enige uitleg.

HET BEGRIP SCROLLING

Het begrip scrolling is u misschien al bekend: het 'opschuiven van het hele danwel een gedeelte van het beeld'. Dit kunt u zich voorstellen als een camera die over een decor glijdt en er steeds maar een gedeelte van laat zien. In spellen worden vaak twee manieren toegepast om van beeld te veranderen: het verspringen van het beeld naar een ander als je eruit loopt (zoals bij bv. JET SET WILLY I & II en VAMPIRE KILLER), of een scrolling waarbij het beeld stapsgewijs opschuift naar een ander zonder dat er duidelijke 'kamers' aan te wijzen zijn. Op zo'n soort scrolling doel ik echter ook niet met dit artikel, omdat er nu geen gehele tekening wordt gescrolld, maar een patroon bestaande uit slechts 1 karakter, meerdere keren op het scherm geprint.
Dit zal u aan de hand van de voorbeeldprogrammaatjes wel duidelijk worden. In elk geval staat dit in de bijvoeging 'repeterende'. En voor wie zich nog afvraagt wat er met 'SMOOTH' bedoeld wordt: een SMOOTH scrolling is een scrolling per beeldpunt; de tekening schuift dus steeds 1 pixel (= 1 beeldpunt) op. Ik zal nu eerst uitleggen waarom een horizontal smooth scrolling op een 'MSX-I compatible' grafisch kleurenscherm nauwelijks mogelijk is.

SCREEN 1

De oorzaak hiervoor is op SCREEN 1 wel duidelijk: eigenlijk is dit geen grafisch scherm, maar een tekstscherm. Alleen doordat bij de karakters de vorm en de kleuren ge(her)definieerd kunnen worden, heeft dit scherm (afgezien van de spritemogelijkheden) een grafisch karakter. Een tekening bestaat echter uit karakters. Net als op SCREEN 0 hebben de karakters vaste kolommen waarin ze geplaatst worden, alleen zijn deze op SCREEN 1 niet 6, maar 8 pixels breed. De enige manier om een tekening horizontaal te scrollen is dus door alle karakters 1 kolom naar links of naar rechts te verplaatsen, en dat gaat dus per 8 pixels.
Dit is nu niet bepaald 'SMOOTH'! Dit zien we bijvoorbeeld bij SPY VS. SPY van Aackosoft gebeuren.

SCREEN 2

Op SCREEN 2 liggen de problemen iets anders, maar is het resultaat hetzelfde , nl. als enige horizontale scrollmogelijkheid die per 8 pixels. Op dit scherm wordt het patroon van 8 naast elkaar liggende punten in 1 adres -van BASE(10), het patroongeheugen- in het VRAM (VideoRAM) bewaard. Op 1 adres staat nl. 1 byte van 8 nullen of enen, dus voor 8 beeldpunten welke voor- resp. achtergrondkleur krijgen. Welke de voor- en achtergrondkleuren dan wel zijn staat op een ander adres (in BASE(11) - het kleurengeheugen).
Hier staat per 8 horizontale beeldpunten de waarde VOORGRONDKLEUR + 16 * ACHTERGRONDKLEUR.
Dus per 8 beeldpunten zijn op dit scherm 2 bytes nodig. Voor welke 8 beeldpunten horizontaal twee kleuren benoemd kunnen worden ligt echter vast: dit zijn alle kolommen met een X-coördinaat van 0 t/m 7, van 8 t/m 15, van 16 t/m 23 ... etc. tot en met van 248 t/m 255. Bij een horizontale SMOOTH scrolling zouden deze kolommen steeds 1 beeldpunt op moeten schuiven en dat is niet mogelijk.
Daarom wordt ook op dit scherm een horizontale scrolling altijd per 8 pixels uitgevoerd, zodat na de scrolling patroon- en kleurengeheugen weer samenvallen met de plaatsen op de tekening waar ze voor de scrolling stonden - alleen nu 8 pixels naar links of rechts. Dit zien we bv. bij NEMESIS I & II van Konami.
Vertikaal geldt deze beperking van 2 kleuren per 8 beeldpunten overigens niet; iedere lijn kan een andere kleur krijgen. Een smooth scrolling is vertikaal op SCREEN 2 prima uitvoerbaar en dit gebeurt bv. in de spellen PASTFINDER van Activision en PIPPOLS van Konami.
Niettemin vinden we dit in veel te weinig programma's terug; zeer vaak wordt ook vertikaal een scrolling per 8 beeldpunten (hier beeldlijnen) gebruikt, bv. bij AVENGER van Gremlin en Konami's KNIGHTMARE.

BITMAPPING

Deze scrollingsproblemen zijn met de komst van MSX-II opgelost. Deze machines hebben wat MSX-I moet ontberen: Bitmapping. Bitmapping wil zeggen dat IEDER beeldpunt op een grafisch scherm (SCREEN 4 t/m 8) een andere kleur kan krijgen. Hierdoor valt de beperking van twee kleuren per 8 hor. beeldpunten weg en is ook het gescheiden kleur- en patroongeheugen verleden tijd. Een smooth scrolling is op MSX-II dus naar alle kanten mogelijk zoals blijkt uit bv. de Radarsoftjes zoals de Aardrijkskunde-reeks of Eindeloos.

SMOOTH SCROLLING

Op MSX-I is een horizontale SMOOTH scrolling echter niet geheel uitgesloten. Het kan bv. heel goed als op de te scrollen lijnen slechts twee kleuren voorkomen. Het kleurengeheugen op die regel heeft dan toch overal dezelfde waarde: VOOR GR.KL. + (16*ACHTERGR.KL). Dan kan het schermgeheugen (= patronengeheugen) naar believen veranderd worden. Dit werkt echter alleen op SCREEN 2 en is vanuit BASIC nauwelijks toepasbaar omdat dat veel te langzaam zou gaan.

VANUIT BASIC

Wat wel vanuit BASIC toepasbaar is, is een repeterende smooth scrolling op SCREEN 0. Niet alleen qua snelheid is er een methode om dit zeer goed mogelijk te maken, maar bovendien heb je de voornoemde twee kleuren per regel toch al op de 40-kolomsmode van dit scherm.
Het gaat hier nu natuurlijk niet om een scrolling van een gehele tekening (tekeningen zult u toch niet gauw aantreffen op SCREEN 0) maar om een patroon dat gevormd wordt door meerdere karakters (zoveel u wilt en op het scherm passen; het aantal karakters dat u gebruikt beïnvloedt de snelheid niet merkbaar!).

PRINCIPE

Het principe is eenvoudig maar efficiënt: we zetten een aantal karakters met dezelfde ASCII-waarde op het scherm en veranderen deze vervolgens van vorm en wel zodanig, dat ze tezamen een scrollend patroon vormen. Dit veranderen doen we door rechtstreeks in het VRAM te VPOKEn.
Ik maak in de voorbeelden in dit artikel hiervoor gebruik van CHR$(219), dus met een ASC-waarde van 219. Dit karakter krijgt u vanuit het toetsenbord met GRAPH + P.
Ik heb dit karakter gekozen omdat het in teksten niet zo vaak gebruikt wordt en bovendien op een mooi adres in het karaktersetgeheugen staat (BASE(2)): de vorm staat hier vanaf 3800 decimaal.

Omdat er voor iedere lijn (van 8 pixels breed) van een karakter 1 byte nodig is en een karakter 8 lijnen telt, loopt de vorm van CHR$(219) van 3800 t/m 3807 in het VRAM van SCREEN 0.
Op dit scherm loopt het schermgeheugen , dat de ASCII-waardes van alle karakters (ook spaties) die op het scherm staan, bevat (dit is 24*40 = 960 bytes groot) van 0 t/m 960. Daarna wordt een deel van de VRAM niet gebruikt (maar er is toch meer dan genoeg geheugenruimte voor dit scherm in het VRAM) en start BASE(2) op adres 2048 - type maar eens ? BASE(2) in op SCREEN 0.
In het geheugen van BASE(2) staat voor elk van de 256 karakters de vorm opgeslagen in 8 bytes (voor 8 lijnen).

De totale lengte van dit geheugendeel is 256*8 = 2048 bytes en het loopt dus van 2048 t/m 4096 in het VRAM. Als u een ander karakter voor de scrolling wenst te gebruiken, kunt u dit veranderen door S in regel 10 een andere waarde te geven, nl. 2048 + (ASCII-WAARDE v/h KARAKTER * 8).
Als u S verandert en u gebruikt de TEST-DEMO, dient u ook de waarde 219 in regel 20 aan te passen; dit moet de ASCII-waarde van het te gebruiken karakter worden. Nu volgt eerst het TEST-DEMO programma(atje). Hiervanuit wordt de subroutine aangeroepen die voor de scrolling zorgt.

Ik zal later een paar voorbeeldroutines geven. Wanneer u de scrollroutines niet direkt in een eigen programma opneemt kunt u ze hiermee mooi uitproberen.

TESTDEMO D2

```
1 '*********************
2 '****** TEST/DEMO ****
3 '*********************
4 '* Demo rep. smooth scrolling screen
0
5 '* voor het aanroepen van de gepubli
-
6 '* ceerde routines waarvan de tite
l
7 '* met #A begint.
8 '
10 KEYOFF:SCREEN0:WIDTH40:COLOR14,1:S=
3800:A=50:'A=Snelheid scrolling
15 ' Vult het scherm met chr$(219).Als
 U
16 ' een ander karakter gebruikt dient
 U
17 ' behalve S in regel 10 ook de waar
de
18 ' 219 in regel 20 aan te passen.
20 FORI=0TO24:LOCATE0,I:PRINTSTRING$(4
0,219);:NEXT
30 FORI=0TO4:READA$:LOCATE20-LEN(A$)/2
,5+I*3:PRINTA$:NEXT:LOCATE5,2:PRINT"Ge
bruik cursor >< voor snelheid":'PRINT
TEKST
40 FORI=1TOA:IFSTICK(0)=0THENNEXT:GOSU
B1000:GOTO40ELSEI=A:NEXT:GOTO50
50 IFSTICK(0)=3THENA=A+1:LOCATE18,21:P
RINTA:GOTO40ELSEIFSTICK(0)=7ANDA>1THEN
A=A-1:LOCATE18,21:PRINTA:GOTO40:NEXTEL
SE40
500 'Tussen de woorden in i.p.v. spa-
510 'ties CHR$(219) typen (=GRAPH + P)
530 DATADEMO SCREEN0,REPETERENDE,SMOOT
H,SCROLLING,door*MARTIJN*HONDEMA
```

TEST/DEMO

Om de scrollsnelheid naar uw wensen aan te kunnen passen, kunt u het interval, waarop de routine steeds zal worden aangeroepen, aanpassen met de cursor links/rechts.
De lengte van de FOR-NEXT wachtlus, die in regel 40 voor de vertraging zorgt, komt onderaan het scherm te staan wanneer u de snelheid verandert; dit is de variabele A.
De beginwaarde van A is vastgelegd in regel 10.
De aanroep van de routines vindt plaats in regel 40 waar 'GOSUB 1000' staat; dit is dus het regelnummer waarop ik de voorbeeldroutines laat beginnen. U kunt daar ten alle tijde een ander regelnummer van maken. Als u bediening met de joystick prefereert dient u van het 'STICK(0)' in regel 40 en 50 'STICK(1)' of 'STICK(2)' te maken, afhankelijk van de gekozen joystickpoort. De gegeven routines kunnen natuurlijk niet zelfstandig werken; als u de TEST/DEMO niet gebruikt moet u de routine vanuit eigen programmatuur aanroepen, en wanneer u de routines tenslotte in eigen programma's gebruikt, heeft de TEST/DEMO uiteraard ook geen functie meer.
In dat geval is het belangrijk ervoor te zorgen dat de scrollroutine met zo gelijkmatige intervallen wordt aangeroepen, zodat de scrolling zo soepel mogelijk zal lopen. Ik zal nu eerst een voorbeeld van een scrollroutine geven:

#A 001 D2

```
1000 'Routine #A.001
1010 F=255:G=G+1:IF G>7 THEN G=0
1020 IF G<7 THEN VPOKE S+G+1,F:VPOKE S
+G,0:RETURN ELSE VPOKE S,F:VPOKE S+7,0
:RETURN
```

LISTING #A.001

Het is eigenlijk een knap staaltje van de VDP (de Video Display Processor, de chip die de beeldopbouw in uw computer verzorgt), dat deze iedere verandering aan de vorm (of kleur op SCREEN 1) van een karakter onmiddellijk op het scherm toont als dit karakter tenminste op het scherm staat. Zonder die eigenschap van de VDP zou deze scrolltechniek onmogelijk zo eenvoudig te verwezenlijken zijn.
Ik zal nu eerst de werking van deze routine uit de doeken doen, hoewel de listing al het een en ander verraadt.
Het eerste, dat u moet weten en ook inmiddels al weet, is dat een karakter uit beeldpunten in een raster van 8*8 pixels bestaat. De vorm van een karakter, m.a.w. welke punten in dat raster 'aan' danwel 'uit' staan, en dus voor- resp. achtergrondkleur krijgen, staat zoals u weet in het patronengeheugen (BASE(2)). Een karakter is 8 lijnen van 8 pixels en dus 8 bytes groot.
Zoals u gezien hebt, bewerkstelligt dit routinetje een vertikale scrolling van lijnen van boven naar beneden. Eerst wordt in regel 20 van de TEST/DEMO het scherm volgeprint met CHR$(219). U kunt hier een ander karakter van maken zoals aangegeven is boven dit programma. Vervolgens wordt het karakter steeds van vorm veranderd en wel zodanig, dat het steeds uit 1 horizontale lijn bestaat die op lijn 0 bovenaan op het karakter begint en dan bij iedere aanroep 1 lijn omlaag schuift. Wanneer de onderste lijn bereikt is, gaat de lijn weer helemaal bovenaan op lijn 0 van het karakter verder om weer opnieuw omlaag te schuiven.

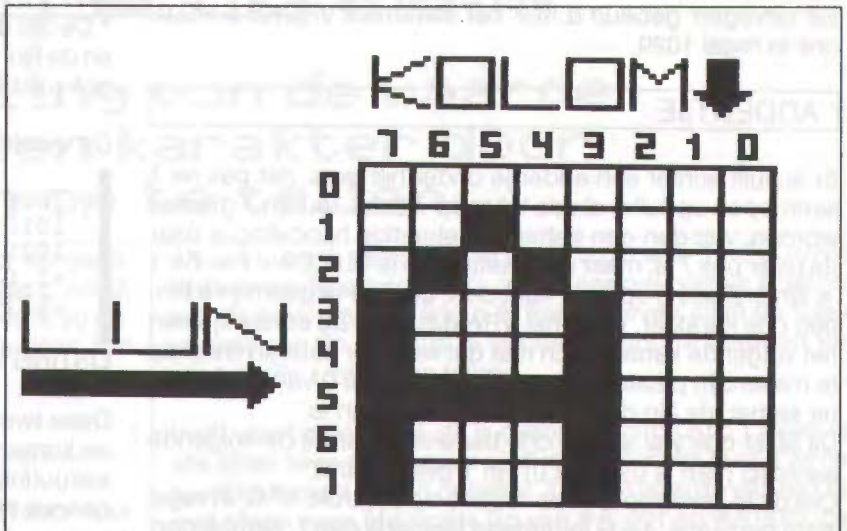


Fig. 1. De opbouw van een karakter. Op screen 0 worden kolom 0 en 1 niet afgebeeld.

DOORSCHUIVEN

De lijnen van het karakter die achtereenvolgens geset worden (= een patroon krijgen, hier een door lopende lijn) zijn dus 0 » 1 » 2 » 3 » 4 » 5 » 6 » 7 » 0 » 1 » 2 etc.
Deze sequentie (opeenvolging) wordt bijgehouden in de teller G in regel 1010. In feite gaat de lijn dus over het karakter van boven naar beneden en start dan weer bovenaan, maar het lijkt net of hij omlaag doorloopt in het karakter daaronder omdat de onderkant van een karakter aansluit aan de bovenkant van het karakter daaronder.

VEGEN

Het is echter niet voldoende alleen lijnen van het karakter te setten (= patroon geven); wanneer dat alleen maar zou gebeuren krijgen alle lijnen van het karakter het patroon en bestaat het karakter na 8 aanroepen op alle lijnen uit het patroon waarna er geen verandering meer aan te zien is.
Om dat te voorkomen, wordt niet de lijn die de teller G aangeeft geset, maar de lijn die 1 lager op het beeldscherm ligt, en dus met het nummer 1 hoger dan de teller (bv. lijn 4 van het karakter ligt op uw beeldscherm 1 lijn lager dan lijn 3, en lijn 2 1 hoger dan lijn 3). Het setten van deze lijn gebeurt met 'VPOKE S+G+1,F' in regel 1020 van de routine. S is het startadres van het karakter in het VRAM (=208 +ASC(KARAKTER)*8), G is de teller voor het te setten lijnnummer en F het patroon dat de lijn krijgt; dat leg ik later uit.
Hiermee schieten we nog niets op, dat komt pas als we ook de lijn die 1 lager ligt dan de gesette lijn (dus 1 hoger op uw beeldscherm, dit is de lijn die de teller aangeeft) gaan resetten (= leeg maken). Hierdoor schuift de lijn op zonder een spoor achter te laten, of preciezer gezegd, de lijn laat wel een spoor achter maar veegt het direkt weer uit.
U kunt zich dit voorstellen als iemand die door het zand loopt met een bezem achter zich aan om z'n spoor uit te vegen. Dit 'uitvegen' gebeurt d.m.v. het statement VPOKE S+G,0, ook in regel 1020.

ADDERTJE

Er schuilt echter een addertje onder het gras, dat pas na 7 aanroepen opduikt: als de teller op 7 staat zou lijn 7 gereset worden, wat dan een geheel regelmatige handeling is daar de teller ook 7 is, maar de te setten lijn is TELLER+1 = 7+1 = lijn 8. Zoals u op fig. 1 kunt zien, bestaat er geen lijn 8 binnen ons karakter, en in het VRAM is lijn 8 de eerste lijn van het volgende karakter. En met dat karakter hebben we niets te maken. In plaats hiervan moeten we lijn 0 van het karakter setten; de lijn dus, die dan aan de beurt is.
Dit blijkt ook wel als we ons bedenken dat bij de volgende aanroep (dan is de teller 0) lijn 1 geset wordt.
Om dit te bereiken is in de routine de controle 'IF G' in regel 1020 geplaatst. Als G inderdaad kleiner is dan 7, kan de 'normale' handeling (set teller+1, reset teller) plaatsvinden. Is G echter niet kleiner dan 7 (dan is G altijd 7; zie de teller in 1010) dan moet de alternatieve handeling plaatsvinden: VPOKE S,F voor het setten van lijn 0 en VPOKE S+7,0 voor het resetten van lijn 7.

BINAIR

Dan nu nog wat de variabele F betreft, gedefinieerd in regel 1010: dit is de waarde waarmee de lijn geset wordt, en is dus het patroon wat de scrollende lijn gaat krijgen. Het is duidelijker deze waarde binair te schrijven; dan vormt het patroon van nullen en enen het patroon van de lijn.
Zo is de waarde die voor F in de listing staat, 255, te herschrijven als &b11111111; hieruit blijkt dat het een aaneengesloten lijn wordt. Om een gestippeld patroon te krijgen, kunt u bijvoorbeeld in regel 1010 F=&B10101010 zetten, of F=&B11011000 voor een grover patroon.

ZES PIXELS BREED

Hierbij dient u zich wel te bedenken, dat we op SCREEN 0 werken, waar de karakters niet 8 pixels breed zijn zoals op SCREEN 1, maar 'slechts' 6 pixels. Dit betekent, dat de laatste 2 bits (de laatste 2 nullen of enen) die u F geeft, geen zichtbare invloed hebben op de vorm van het karakter (dit zijn lijn 0 en 1; zie fig. 1) Hoewel ze dus geen invloed hebben dient u ze wel te benoemen, want &B101010 is bv heel wat anders als &B10101010 ook al zie je die laatste '10' niet terug in het karakter. U zult begrijpen op SCREEN 1 alle 8 bits invloed hebben op de vorm van de lijn van 1 karakter. Ik zal het echter voorlopig eerst over SCREEN 0 hebben.

OPBOUW

Wanneer u de opbouw van dit routinetje begrijpt, zal het u duidelijk zijn dat een scrolling omhoog (van onder naar boven) ook niet erg moeilijk te maken is. Slechts 2 aanpassingen zijn nodig, namelijk:

* De teller moet andersom werken (volgorde nu lijn 7 » 6 » 5 » 4 » 3 » 2 » 1 » 0 » 7 » 6 » 5 etc.)
* De lijn die de teller aangeeft wordt nu geset en de lijn 1 hoger (1 lager op het beeldscherm) gereset; wat ook juist andersom is.

Dit wordt dan de routine:

```
1000 'Routine #A.002
1010 F=255:G=G-1:IF G<0 THEN G=7
1020 IF G<7 THEN VPOKE S+G,F:VPOKE S+G
+1,0:RETURN ELSE VPOKE S,0:VPOKE S+7,F
:RETURN
```

LISTING #A.002

#A 002 D2

Deze twee routines (#A.001 en #A.002) zijn de basis voor de komende voorbeelden in dit artikel. Nu we deze twee basisroutines voor een liniaire vertikale scrolling hebben, kunnen we hiermee tal van grapjes en variaties uithalen.

DIAGONAAL

Ik zal eerst een scrolling in de diagonale windrichtingen laten zien. Ik zal met de eenvoudigste beginnen, hoewel de andere niet veel moeilijker zijn. Dit is een scrolling in ZUIDWESTELIJKE richting.

ZUIDWESTELIJK

Als we alleen kijken naar het feit dat er omlaag gescrolld wordt (dus naar het zulden), kunnen we gewoon de listing #A.001 in stand houden, en grotendeels doen we dat ook, vergelijk regel 1010 en 1020 van bv. #A.001 en #A.003 maar eens. We moeten nu echter ook in westelijke richting scrollen en daarvoor is een aanpassing nodig, die u in regel 1015 aantreft. U ziet dat hier de waarde die F moet hebben wordt berekend; deze kunt u dus niet meer zelf definiëren.
Eerst beschouwen we de berekening F=2^(G+2). Zoals u uit het voorgaande opgemaakt zult hebben, bepaalt de waarde van F het (horizontale) patroon van de lijn.
Waar we in de vorige routines een doorlopende lijn gebruikten, moeten we nu een ander patroon gebruiken. Bij een doorlopende lijn zie je er immers niets van, dat er ook horizontaal (hier naar het westen) gescrolld wordt.
Daarom zet ik nu losse punten op het scherm, die zowel omlaag als naar links bewegen. We moeten nu het karakter zodanig definiëren, dat deze op de lijn die de teller aangeeft, bestaat uit 1 punt dat op een kolom ligt, die verband houdt met de stand van de teller. Welk verband dat is, leg ik zo dadelijk uit, eerst iets over de waarden die F kan hebben, om de te setten lijn uit 1 punt te laten bestaan.

PROBEREN

U kunt deze waarden proefondervindelijk bepalen door voor alle 8 horizontale posities, die het punt kan hebben, de binaire waarde op het scherm te zetten.
Vanuit BASIC kan dit met ?&B00000001, vervolgens ?&B00000010, dan ?&B00000100 en zo door tot de een de eerste positie heeft 'bereikt'. U zult dan merken, dat deze decimale waarde steeds een macht van 2 is. Voor de een op de eerste positie (kolom 0) is F 2^0 (=1), voor op de tweede positie (kolom 1) 2^1 (=2), op de derde positie (kolom 2) 2^2 (=4) en zo door tot F tenslotte op de achtste positie (kolom 7) 2^7 is, wat gelijk is aan 128 decimaal.
Het zal u opgevallen zijn, dat ik 'posities' anders tel dan 'kolommen', maar dit nuanceverschil is van weinig belang. Omdat er 8 verschillende posities zijn, tel ik deze van 1 t/m 8 waarbij ik de 0 verwaarloos. 0 heeft voor de computer wel waarde, zodat ik kolommen van 0 t/m 7 tel, wat wetenschappelijk gezien juister is.

WETENSCHAPPELIJK

Deze waarde, die F moet krijgen om niet meer dan 1 bepaalde kolom te setten, is ook iets wetenschappelijker te berekenen, als we weten hoe we een binair getal omzetten naar decimaal. Dit is niet zo moeilijk: we bekijken het binaire getal bit voor bit, dus positie voor positie, van kolom 0 t/m kolom 7. Als het bit dat we bekijken 0 is, telt dit niets bij het totaal op.

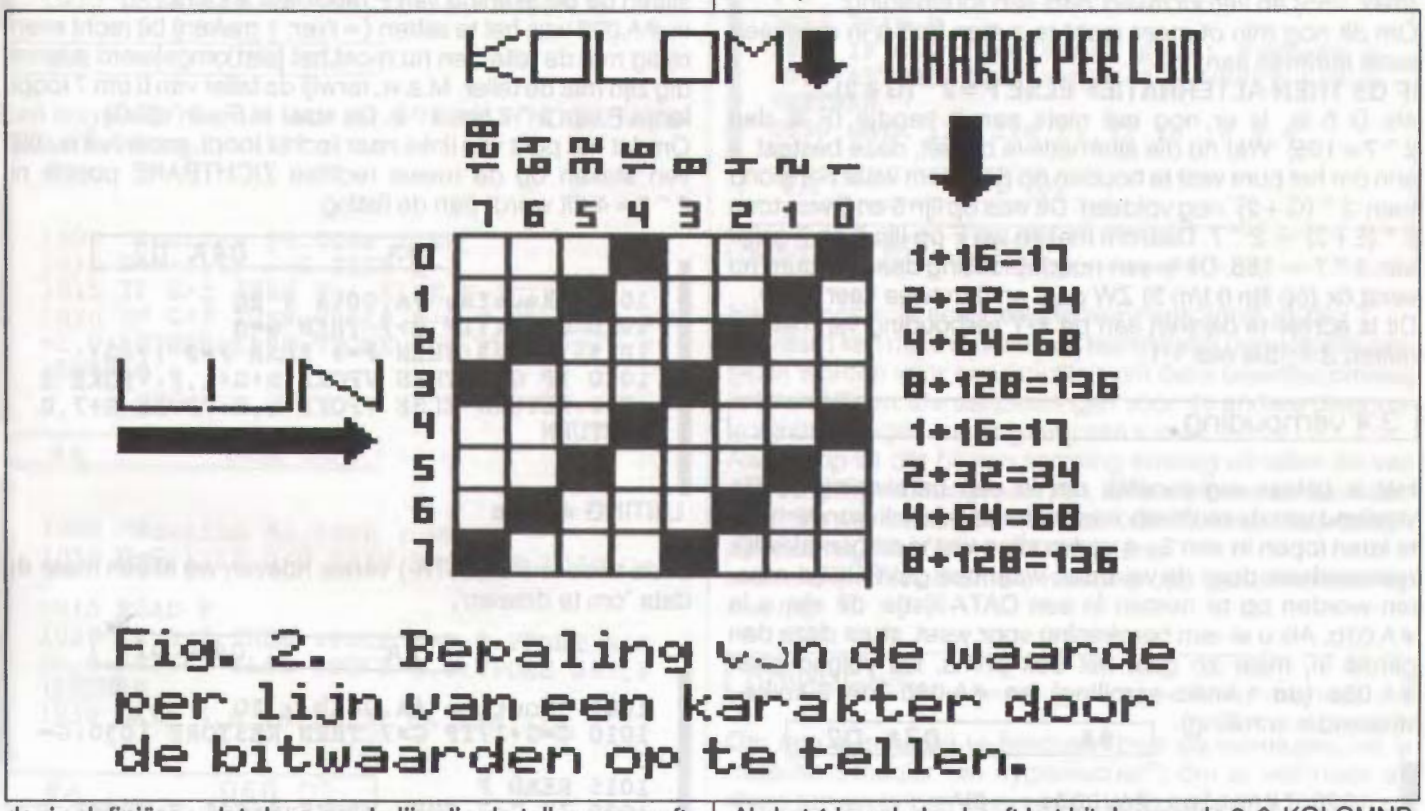


Fig. 2. Bepaling van de waarde per lijn van een karakter door de bitwaarden op te tellen.

Als het 1 is wel: dan heeft het bit de waarde 2 ^ KOLOMNR. Welke waarde iedere positie dan vertegenwoordigt staat boven de kolommen in figuur 2. Rechts ervan staat de totale waarde per lijn, die dus te vinden is door de waarden van alle bits die 1 zijn op te tellen.

In dit voorbeeld (fig. 2, dit is overigens chr$(203)) staan op alle lijnen twee enen, maar als het er meer zijn kunt u ze natuurlijk allemaal blijven optellen. Zo is de waarde van een lijn met alleen enen (de dichte lijn van #A. 001 en #A.002), gelijk aan 128 + 64 + 32 + 8 + 4 + 2 + 1 = 255.
En u ziet, dat F hier ook als 255 gedefinieerd wordt.
Bij de diagonale scrolling wordt er ,zoals al gezegd, geen dichte lijn, maar worden er losse punten gescrolld.
Dit betekent, dat F steeds een macht van twee moet zijn omdat er steeds maar 1 punt per regel benoemd wordt.

We zouden F nu gewoon de waarde 2 ^ KOLOMNR. kunnen geven, en dan krijgen we bij een karakter van 8*8 pixels inderdaad een keurige diagonale scrolling.
Ons karakter is echter op SCREEN 0 6*8 pixels groot omdat kolom 0 en kolom 1 wegvallen. Als we nu gewoon de berekening 2 ^ KOLOMNR. aan zouden houden, worden op lijn 0 en lijn 1 deze twee kolommen wel benoemd waarvan we dan niets zien op het scherm.
Vandaar dus dat we niet 2 ^ KOLOMNR. in F zetten, maar 2 ^ (KOLOMNR. + 2). We schuiven dus de scrollende punten twee posities naar rechts, zodat op lijn 0 en 1 resp. kolom 2 en kolom 3 worden geset, die wel zichtbaar zijn. We zetten in regel 1015 dus F = 2 ^ (G + 2) i.p.v. 2 ^ G en klaar is Kees.

Of toch niet.....

Nu komen de problemen namelijk op lijn 6 en 7. Hier is F 2 ^ 8 resp. 2 ^ 9 en deze getallen zijn te groot voor een byte (max. 255) en veroorzaken zelfs een foutmelding.
Om dit nog min of meer recht te zetten treft u in regel een extra kontrole aan :
IF G5 THEN ALTERNATIEF ELSE F = 2 ^ (G + 2).
Als G 5 is, is er nog net niets aan 't handje (F is dan 2 ^ 7 = 128). Wat nu die alternatieve betreft, deze bestaat erin om het punt vast te houden op de kolom waar het stond toen '2 ^ (G + 2)' nog voldeed. Dit was op lijn 5 en F was toen 2 ^ (5 + 2) = 2 ^ 7. Daarom maken we F op lijn 6 en 7 gelijk aan 2 ^ 7 = 128. Dit is een noodoplossing daar het punt nu eerst 6x (op lijn 0 t/m 5) ZW gaat en dan twee keer ZUID.
Dit is echter te danken aan de X-Y verhouding van het karakter: 3:4, dus niet 1:1.

3:4 verhouding

Het is helaas erg moeilijk om in een berekening de lijn vloeiend van de rechterbovenhoek naar de linkeronderhoek te laten lopen in een 3 : 4 verhouding wel is dit gemakkelijk realiseerbaar door de waarden waarmee geVPOKEd moeten worden op te nemen in een DATA-lijstje; dit ziet u in #A.03b. Als u er een berekening voor weet, stuur deze dan gerust in, maar zo gaat het ook prima. Nu volgen eerst #A.03a (de 1-kniks-scrolling) en #A.03b (de 2-kniks-, vloeiender scrolling).

#A 03A D2

```
1000 'Routine #A.003a : ZW
1010 G=G+1:IF G>7 THEN G=0
1015 IF G>5 THEN F=128 ELSE F=2^(G+2)
1020 IF G<7 THEN VPOKE S+G+1,F:VPOKE S
+G,0:RETURN ELSE VPOKE S,F:VPOKE S+7,0
:RETURN
```

#A 03B D2

```
1000 'Routine #A.003b : ZW
1010 G=G+1:IF G>7 THEN RESTORE 1030:G=
0
1015 READ F
1020 IF G<7 THEN VPOKE S+G+1,F:VPOKE S
+G,0:RETURN ELSE VPOKE S,F:VPOKE S+7,0
:RETURN
1030 DATA 4,8,16,16,32,64,128,128
```

Deze laatste listing (de 'DATAVERSIE') geeft nog steeds geen perfect vloeien de scrolling, maar wanneer de routine vaak plaatsvindt (met een klein interval tussen de aanroepen) is het resultaat toch niet onaardig. uit de data blijkt wel dat er hier twee knikken in zitten, nl. op kolom 4 (2 ^ 4 = 16) en kolom 7 (2 ^ 7 = 128).
De positie van de knikken (waar het punt recht omlaag - ZUID- gaat) kunt u natuurlijk veranderen. Houdt er wel rekening mee dat de waarden in de data voor 1 scrollend punt een macht van 2 tussen 2 ^ 2 en 2 ^ 7 zijn.
Geen macht van twee levert twee of meer punten op de te setten lijn op en 2 ^ 0 en 2 ^ 1 setten kolom 0 of 1. Waarden van 2 ^ 8 of groter zijn natuurlijk ook uit den boze.

ZUIDOOSTELIJK

Een scrolling in ZUIDOOSTELIJKE richting (#A.004) is gemakkelijk af te leiden uit de in ZUIDWESTELIJKE richting (#A.003). Van de regels 1010 en 1020 kunnen we afblijven; alleen de berekening van F moet iets anders.
In #A.003 was het te setten (= hier: 1 maken) bit recht evenredig met de teller; en nu moet het juist omgekeerd evenredig zijn met de teller. M.a.w. terwijl de teller van 0 t/m 7 loopt, loopt F van 2 ^ 7 t/m 2 ^ 2. Dit staat in F = 2 ^ (7-G).
Omdat het punt van links naar rechts loopt, moet het nu blijven steken op de meest rechtse ZICHTBARE positie nl. 2 ^ 2 = 4 dit wordt dan de listing :

#A 04A D2

```
1000 'Routine #A.004a : ZO
1010 G=G+1:IF G>7 THEN G=0
1015 IF G>5 THEN F=4 ELSE F=2^(7-G)
1020 IF G<7 THEN VPOKE S+G+1,F:VPOKE S
+G,0:RETURN ELSE VPOKE S,F:VPOKE S+7,0
:RETURN
```

LISTING #A.04a

In de twee-kniks (DATA-) versie hoeven we alleen maar de data 'om te draaien'.

#A 04B D2

```
1000 'Routine #A.004b : ZO
1010 G=G+1:IF G>7 THEN RESTORE 1030:G=
0
1015 READ F
1020 IF G<7 THEN VPOKE S+G+1,F:VPOKE S
+G,0:RETURN ELSE VPOKE S,F:VPOKE S+7,0
:RETURN
1030 DATA 128,128,64,32,16,16,8,4
```

LISTING #A.04b

De routines #A.003 en #A.004 zijn nu niet moeilijk meer om te zetten voor een diagonale scrolling omhoog, dus naar NOORDWESTEN en NOORDOOSTEN.
De berekeningen (regel 1015) blijven hetzelfde alleen maar alleen nu met de routine #A.002 voor regel 1010 en 1020.

```
1000 'Routine #A.005a : NO
1010 G=G-1:IF G<0 THEN G=7
1015 IF G>5 THEN F=128 ELSE F=2^(G+2)
1020 IF G<7 THEN VPOKE S+G,F:VPOKE S+G
+1,0:RETURN ELSE VPOKE S,0:VPOKE S+7,F
:RETURN
```

```
#A          05A D2
```

En de vloeiender dataversie (data van #A.003b weer 'omgedraaid'):

```
1000 'Routine #A.005b : NO
1010 G=G-1:IF G<0 THEN RESTORE 1030:G=
7
1015 READ F
1020 IF G<7 THEN VPOKE S+G,F:VPOKE S+G
+1,0:RETURN ELSE VPOKE S,0:VPOKE S+7,F
:RETURN
1030 DATA 128,128,64,32,16,16,8,4
```

```
#A          05B D2
```

Dan nog beide routines voor een scrolling in NOORDWESTELIJKE richting:

```
1000 'Routine #A.006a : NW
1010 G=G-1:IF G<0 THEN G=7
1015 IF G>5 THEN F=4 ELSE F=2^(7-G)
1020 IF G<7 THEN VPOKE S+G,F:VPOKE S+G
+1,0:RETURN ELSE VPOKE S,0:VPOKE S+7,F
:RETURN
```

```
#A          06A D2
```

```
1000 'Routine #A.006b : NW
1010 G=G-1:IF G<0 THEN RESTORE 1030:G=
7
1015 READ F
1020 IF G<7 THEN VPOKE S+G,F:VPOKE S+G
+1,0:RETURN ELSE VPOKE S,0:VPOKE S+7,F
:RETURN
1030 DATA 4,8,16,16,32,64,128,128
```

```
#A          06B D2
```

LEUK EFFECT

Een erg leuk effect geeft het, als we de teller voor een scrolling omhoog (G steeds van 7 t/m 0) combineren met de VPOKEs (=regel 1020) voor een scrolling omlaag.
Dit heeft tot gevolg dat het 'uitwissen van het spoor' (VPOKE...,0) steeds 1 lijn vooroploopt op de te setten lijn, of zo u wilt, 7 lijnen achter loopt. U kunt zich dit voorstellen als u denkt aan een ander persoon met een bezem, die er een eigen veegwijze op na houdt. Hij loopt rond in een kringetje van 8 stappen en veegt steeds 1 stap voor zich uit het spoor weg, dat dan al 7 stappen oud is.
Er is op het hele kringetje steeds precies 1 stap schoon en de rest van het kringetje vertoont steeds het spoor.

Dit kringetje is de teller in de routine, die ook steeds rondgaat. Kijkt u maar eens naar alle worpjes die u zo op het scherm tovert:

```
1000 'Routine #A.007a : NO
1010 G=G-1:IF G<0 THEN G=7
1015 IF G>5 THEN F=128 ELSE F=2^(G+2)
1020 IF G<7 THEN VPOKE S+G+1,F:VPOKE S
+G,0:RETURN ELSE VPOKE S,F:VPOKE S+7,0
:RETURN
```

```
#A          07A D2
```

Ook dit is heel best in de tweekniks versie mogelijk; of u dit mooier vindt is een kwestie van smaak:

```
1000 'Routine #A.007b : NO
1010 G=G-1:IF G<0 THEN RESTORE 1030:G=
7
1015 READ F
1020 IF G<7 THEN VPOKE S+G+1,F:VPOKE S
+G,0:RETURN ELSE VPOKE S,F:VPOKE S+7,0
:RETURN
1030 DATA 128,128,64,32,16,16,8,4
```

```
#A          07B D2
```

WORMENKOLONIE

Heeft u ooit zo'n eensgezinde wormenkolonie gezien ?
Uiteraard kan deze routine ook betrekkelijk gemakkelijk omgezet worden voor een scrolling om deze beestjes omlaag te laten glijden. De aanpassingen voor de andere diagonale windrichtingen laat ik graag aan u over.
Als u erop let dat bij een scrolling omlaag de teller die van een scrolling omhoog moet zijn, en andersom, zal dat waarschijnlijk wel lukken. Bekijkt u ook de verwantschap tussen de routines #A.003 t/m #A. 006 eens.
Mocht u er niet uitkomen, neemt u dan gerust kontakt op met mij.

KRINGETJE

Om nog even voort te borduren over de wormpjes; het is misschien mooier (en hygiënischer?) om er wat meer afstand tussen te brengen, zodat de afzonderlijke lijntjes ook als zodanig herkenbaar blijven.
Hoe we dit doen, zal duidelijk zijn als we de vegende persoon er nog even bijhalen (U weet wel, die figuur die voor zich uit veegde).
Niet geheel tevreden over het bereikte resultaat met de bezem besloot hij nu twee stappen voor zich uit te gaan vegen, en tot zijn/haar grote voldoening kwam er nu een twee keer zo groot 'schoon' gedeelte in het kringetje.
We kunnen dit effect bereiken door niet alleen de lijn die de teller aangeeft te vegen (resetten), maar ook de lijn 1 lager dan de teller (verlenging van het veegbereik), terwijl nog steeds de lijn 1 hoger dan de teller geset wordt (hier loopt nl. onze veegheld). Ik geef het voorbeeld voor een scrolling omhoog van de worpjes met 2 lege lijnen ertussen; bij een scrolling omlaag moet de bezem aan de andere kant van de te setten lijn staan.

```
1000 'Routine #A.007c : NO
1010 G=G-1:IF G<0 THEN G=7
1015 IF G>5 THEN F=128 ELSE F=2^(G+2)
1020 GOSUB 1025:IF G<7 THEN VPOKE S+G+
1,F:VPOKE S+G,0:VPOKE S+G-1,0:RETURN E
LSE VPOKE S,F:VPOKE S+7,0:VPOKE S+6,0:
RETURN
1025 IF G<>0 THEN RETURN ELSE VPOKE S+
7,0:RETURN
```

```
#A          07C D2
```

De 'DATAVERSIE':

```
1000 'Routine #A.007d : NO
1010 G=G-1:IF G<0 THEN RESTORE 1030:G=
7
1015 READ F
1020 GOSUB 1025:IF G<7 THEN VPOKE S+G+
1,F:VPOKE S+G,0:VPOKE S+G-1,0:RETURN E
LSE VPOKE S,F:VPOKE S+7,0:VPOKE S+6,0:
RETURN
1025 IF G<>0 THEN RETURN ELSE VPOKE S+
7,0:RETURN
1030 DATA 128,128,64,32,16,16,8,4
```

```
#A          07D D2
```

TWEE KEER VEGEN

U ziet, dat zowel bij de normale als bij de alternatieve handeling de bezem twee keer z'n werk doet (dit zijn de VPOKE'S ...,0). Er vindt nu nog een extra controle plaats in de routine. Deze heb ik voor de overzichtelijkheid in een extra routine binnen de totale routine gezet. Er is hier dus sprake van een 'genestelde GOSUB'. Zonder deze controle in regel 1025, zouden de wormen gaan krimpen als de teller nul is - doordat er dan een lijn buiten het karakter gereset zou worden - en daarna weer rekken.
Dit is wel een natuurlijk gezicht als het om wormen gaat en u kunt de kontrole naar believen weglaten (de 'GOSUB 1025' op regel 1020 verwijderen), maar ik zal u de gecorrigeerde versie niet onthouden (zie boven).
De routine op 1025 wordt steeds weer aangeroepen maar heeft als G niet 0 is geen functie. Een controle in 1020 of de 2e alternatieve handeling (in 1025) wel nodig is maakt deze regel overbodig ingewikkeld; deze kontrole in 1025 zelf is overzichtelijker. Waarom is die extra controle nu nodig ?

PROBLEEM OPGELOST

Omdat de bezem twee lijnen leegveegt, worden de lijn 'TELLER' en de lijn 'TELLER-1' gereset. Als de teller 0 is, is het vegen van lijn 'TELLER' geen probleem, maar lijn 'TELLER-1' zou lijn -1 worden, de bezem steekt dus uit het karakter. In plaats van lijn -1 wordt lijn 7 geset (VPOKE S+7,0) en voila, het probleem is opgelost.

EXTRA LIJN

De oorspronkelijke kontrole IF G blijft natuurlijk gewoon staan, alleen wordt een extra lijn gereset. Bij de alternatieve handeling die plaatsvindt als G wel 7 is, zijn de te resetten lijnen gewoon TELLER en TELLER-1 (S+7 en S+6), alleen wordt i.p.v. TELLER+1 (de niet-bestaande lijn 8) lijn 0 geset (VPOKE S,F).

BESLUIT

Ter besluit van deze aflevering geef ik nog enkele suggesties wat u met een scrolling kunt doen in de vorm van een paar variaties.

TWEE TELLERS

De eerste geeft een vertikale scrolling van niet 1 maar 2 lijnen. Welke lijn en geset moeten worden wordt nu in TWEE tellers bijgehouden: G en H in regel 1010 resp. 1015.
Voor beide lijnen staat het patroon in een aparte constante: E en F, gedefinieerd in 1010. U kunt deze ook in de TEST-DEMO of in Uw eigen programma benoemen zodat ze niet steeds weer herbenoemd worden. Het verschil in de beginstand van de tellers moet U in het begin van de TEST/DEMO (bv. regel 10) of het aansturingsprogramma zetten want dit mag maar 1 keer benoemd worden. Met G=0:H=4 krijgt u mooi tussen alle lijnen dezelfde afstand (nl. 3 lege lijnen) daar het verschil tussen de teller daarna behouden zal blijven. Verder kunt U G=0 weglaten want dat is G (en iedere andere onbenoemde variabele) altijd als u het (een) programma start. Het VPOKEn vindt als gebruikelijk plaats in regel 1020 en 1030, nu dus voor beide tellers afzonderlijk. De RETURNs staan nu uiteraard in 1030.

10 KEY OFF: SCREEN 0:WIDTH 40:COLOR 14,1:H=4:
S=3800:A=50:'A=SNELHEID SCROLLING

```
1000 'Routine #A.008
1010 E=255:F=170:G=G+1:IF G>7 THEN G=0
1015 H=H+1:IF H>7 THEN H=0
1020 IF G<7 THEN VPOKE S+G+1,E:VPOKE S
+G,0 ELSE VPOKE S,E:VPOKE S+7,0
1030 IF H<7 THEN VPOKE S+H+1,F:VPOKE S
+H,0:RETURN ELSE VPOKE S,F:VPOKE S+7,0
:RETURN
```

```
#A          008 D2
```

TWEEDE VARIATIE

De tweede variatie (#A.009) is een combinatie van een vertikale scrolling omhoog en een vertikale scrolling omlaag. Dit is ook wel mogelijk met twee afzonderlijke tellers zoals in de vorige routine.
Het kan echter korter en efficiënter: omdat de tellers altijd tegengesteld moeten zijn (als de eerste teller 7 is, moet de tweede teller 0 zijn, zo ook 6-1, 5-2, 4-3, 3-4, 2-5, 1-6, 0-7) kan de tweede teller gemakkelijk afgeleid worden uit de eerste.

Reken de formule 'TWEEDE TELLER = (7-EERSTE TELLER)' maar na en konkludeer dat dit voor alle waarden van 0 t/m 7 klopt.
We herschrijven de tweede teller dus als (7-EERSTE TELLER). Ik hou nu dus gewoon 1 teller aan zoals U ziet in regel 1010. Alleen naast de VPOKEs uit #A.001 voor de scrolling omlaag , heb ik nu ook de VPOKEs uit #A.002 in regel 1020 toegevoegd, met als enige verschil dat ik als teller nu niet G heb genomen, maar (7-G).
Daardoor wordt deze laatste scrolling omgetoverd in een scrolling omhoog met de teller van een scrolling omlaag. Try it out en aanschouw het resultaat.

```
1000 'Routine #A.009
1010 E=255:F=170:G=G+1:IF G>7 THEN G=0
1030 IF G<7 THEN VPOKE S+(7-G),F:VPOKE
 S+G+1,E:VPOKE S+G,0:VPOKE S+(7-G)+1,0
:RETURN ELSE VPOKE S+7,E:VPOKE S,F:VPO
KE S,0:VPOKE S+7,0:RETURN
```

#A 009 D2

BEREKENING VAN F

Als sluitstukje voor deze aflevering geef ik geen gehele routine, maar slechts de berekening van F.
Deze berekening kunt U op regel 1015 van #A.004a of een soortgelijke routine in plaats van de oude 1015 zetten.

1015 F=128-(2^G)/2 of 1015 F=128-(2^G)

WIE KAN DIT ?

Tenslotte hoop ik nog een beroep te kunnen doen op Uw programmeertalent. Het is mij nog niet gelukt om de routine van de glijdende wormen zodanig te schrijven, dat het aantal spaties tussen de lijntjes bepaald kan worden door een in het stuurprogramma gedefinieerde variabele.
Het probleem is dat bij een groot aantal lege lijnen tussen de sliertjes de 'bezem' ook heel breed moet worden, en dus binnen de kortste keren buiten het karakter gaat steken.
Dit is te verhelpen door waar de bezem uitsteekt en dus lijnen buiten het karakter geVPOKEd (gereset) dreigen te worden, deze lijnen in plaats daarvan aan de andere kant (de kant waar ze niet uitsteken) te gaan resetten. We moeten de bezem waar hij uitsteekt dus afsnijden en aan de andere kant v/h karakter weer naar binnen steken. Het gaat hier natuurlijk altijd om de boven- of de onderkant v/h karakter.
Dit kan door vele controles uit te voeren en vele lijnen te resetten, maar dan wordt het geheel te groot en te traag.
Als U een redelijk korte en efficiënte oplossing voor dit probleem heeft, of U heeft vragen, opmerkingen of aanvullingen, neem dan kontakt met het clubblad op of met mij.
Ik zal in de tweede (laatste?) aflevering meer variaties geven en waarschijnlijk nog over andere onderwerpen uitwijden.
Rest mij nog U veel sterkte te wensen!

MARTIJN HONDEMA
YLOSTINSLAAN 75
NL8651 AS IJLST
NEDERLAND

KRUIMEL.BAS

KRUIMEL BAS D1

Een spelprogramma voor de MSX 1 en de MSX 2 computer. De bedoeling is om met een kruimeldief zo snel mogelijk (tegen de tijd) de kruimels van de tafel af te ruimen. U start met 10 kruimels op de tafel. Na elke geslaagde opkuisbeurt volgt er een nieuwe tafel met telkens één kruimel meer. Er wordt een score bijgehouden. De besturing gebeurt zowel met de joystick als met het toetsenbord.
(Timmermans Luc)

MEERMIN.BAS

MEERMIN BAS D1

Een edukatief programma. De bedoeling is dat de leerling uit een aantal getallen het meeste, het minste, het op één na meeste en het op één na minste getal kan herkennen. Dit programma werd geschreven voor de MSX 2 computer.
De moeilijkheidsgraad kan worden ingesteld en er is visuele ondersteuning.
(Desaeger Joris)

RACER.BAS

RACER BAS D1

Een spelprogramma voor de MSX 1 en de MSX 2 computer. Als autopiloot dient u halsbrekende toeren te verrichten om het einde van deze 'danger race' te bereiken. Uw reaktiesnelheid zal danig op de proef worden gesteld. De score en de high-score wordt bijgehouden. De besturing gebeurt met de joystick.
(Roels Guy)

TERMINUS.BAS

TERMINUS BAS D1

Een cheat-programma voor het spel 'Terminus' van Mastertronic. Een programma voor een onbeperkt aantal kogels en energie.
(Galarza Sjm)

YESTERD.BAS

YESTERD BAS D1

Dit is het zeer bekende liedje van de Beatles. Vervlogen romantiek gekruid met een digitaal tintje. Leuk !
(Stroobants Bart)

GEBRUIK VAN LOGISCHE OPERATOREN

Door bij de scroll-routines gebruik te maken van logische bewerkingen staan er legio nieuwe variaties open. Ik zal eerst een korte beschrijving van de logische functies geven.

LOGISCH?

Bij iedere logische bewerking worden twee getallen (van doorgaans 8 of 16 bits) bit voor bit met elkaar vergeleken. Het gaat hier steeds om een vergelijking tussen twee even significante bits, bv. bit 6 met bit 6, bit 3 met bit 3 enz.
De uitkomst (0 of 1) hangt van de waarden van beide bits af en wordt het even significante bit van het resultaat. De mogelijke logische operatoren zijn **AND, OR, XOR, NOT, EQV** en **IMP**. Als u ze decimaal bekijkt lijken ze echter allesbehalve logisch... Of ziet u wel direkt waarom 234 AND 156 gelijk is aan 136 terwijl 134 AND 156 132 is en 96 AND 156 = 0?

WEDEROM BINAIR

Om de werking van de logische operatoren goed te begrijpen is het nodig om de betreffende getallen binair te bekijken. In de vorige aflevering had ik al vermeld hoe een binair getal decimaal te schrijven is. Als de opbouw van een binair getal u duidelijk is, zal het eveneens weinig problemen geven om een decimaal getal binair te schrijven. Voor een 8-bits getal begint u dan bij bit 7 en kijkt of het om te zetten getal groter is dan de waarde van dit bit (2^7=128; zie fig. 2 in de vorige aflevering). Als dat inderdaad zo is, trekt u deze waarde van het getal af en is bit 7 van het resultaat gelijk aan 1. Zo niet, dan is bit 7 0 en verandert het getal niet.
Vervolgens onderzoekt u het volgende bit; dit is bit 6 en heeft de waarde 2^6=64. Volgens dezelfde procedure bepaalt u of bit 6 van het resultaat 1 of 0 moet worden naar gelang de grootte van de rest van het getal. Deze bewerking doet u voor alle bits en als resultaat heeft u dan een binair getal. Dit alles kan natuurlijk ook vanuit basic met de BIN$-functie bv. ? BIN$(156).

LOGISCH!

Als u uitgaat van de binaire getallen, zit er wel een duidelijke logica in deze dito bewerkingen. Dit is het duidelijkst als u de getallen binair onder elkaar zet en de bits dan (vertikaal) vergelijkt.
Bij de AND-bewerking zal het bit van van het resultaat 1 zijn als twee gelijkwaardige (even signifikante) bits van de te vergelijken getallen beide 1 zijn; anders is het resultaat 0.
Zo kunnen we schrijven voor 234 and 156:

```
11101010   =   234
10011100   =   156
--------   ---  AND
10001000   =   136
```

Op deze manier kunt u 134 AND 156 = 132 en 96 AND 156 = 0 ook kontroleren.
In schema zien de bewerkingen er als volgt uit:
A:BIT1=0 BIT2=0; B:BIT1=1 BIT2=0; C:BIT1=0 BIT2=1; D:BIT1=1 BIT2=1

RESULTAAT BIJ:	A	B	C	D
AND	0	0	0	1
OR	0	1	1	1
XOR (=EXCLUSIVE OR)	0	1	1	0
EQV	1	0	0	1
IMP	1	1	0	1

De NOT-operator, tenslotte, heeft slechts betrekking op 1 getal en 'set' een bit van het resultaat als dat van het orgineel 0 is en vice versa; zo krijgt u de inverse waarde.

GEBRUIK

Hoe vallen deze logische bewerkingen nu toe te passen in de scroll-routines? Een mogelijkheid is om samengestelde scrollingen te maken door de twee waarden waarmee de lijnen van het karakter geset worden (bv. E en F) bij verschillende scroll-routines, te 'mergen' door ze op elkaar te projekteren.

OR-PROJECTIE

Met andere woorden: als je beide waarden in binaire vorm onder elkaar schikt, zet je op elke vertikale positie waar in minimaal een van beide waarden een 1 voorkomt, in het resultaat ook een 1. Zoals u ziet, is dit precies de werking van OR-funktie. Zo is het erg eenvoudig om bv. een diagonale scrolling te combineren met een vertikale omlaag. We houden gewoon de telling en de VPOKEs aan van een scrolling omlaag (dit zijn de regels 1010 en 1020) terwijl we de te VPOKEn waarde op iedere lijn berekenen door de waarde van een diagonale scrolling omlaag (2^G+2) te kombineren met de waarde van een enkel scrollend punt omlaag middels een OR-bewerking. Om de waarde van één punt te vinden, had ik in de vorige aflevering al aangetoond dat dit altijd een macht van twee moest zijn. Hiervoor neem ik in de waarde 16 maar in principe voldoet iedere macht van twee kleiner dan 256. Verder moet de alternatieve waarde - als de teller groter is dan 5 (dus 6 of 7) - nu niet 128 zijn, maar 128 OR 16. Dit alles ziet terug in regel 1015.
De berekening van F wordt dus (2^G+2) OR 16 en de listing:

#A-010A BAS D2

```
1000 'Routine #A.003a : ZW

1010 G=G+1:IF G>7 THEN G=0
1015 IF G>5 THEN F=128 OR 16 ELSE F=2^(G+2) OR 16
1020 IF G<7 THEN VPOKE S+G+1,F:VPOKE S+G,0:RETURN
 ELSE VPOKE S,F:VPOKE S+7,0:RETURN
```

* In de DATAVERSIE ligt het nog eenvoudiger: omdat de te setten waarde hier gewoon wordt ingelezen (als F) kunnen we hier zonder meer OR 16 achter zetten bij een herdefiniering ervan na dit inlezen.

#A-010B BAS D2

```
1000 'Routine #A.003b : ZW

1010 G=G+1:IF G>7 THEN RESTORE 1030:G=0
1015 READ F:F=F OR 16
1020 IF G<7 THEN VPOKE S+G+1,F:VPOKE S+G,0:RETURN
 ELSE VPOKE S,F:VPOKE S+7,0:RETURN
1030 DATA 4,8,16,16,32,64,128,128
```

* En het is duidelijk dat bit 5 (waarde 16) netjes 'aan blijft staan'. Aanpassingen voor OOSTELIJKE en NOORDELIJKE richtingen liggen nu ook voor de hand. Dit is slechts een kwestie van de waardedefinieringen van F aan te vullen met 'OR 16' of een andere passende waarde in de orginele routines uit de vorige aflevering.

Een andere mogelijkheid is om twee diagonale scrollingen te kombineren. Als we weer uitgaan van een vertikale scrolling omlaag, combineren we nu de beide formules F=2^(G+2) en F=2^(7-G) uit resp. #A.003a en #A.004a tot F=2^(G+2) OR 2^(7-G). Omdat beide 'overflow'-controles op hetzelfde moment plaats moeten vinden (nl. als G groter is dan 5), kunnen we deze ook kombineren tot F=128 OR 4.

#A-011A BAS D2

```
1000 'Routine #A.011a : ZW + ZO

1010 G=G+1:IF G>7 THEN G=0
1015 IF G>5 THEN F=128 OR 4 ELSE F=2^(G+2) OR 2^(
7-G)
1020 IF G<7 THEN VPOKE S+G+1,F:VPOKE S+G,0:RETURN
 ELSE VPOKE S,F:VPOKE S+7,0:RETURN
```

* Echt het idee van twee diagonale scrollingen heb je zo echter niet meer, omdat op lijn 6 en 7 de twee punten (bit 0 en bit 7) van twee naast elkaar liggende karakters tegen elkaar plakken. Om hier in de diagonale scrolling twee kleine knikken i.p.v. een grote te zetten, en dan ook nog op posities die niet tegen elkaar aan liggen, moet er nu een dataversie gemaakt worden met in de data-lijst welke macht van twee gezet moet worden (zonder deze uit te schrijven als getal), omdat anders de waarde van het tweede punt 2^(7-G) niet uit te rekenen is. Nu is deze echter direct af te leiden uit de waarde van de macht van twee die voor het eerste punt werd ingelezen.

#A-011B BAS D2

```
1000 'Routine #A.011b : ZW + ZO

1010 G=G+1:IF G>7 THEN RESTORE1030:G=0
1015 READ E:F=2^(E+2) OR 2^(7-E)
1020 IF G<7 THEN VPOKE S+G+1,F:VPOKE S+G,0:RETURN
 ELSE VPOKE S,F:VPOKE S+7,0:RETURN
1030 DATA0,1,1,2,3,4,4,5
```

AND-MASKER

We hadden al gezien dat de AND-funktie dient om te kontroleren of beide te vergelijken bits al dan niet 1 zijn. In de praktijk wordt deze operand vaak gebruikt als masker, d.w.z. om bepaalde bits af te schermen door ze te resetten. Dit werkt als volgt: de te ressetten bits van het getal moeten overeenkomen met bits 0 van de AND-operand. Het resultaat zal dan altijd 0 zijn, immers slechts als beide bits 1 zijn wordt het resultaat 1 en met één 0-bit wordt het dus 0.

Voorspelbaar is het dan, dat de niet af te schermen bits, die dus gewoon hun oude waarde dienen te behouden, ge-AND moeten worden met 1. Is het originele bit 0, dan is het resultaat 0 AND 1 = 0, anders 1 AND 1 = 1.
Zo verandert er dus niets aan deze bits.

Als we dit toepassen op de gecombineerde diagonale scrollingen van #A.011 kan dat bv. door de linkse 3 kolommen af te schermen, zodat de beide te setten punten ook niet meer aan elkaar plakken en een aardig effect het gevolg is. Dit wordt dus AND &B00011111 oftewel AND 31.

```
1000 'Routine #A.012a : Z
                    #A-012A BAS D2
1010 G=G+1:IF G>7 THEN G=0
1015 IF G>5 THEN F=128 OR 4 ELSE F=(2^(G+2) OR 2^
(7-G)) AND 31
1020 IF G<7 THEN VPOKE S+G+1,F:VPOKE S+G,0:RETURN
 ELSE VPOKE S,F:VPOKE S+7,0:RETURN
```

```
1000 'Routine #A.012b : Z
                    #A-012B BAS D2
1010 G=G+1:IF G>7 THEN RESTORE1030:G=0
1015 READ E:F=(2^(E+2) OR 2^(7-E)) AND 31
1020 IF G<7 THEN VPOKE S+G+1,F:VPOKE S+G,0:RETURN
 ELSE VPOKE S,F:VPOKE S+7,0:RETURN
1030 DATA0,1,1,2,3,4,4,5
```

* U kunt zelf waarschijnlijk wel meer dergelijke voorbeelden bedenken. Alle mogelijke maskers en projecties zijn nu eenvoudig te verwezenlijken; u kunt elke kolom die u wilt afschermen en alle funkties kombineren. Nog een mogelijkheid is om de te setten waarde te inverteren (1 wordt 0 en 0 wordt 1) d.m.v. XOR 255 (lees: XOR &B11111111). Ga dit zelf maar eens na... Let er wel op dat als u twee funkties kombineert, ook de alternatieve waardedefinieringen (vgl. IF G5) gekombineerd moeten worden, zoals F=128 OR 4 bij de beide diagonalen.

We hebben nu al aardig wat reperterende smooth-scrollingen gezien waarbij we steeds niet meer dan één of twee lijnen hebben geset. Hierdoor ging het gebruikte karakter echter geheel verloren. Is het nu niet mogelijk een heel karakter te scrollen, zodat de vorm ervan behouden blijft?

SCROLLINGEN MET GEHELE KARAKTERS

Om te beginnen met een vertikale scrolling; u zult merken dat dit inderdaad vrij eenvoudig kan, maar het is wel bewerkelijk voor uw arme computer. Het werkt als volgt:

Als we de procedure lijn voor lijn bekijken en we gaan uit van een vertikale scrolling omhoog, dan moeten we iedere lijn als het ware oppakken en een lijn hoger plaatsen. Hierbij dreigt echter direkt al een probleem: als we lijn 0 oppakken en op lijn -1 plaatsen, zodat lijn 0 vrijkomt voor de waarde van lijn 1 in de beurt daarna, 'setten' we deze lijn buiten het karakter. Hier hoeft u zich echter geen zorgen over te maken; deze lijn fungeert namelijk als buffer. Dit moet er namelijk gebeuren:

lijn 0	-	lijn -1
lijn 1	-	lijn 0
lijn 2	-	lijn 1
.....		
lijn 7	-	lijn 6

Dit zijn dus 8 handelingen, wat u in de listing kunt nalezen als 'FOR G = S-1 TO S+6'. En dan is de situatie als volgt: het hele karakter is 1 lijn omhoog geset waardoor het 1 lijn (lijn -1) buiten zijn territorium steekt, terwijl lijn 7 geen functie meer heeft omdat zijn waarde al op lijn 6 staat.

Hoe zetten we dit weer recht? De oplossing ligt nu wel voor de hand: we moeten de lijn -1 pakken, en die op lijn 7 zetten. En dan is het karakter werkelijk een lijn opgeschoven. Als u nu i.p.v. lijn 0 lijn S leest (dit was immers het beginadres van het karakter) en i.p.v. bv. lijn -1 lijn S-1, gebeurt hier in regel 1000 (listing #A.013) exact hetzelfde. Met de VPOKEs wordt de lijn, die met de VPEEK direkt erachter wordt ingelezen, 1 lijn lager geset.

Verder ziet u dat ik een extra regel 11 opgenomen heb, waarin karakter 219 een vorm krijgt toegewezen met de formule 128-2^G, die u nog uit de vorige aflevering kunt herkennen. Dit is om het karakter een herkenbare vorm te geven; immers een vertikale scrolling bij een karakter waarvan alle lijnen gelijk zijn heeft weinig zin. Deze regel 15 moet u, als u CHR$(219) gebruikt zonder deze gedefinieerd te hebben, invoegen in de test/demo.

Maar natuurlijk voldoet ieder ander karakter of iedere andere vorm ervan ook. Let wel: omdat lijn -1 - dus een lijn van het voorgaande karakter aan het te setten karakter - als buffertje wordt gebruikt, verandert de vorm ervan en dus ook de vorm van lijn 7 van dit voorgaande karakter.

```
11 FOR I=S TO S+7:VPOKE I,128-2^(I-S):NEXT
1000 'Routine #A.013
                    #A-013 BAS D2
1010 FOR G=S-1 TO S+6:VPOKE G,VPEEK(G+1):NEXT:VPO
KE S+7,VPEEK(S-1):RETURN
```

AANPASSINGEN SCREEN 1

Het zou natuurlijk ook erg handig zijn de scrollingen op screen 1 te kunnen gebruiken. De theorie werkt op dit scherm even goed als op screen 0, en vanwege de breedte van de karakters (niet 6, maar 8 pixels breed) en de kleurmogelijkheden op dit scherm biedt dit weer vele nieuwe mogelijkheden, bv. om een mooie tekstintro bij een programma te maken met kleuren en animatie. Echter, we kunnen niet onder een kleine aanpassing uit, maar deze is erg miniem. De geheugenindeling van screen 1 in het VRAM is namelijk iets anders. Dit komt ondermeer omdat op screen 1 3 extra geheugengedeelten in het VRAM aanwezig zijn voor de kleur- en spritemogelijkheden, nl. BASE(6), BASE(8) en BASE(9). Deze zijn nu niet van belang, wel de nieuwe positie van de patroontabel, waarin de vorm van de karakters in staat. Dit was op screen 0 BASE(2) (=2048) en is op screen 1 BASE(7) met als waarde 0. De formule voor een plaats van een karakter in de patroontabel op screen 0 was: ASCII-WAARDE * 8 + 2048 (zie magazine nr. 16). Dit wordt op screen 1 dus gewoon ASCII-WAARDE * 8.

Nu kunnen we dus iedere keer de waarde voor S op zowel screen 0 als screen 1 berekenen. Maar het zou gemakkelijker zijn om het programma zelf dit te laten uitzoeken, zodat het programma 'zelfstandig' op beide schermen kan werken en dus algemeen toepasbaar is. Hiervoor moet het programma wel weten op welk scherm het werkt. Gelukkig houdt de computer, geordend als hij is, dit altijd bij in een geheugenplaats in de systeemwerkruimte (lopend van adres &HF381 to &HFFFF), te weten &HFCAF. Hier staat als byte het schermnummer, dus 0 op screen 0 en 1 op screen 1. Als we S nu gewoon de ASCII-waarde van het te gebruiken karakter geven veranderen we dit direkt daarna in het programma tot het beginadres van dit karakter in het VRAM. Dus in ons voorbeeld: S=219. En om maar in de TEST/DEMO te blijven (dit wordt de nieuwe regel 10), na de voorlopige definiering van S als 219, lezen we &HFCAF uit. Nu moeten we S aanpassen tot de startwaarde in het VRAM. Op screen 0 wordt deze waarde ASCII-WAARDE * 8 + 2048 en op screen 1 ASCII-WAARDE * 8. De uiteindelijke definieringen van S worden dus respectievelijk S*8+2048 en S*8. In de oude regel 10 werd ook het scherm en de schermbreedte gedefinieerd, maar omdat het programma op beide (van te voren ingestelde) tekstschermen moet kunnen werken dienen we dit te verwijderen. Regel 10 wordt dus:

```
10 KEY OFF:COLOR14,1: A=50:S=219: R=
PEEK(&HFCAF): IF R0 THEN S=S*8 ELSE S=S*8
+2048.
```

En nu is ons programma al helemaal zelf-aanpassend. Alleen zijn er op screen 0 nog wat LAY-OUT foutjes in de TEST/DEMO vanwege de 32 kolommen.
Dit is met wat kleine ingrepen te verhelpen:

- WIDTH 40-8*R voor maximale schermbreedte,bv in regel 10 na definiering van G
(G is 0 of 1 voor screen 0 resp. 1)
- STRING$(40-8*R,219) in regel 20
- LOCATE 20-LEN(A$)/2-4*R,5+I*3 in 30
- LOCATE 5-4*R,2 i.p.v. LOCATE5,2 in 30
-LOCATE 18-4*R,21 i.p.v. LOCATE 18,21 in 50
(2 keer)

...zodat we ons nu verder niet meer om het schermtype hoeven te bekommeren. Het is erg verstandig deze veranderingen aan te brengen, omdat de oude test/demo niet goed op screen 1 werkte. Verder is het zeer aan te raden nog een extra RESTORE 530 aan het begin van regel 30 te plaatsen want anders gaat het bij gebruik van nog een andere DATA-lijst, zoals in #A.014ML, verkeerd met het inlezen.

Nu terug naar #A.013:

NADELEN

U ziet: De routine werkt. Maar dat wil nog niet zeggen dat dit tot volle voldoening zal zijn, want er zijn aan deze routine enige nadelen verbonden. Het figuurlijk gezien het meest in het oog springende is wel de snelheid. Omdat de computer voor iedere lijn dat het karakter opschuift maar liefst 9 VPOKE-handelingen moet uitvoeren, is het onmogelijk een redelijk hoge scroll-snelheid te bereiken vanuit basic en neemt de routine bovendien te veel (kostbare!) computertijd in beslag. Een mooiere oplossing zal ik hierna geven. Tweede minpunt, nu letterlijk in het oog springend, is dat de scrolling niet vloeiend verloopt. Ook dit is te wijten aan de lage snelheid in BASIC, waardoor de VPOKES zo snel achter elkaar komen, dat ze elkaar schijnbaar overlappen, hetgeen een erg onrustig beeld geeft. Dit komt weer omdat er onvermijdelijk enige tijd zit tussen het setten en het resetten van de lijnen, waardoor er altijd eventjes of twee lijnen geset zijn of geeneen, afhankelijk van welke instructies het eerst in de routine staan.

DE OPLOSSING

Als het in basic niet goed kan, is de meest logische stap om naar een lagere programmeertaal te grijpen; een programmeertaal dus, die dichter bij de microprocessor staat en daardoor veel sneller en efficienter is, maar ook moeilijker te begrijpen en wellicht minder overzichtelijk. Omdat talen als PASCAL en C echter niet voor iedere MSX-er beschikbaar zijn en ik hier zelf ook nauwelijks ervaring mee heb, heb ik voor een machinetaal-benadering gekozen, die voor elke MSX-er bruikbaar is (immers, iedere computer werkt standaard al in machinetaal, terwijl de BASIC-compiler een algemeen aanvaard extra is. Dit moesten de computers van de eerste generaties echter ontberen!). Dat we deze voornoemde snelheid in machintaal zeer goed kunnen gebrui ken, was al duidelijk. Om de routine te begrijpen, is

enige kennis van machinetaal en MNEMONICS wel vereist; gebruiken echter kan iedereen hem.

Om de routine te kunnen gebruiken, moet de machinecode eerst in het geheugen geladen worden en het executieadres ingegeven. Dit kan met de volgende regel, die ik als regel 12 geef om in te kunnen voegen in de test/demo:
12 RESTORE 1030: FOR I=&HE000 TO &HE030:READ A$:POKE I,VAL("&H"+A$):NEXT :DEFUSR=&HE000

Handhaaf ook de eerder genoemde regel 11 om CHR$(219) vorm te geven, indien u dat in de BASIC-versie van deze routine ook deed. Achter RESTORE het eerste regelnummer met MT-data zetten.

Dergelijke routines, die in combinatie met BASIC programma's gebruikt worden, kunnen het beste zo hoog mogelijk in het geheugen geplaatst worden. Dit is omdat de BASIC programma's vanaf adres &H8000 omhoog groeien in het geheugen, en als ze de machinecode 'ingehaald' hebben, volgt er bij een nieuwe aanroep ervan onvermijdelijk een crash (de computer reset of loopt vast) en heeft u van geen van beide veel plezier meer... Ik heb de machinecode hier vanaf adres &HE000 laten lopen, maar dit kan problemen geven als u twee of meer diskdrives aangesloten heeft. Daarom is het belangrijk om te weten dat ik de routine RELOCATABLE heb gemaakt. Dit houdt in, dat u de routine overal in het beschikbare RAM-geheugen kunt plaatsen zonder dat er problemen kunnen komen. Dit is te bereiken, door alleen relatieve sprongopdrachten te gebruiken (JR), die een te springen afstand aangeven en geen absoluut 'aankomstadres'.

Ik geef de machinecodes hier als BASIC-data, met achter de hexadecimale instructies als remarks het adres van de eerste byte van de instructie wanneer u de machinecode op &HE000 start, vervolgens de eventuele labels, en tenslotte de mnemonic's.

De machintaal-versie doet in principe hetzelfde als de basic-uitvoering:
- In de regels 1010 en 1040 worden de labels RDVRAM (READ VRAM) en WRVRAM (WRITE VRAM) gedefinieerd. Dit zijn BIOS-calls zodat er geen compatibiliteitsproblemen kunnen zijn, in tegenstelling tot rechtstreekse IN en OUT-commando's. Het zijn de equivalenten van de VPEEK en VPOKE-instructies in basic, maar dan met als te lezen/schrijven VRAM-adres de inhoud van HL.
- In de regels 1030 t/m 1080 wordt het schermnummer opgehaald uit &HFCAF, vergeleken met 0, waarna afhankelijk van het resultaat HL op 219*8+2048 = 3800 of 219*8 = 1752 wordt gezet voor screen 0 resp. screen 1. Wilt u met een ander karakter werken dan CHR$(219), dan moet u het startadres ervan op screen 0 en 1 uitrekenen volgens eerdergenoemde formules, deze hexadecimaal schrijven (bv. ?HEX$(GETAL)) , en ze dan in de listing veranderen in regel 1060 (waarde voor screen 0) en 1080 (waarde voor screen 1) als TWEEDE en DERDE byte. Het eerste byte is namelijk de op-code voor LD HL.

#A-014ML BAS D2

```
11 FOR I=S TO S+7:VPOKE I,128-2^(I-S):NEXT
12 RESTORE1030:FOR I=&HE000 TO &HE029:READA$:POKE
 I,VAL("&H"+A$):NEXT:DEFUSR=&HE000
1000 U=USR(0):RETURN

1010 '42 BYTES        RDVRAM:  EQU 4AH
1020 '                WRVRAM:  EQU 4DH

1030 DATA 3A,AF,FC:'E000          LD   A,(FCAFH)
1040 DATA FE,00   :'E003          CP   0
1050 DATA 20,05   :'E005          JR   NZ,S1
1060 DATA 21,D8,0E:'E007          LD   HL,3800
1070 DATA 18,03   :'E00A          JR   VERDER
1080 DATA 21,D8,06:'E00C S1:      LD   HL,1752
1090 DATA 06,08   :'E00F VERDER:  LD   B,8
1100 DATA CD,4A,00:'E011 LUS:     CALL RDVRAM
1110 DATA 2B      :'E014          DEC  HL
1120 DATA CD,4D,00:'E015          CALL WRVRAM
1130 DATA 23      :'E018          INC  HL
1140 DATA 23      :'E019          INC  HL
1150 DATA 10,F5   :'E01A          DJNZ LUS
1160 DATA 2B      :'E01C          DEC  HL
1170 DATA 01,08,00:'E01D          LD   BC,8
1180 DATA ED,42   :'E020          SBC  HL,BC
1190 DATA CD,4A,00:'E022          CALL RDVRAM
1200 DATA 09      :'E025          ADD  HL,BC
1210 DATA CD,4D,00:'E026          CALL WRVRAM
1220 DATA C9      :'E029          RET

(c) MSX-CLUB
```

Let wel dat u dit getal in de volgorde LEAST SIGNIFICANT BYTE - MOST SIGNIFICANT BYTE invoert op deze bytes, dus dat u de LAAGSTE twee hexadecimale cijfers op het tweede byte zet en de HOOGSTE op de derde byte, immers de oprand na de op-code (BYTE 1) heeft de waarde BYTE 2 + 256 * BYTE 3.
- Op 1090 en 1150 wordt een lus gecreeerd die 8 keer doorlopen wordt. Hiertussen wordt iedere keer lijn HL (= lijn S) gelezen, HL met 1 verlaagd en dus op lijn S-1 weer weggeschreven. Om HL (=lijn S-1) op de volgende te lezen lijn te zetten (lijn S+1), moet HL daarna nog met 2 verhoogd worden. Als dit 8 keer uitgevoerd is, is het hele karakter 1 lijn omhoog geschoven. HL is dan, na voor het laatst met 2 opgehoogd te zijn, gelijk aan S+8 (dus een lijn boven het karakter).
- In regel 1160 wordt HL met 1 verlaagd zodat deze teller weer netjes op lijn 7 van het karakter staat, klaar voor de laatste ingreep:
- regels 1170-1220: nu moet lijn -1 nog uitgelezen worden en op lijn 7 geplaatst worden. M.a.w. verlaag HL met 8, lees VRAM uit, verhoog HL met 8, schrijf VRAM. Nu bestaat er geen instructie als SBC HL,8 (aftrekken met carry omdat het 16-bits getallen zijn), maar wel SBC HL,BC. Dus via een kleine omweg lukt het even goed: maak BC gelijk aan 8, verlaag HL met BC, lees VRAM uit, verhoog HL met BC, schrijf VRAM.

TESTDEM2 BAS D2

- Een RETURN sluit de routine af, en omdat er geen CALL aan voorafgegaan is, keren we zo weer terug naar BASIC.

```
1 '***********************
2 '*  TEST DEMO      *
3 '***********************
4 '* Demo rep. smooth scrolling screen 0
5 '* voor het aanroepen van de gepubli-
6 '* ceerde routines waarvan de titel
7 '* met #A begint.
8 '
```

```
10 S=219:KEY OFF:COLOR 14,1:R=PEEK(&HFCAF):WIDTH
40-8*R:IF R<>0 THEN S=S*8 ELSE S=S*8+2048'A=start
snelheid Scrolling
11 FOR I=S TO S+7:VPOKE I,128-2^(I-S):NEXT
12 RESTORE1030:FOR I=&HE000 TO &HE030:READA$:POKE
 I,VAL("&H"+A$):NEXT:DEFUSR=&HE000

15 ' Vult het scherm met chr$(219). Als U
16 ' een ander karakter gebruikt dient U
17 ' behalve S in regel 10 ook de waarde
18 ' 219 in regel 20 aan te passen.
```

```
20 FOR I=0 TO 24:LOCATE 0,I:PRINT STRING$(40-8*R,
219);:NEXT
30 RESTORE 530:FOR I=0 TO 4:READ A$:LOCATE 20-LEN
(A$)/2-4*R,5+I*3:PRINTA$:NEXT:LOCATE5-4*R,2:PRINT
"Gebruik cursor >< voor snelheid":' PRINT TEKST
40 FOR I=1 TO A:IF STICK(0)=0 THEN NEXT:GOSUB 100
0:GOTO 40 ELSE I=A:NEXT:GOTO 50
50 IF STICK(0)=3 THEN A=A+1:LOCATE 18-4*R,21:PRIN
TA:GOTO 40 ELSE IF STICK(0)=7 AND A>1 THEN A=A-1:
LOCATE 18-4*R,21:PRINTA:GOTO 40:NEXT ELSE 40

500 ' Tussen de woorden i.p.v. spa-
510 ' ties chr$(219) typen (=GRAPH + P)
```

```
530 DATA DEMO SCREEN 0,REPETERENDE,SMOOTH,SCROLLI
NG,door*MARTIJN*HONDEMA
```

```
(c) MSX-CLUB
```

VOLGENDE KEER...

De volgende keer zal ik ook een dergelijke machinetaal routine geven voor een horizontale scrolling, iets wat in de BASIC-versie zeker niet aanvaardbaar is i.v.m. de snelheid. Hiermee zijn aardige variaties te creeren. Tevens zal ik mogelijkheden geven om de gehele karakterset te herdefinieren zodat andere lettertypes via korte berekeningen beschikbaar zijn.

Martijn Hondema

DALLAS.M16

DALLAS M16 D1

De auteur vond zijn inspiratie voor dit programma in het tijdschrift 'Humo'. Dankzij deze scenarioschrijver hoeft Dallas nooit meer van het scherm te verdwijnen want er worden steeds meer en meer plots gecreëerd.
(Coremans Raf)

HAVANAGI.M16

HAVANAGI M16 D1

Dit is een zeer bekend oosters wijsje. Je kan er zelf nog wat aan sleutelen zodat het iets hoger of lager of iets sneller of trager wordt gespeeld. Leuk !
(Pauwels Gert)

MISIRLOU.M16

MISIRLOU M16 D1

Dit is een zeer plezant klinkend oosters wijsje. Het doet ons sterk denken aan het achtergrondwijsje van KING'S VALLEY van Konami.
(Pauwels Gert)

RENBAC.M16

RENBAC M16 D1

Met dit programma kan de ijverige spaarder narekenen of zijn bankinstelling wel de juiste berekeningen heeft gemaakt. Het programma geeft de mogelijkheid om een afdruk op papier te maken.
(Wuyts Fons)

RESUME.M16

RESUME M16 D1

Resume is gebaseerd op het kaartspel 'gele dwerg'. De bedoeling is om zo vlug mogelijk van al uw kaarten vanaf te raken. De nodige uitleg wordt, indien gewenst, gegeven.
(Schepers Rohnny)

MEERMIN.DES

MEERMIN DES D1

Een educatief programma dat voor de beginnende cijferaar best bruikbaar en leerzaam is.
(Desaeger Joris)

SCROLLINGEN MET HELE KARAKTERS

HORIZONTAAL

Na de routine voor een vertikale scrolling van een karakter, is het nu tijd om eens te kijken hoe iets dergelijks horizontaal mogelijk is. Dit kan niet slechts door VRAM adressen uit te lezen en elders weer weg te schrijven; want dan verandert er horizontaal nog niets aan het karakter. De lijnwaarden blijven dan immers gelijk! Hoe moet het wel?

Om een lijn een positie bv. naar rechts te schuiven, moet ieder bit een plaats naar rechts verschoven worden. Vervolgens moet, net als bij de vertikale scrolling, het 'uitstekende' bit weer aan de andere kant naar binnen geschoven worden, en de lijn staat in z'n geheel een positie opgeschoven.

BIT VOOR BIT...

Voor bitmanipulaties is onze Z80a uitgebreid uitgerust. Ieder bit van elk register kan geset, gerest en getest worden. Dat deze instrukties voor enkelvoudige registers een byte beslaan, is trouwens wel opmerkelijk. Van de 253 beschikbare bytes als eerste op-code voor de Z80 (0 en 255 zijn NOP-codes en dienen slechts om het lege geheugen te vullen), nemen SET, RES en BIT (=test) er elk 64 in beslag! Dat betekent dat er nog maar 61 overblijven voor andere instrukties, die daardoor vaak een opcode van twee bytes moeten hebben.

OF IN EEN KEER

Het verschuiven van de bits van lijn m.b.v. de bitmanipulaties kost echter erg veel tijd omdat het vaak moet gebeuren. Er is een veel efficientere oplossing, en deze schuilt in de roteerinstrukties. Hiermee kunnen we de inhoud van een processorregister een positie verschuiven, waarbij het uitstekende bit naar keuze al dan niet via da carry-flag aan de andere kant het register weer binnentreedt. We hebben er niks aan als de carry meescrollt, dus moeten we de rotatie hebben die het uittredende bit direkt weer achteraanschuift in het register. Dit is RRC (Rotate Right Circular) om naar rechts te schuiven en RLC (U raadt het al: Rotate Left Circular) voor links. Dit zijn normaal gesproken tweebytes instrukties, maar er bestaan van beide een bijzondere uitvoering (RRCA en RLCA) die de accu roteren en maar 1 byte nemen. Omdat het door RDVRAM uitgelezen adres toch in de accu komt, zal ik deze gebruiken.

SCHUIVEN MAAR!

Met behulp van deze roteerinstrukties is een horizontale scrolling niet moeilijk meer. In #A-015ML ziet u hoe dit toe te passen is.

- de regels 11 en 12 zijn net als in #A-014ML om CHR$(219) te definiëren en de MT-data te instaleren. Als extra heb ik, voor als u op screen 1 werkt, de kleur van CHR$(216) t/m CHR$(223) veranderd.
- Op 1010 en 1020 worden zoals gebruikelijk de labels RDVRAM en WRVRAM gedefinieerd. Deze gelden uiteraard alleen voor de MNEMONINCS, voor de pure machinecodes worden deze vertaald in (relatieve) sprongadressen.
- Registerpaar DE houdt bij hoeveel VRAM-adressen (8 per karakter) per aanroep verschoven moeten worden. Omdat we in dit geval maar 1 karakter gebruiken wordt DE met 8 geladen. Tijdens de 8e aanroep zal DE weliswaar nog steeds 1 zijn, maar omdat pas na de DEC DE op regel 1140 wordt gekeken of DE 0 is (en dat is DE dan natuurlijk), wordt de lus daarna niet nogmaals doorlopen.
- Op de al eerder gebruikte manier wordt HL, dat het te VPOKEN adres bevat, afhankelijk van de schermmode, geladen met 2048+219*8 (=3800), danwel 219*8 (=1752).
- De LUS loopt van 1100 tot en met 1170. Met CALL RDVRAM wordt byte (HL) uitgelezen, daarna vindt de roteerinstruktie plaats, en vervolgens wordt de geroteerde byte, die nog steeds in de accu staat, weer weggeschreven met CALL WRVRAM.
- HL bevat de waarde van het te roteren byte in het VRAM. Deze moet steeds 1 verhoogd worden met INC HL.
- DE was de teller, die bijhoudt hoeveel bytes nog 'behandeld' moeten worden. Deze wordt dus steeds 1 verlaagd. Omdat dit aantal bytes groter dan 255 kan worden, heb ik een registerpaar gebruikt en niet een enkel register. Dit heeft als nadeel, dat de CP instruktie niet gebruikt kan worden.

Hoe kunnen we nu weten wanneer DE 0 is? Daarvoor zijn twee mogelijkheden:

- Ten eerste kunnen we de registers D en E afzonderlijk vergelijken met 0, maar dit kost veel extra instrukties; - Een snellere truuk is de volgende: we zetten de waarde van D (Of van E, dat maakt in dit geval niets uit) in de accu, en geven hierop een OR-projectie van register E (of D). Nu zal, als er ook maar een bit (van D of E) 0 is, dit bit ook in het resultaat voorkomen, en er zal dan geen 0 in de accu komen te staan. Slechts als alle bits 0 zijn, zal de accu (=het resultaat) ook nul zijn.

- Nu is een CP 0 instruktie niet eens meer nodig, want de OR E in regel 1160 beinvloedt de Z-vlag al. De JR NZ,LUS zorgt ervoor, dat als de teller DE niet 0 is, de LUS herhaald wordt.

- Met een RETURN gaan we weer naar BASIC terug, en is het door DE en HL gedefinieerde VRAM-gebied een pixel naar rechts verschoven.

```
11 FOR I=S TO S+7:VPOKE I,128-2^(I-S):NEXT
12 RESTORE1000:FORI=&HE000TO&HE020:READA$:POKEI,VAL("&H"+A$):NEXT:DEFUSR=&HE000:VPOKE BASE(6)+27,99
1000 U=USR(0):RETURN
1010 '                          RDVRAM:  EQU  4AH
1020 '                          WRVRAM:  EQU  4DH
1030 DATA 11,07,00     :'E000           LD   DE,8          ;DE-aantal VRAM-bytes rotr.
1040 DATA 3A,AF,FC     :'E003           LD   A,(FCAFH)     ;Startadres te roteren
1050 DATA FE,00        :'E006           CP   0             ;bytes in HL afhankelijk
1060 DATA 20,05        :'E008           JR   NZ,S1         ;van schermmode
1070 DATA 21,D8,0E     :'E00A           LD   HL,3800
1080 DATA 18,03        :'E00D           JR   LUS
1090 DATA 21,D8,06     :'E00F S1:       LD   HL,1752
1100 DATA CD,4A,00     :'E012 LUS:      CALL RDVRAM        ;Roteer byte (HL)
1110 DATA 00,0F        :'E015           RRCA
1120 DATA CD,4D,00     :'E017 VERVLG2:  CALL WRVRAM
1130 DATA 23           :'E01A           INC  HL            ;verhoog byte VRAM (HL)
1140 DATA 1B           :'E01B           DEC  DE            ;verlaag teller
1150 DATA 7A           :'E01C           LD   A,D           ;is DE 0?
1160 DATA B3           :'E01D           OR   E
1170 DATA 20,F2        :'E02E           JR   NZ,LUS        ;zo niet->ga naar LUS2
1190 DATA C9           :'E020           RET                ;anders ->ga naar BASIC
```

#A-015ML BAS D2

Heeft u dit op screen 0 uitgeprobeerd? Dan zult u het met me eens zijn dat dit een weinig overtuigend resultaat geeft. Ik had al eerder vermeld dat een screen 0-karakter weliswaar 6 pixels horizontaal op het scherm inneemt, maar in het VRAM 8. En deze scrollen gewoon alle 8 mee. Maar op screen 1 heeft u dit probleem niet en is het resultaat meer bevredigend. Dit kunt u bijvoorbeeld erg goed in spelletjes gebruiken voor stromend water, rupsbanden o.i.d. Let er wel op dat de scrolling alleen werkt wanneer de te scrollen karakters die naast elkaar staan dezelfde zijn.

MEERDERE KARAKTERS

U kunt ook meerdere karakters gebruiken in deze routine. Het te scrollen geheugendeel wordt immers bepaald door DE en HL. Zo kan het gewenst zijn om de hele karakterset te scrollen. Door in #a-015ML de volgende regels te veranderen kan dat; de routine wordt echter wel aanzienlijk langzamer:

1030 DATA 11,00,08 :'E000 LD DE,2048
1070 DATA 21,00,08 :'E00A LD HL,2048
1090 DATA 21,00,00 :'E00F LD HL,0

Hiermee staan leuke mogelijkheden open op de tekstschermen. Als de karakters wegdraaien, lijkt het of ze volledig vernield worden, vooral op screen 0. Maar ze komen na 8 aanroepen weer terug!

VARIATIES

Misschien had u zich al verwonderd waarom de een-bytes instruktie RRCA op regel 1110 in de listing twee bytes neemt. Wel, het eerste byte 0 heeft geen funktie (NOP = No OPeration-byte), maar dient om dit byte te reserveren voor als we op die regel andere instrukties willen plaatsen. En daar is alle reden toe!

LINKS

- Het eerste waar we naar kunnen kijken is om de andere kant uit te scrollen: naar links. Daarvoor gebruiken we de equivalent voor RRCA: RLCA. Ook dit is een een-bytes instruktie en heeft als operatiecode 07H. Dus met 00H op adres E015H en 07H op E016H hebben we een scrolling naar links.

CLS

- Naast de roteerinstruktles, kennen we ook de schuifinstrukties (die zijn doorgaans twee bytes lang). Waar bij de roteerinstrukties ieder bit weer terugkomt in de lijn, gaat dit bij de schuifinstrukties naar de carry. Het bit dat bij de verschuiving vrijkomt, wordt naar keuze gevuld met 0 of houdt de-

zelfde waarde. Als wij nu voor het eerste geval kiezen, draaien alle karakters langzaam van het scherm af. Dit is een erg fraai gezicht, en kan dienen om eens een leukere CLS in uw programma's te gebruiken. Naar rechts schuiven kan met SRL (register) en naar links met SLA (register). Om de accu te schuiven worden dit de operatiecodes CB 3F resp. CB 27. Ook deze weer op E015 en E016 poken. Nu ziet u dat we ook het gereserveerde byte nodig hebben.

Deze mogelijkheden kunt u ook vanuit onderstaande listing kiezen. Met deze variaties kunt u in programma's de tekst roteren, laten dansen (links/rechts), en mooie schermovergangen bewerkstelligen. Let er wel op dat nadat u de CLS heeft gebruikt, de hele karakterset verloren is gegaan en u deze d.m.v. het SCREEN-commando weer moet oproepen.

```
5 IFPEEK(&HFCAF)=0THENWIDTH40ELSEWIDTH32
10 KEYOFF:COLOR14,1:LOCATE4,0:PRINT"SCHUIV
ENDE CLS/ROTATIES":LOCATE4,1:PRINTSTRING$(
23,195)
20 RESTORE1000:FORI=&HE000TO&HE020:READA$:
POKEI,VAL("&H"+A$):NEXT
30 LOCATE2,4:PRINT"CLS naar rechts:":PRINT
"POKE&HE015,&HCB:POKE&HE016,&H3F":PRINT
40 PRINT"  CLS naar links :":PRINT"POKE&HE
015,&HCB:POKE&HE016,&H27":PRINT
50 PRINT"  Roteren karakters naar rechts:"
:PRINT"POKE&HE015,0:POKE&HE016,&H0F":PRINT
60 PRINT"  Roteren karakters naar links :"
:PRINT"POKE&HE015,0:POKE&HE016,&H07
70 LOCATE3,16:PRINT"Om routine aan te roep
en:":LOCATE4,17:PRINT"DEFUSR=&HE000:U=USR(
0)":PRINT"DE ROUTINE (33 Bytes) IS":PRINT"
RELOCATABLE":PRINT"NA AANROEP SCREEN0/1 OM
 DE":PRINT"KARAKTERSET TERUG TE KRIJGEN.":
LOCATE0,0
1000 DATA 11,00,08,3A,AF,FC,FE,00,20,05,21
,00,08,18,03,21,00,00,CD,4A,00,00,0F,CD,4D
,00,23,1B,7A,B3,20,F2,C9
```

CLS-ROTR BAS D2

*NIEUWE KARAKTERSETS

Bent u uitgekeken op de standaard MSX karakterset? Of wilt u naast de oude set nieuwe lettertypes gebruiken? En u heeft geen zin de hele set karakter voor karakter te veranderen (wat ook erg veel dataruimte vraagt)? Dan staan er nog genoeg mogelijkheden open!
We kunnen namelijk de hele karakterset veranderen door simpelweg de oude waardes uit het VRAM te lezen, en na een modifikatie weer terug te VPOKEn. Dat betekent weliswaar dat de nieuwe set direkt van de oude afstamt, maar toch zijn er zo verrassend nieuwe letters te verkrijgen. Daarbij moet ik wel twee kanttekeningen plaatsen:

BEPERKINGEN

Ten eerste worden sommige nieuwe karaktersets breder gemaakt als de oude, waardoor het resultaat op screen 0, met de 6 pixels smalle karakters, niet altijd even fraai is. Op screen 1 zijn er evenwel geen problemen hiermee.
Ten tweede moeten we de nieuwe karakters wel ergens kwijt. Hiervoor zijn twee mogelijkheden:
- We veranderen de hele karakterset in een nieuw lettertype, met als beperking dat we de oude niet meer beschikbaar hebben.
- We houden het meest gebruikte deel van de karakterset (letters, cijfers en leestekens) normaal, en plaatsen de nieuwe karakterset hoger in het VRAM, over de grafische en internationale tekens heen.

LOCATIE

Ik prefereer zelf de laatste methode en die zal ik dan ook aanhouden. Welke karakters zijn nu de 'meest gebruikte'? Hiertoe beschouw ik het - voor de meeste computers gestandaardiseerde- deel dat loopt van CHR$(32) t/m CHR$(127). Dit zal ik dan ook behouden. Het is dan logisch om de nieuwe set te laten lopen van CHR$(128) t/m CHR$(223). Maar u kunt natuurlijk altijd andere delen van de karakterset hiervoor gebruiken. Als u bijvoorbeeld alleen hoofdletters en enkele leestekens wenst te gebruiken kunt u zeker 6 sets tegelijk gebruiken!
Onderstaand programma gaat er echter van uit dat er 1 nieuwe karakterset is van 128 t/m 243. Als u de routines, beginnend met #B hiermee MERGEd, kunt u het resultaat direkt bewonderen. Natuurlijk is ook gebruik in eigen programmatuur een fluitje van een cent.

MSX-VERSIE

Regel 10 hiervan verdient nog wel even uw aandacht. Hier wordt eerst d.m.v. de bekende locatie FCAFH gekeken op welk scherm we werken. Echter, om ook de SVI X-PRESS en MSX II gebruikers (waar ikzelf nu ook een van ben) van dienst te zijn, volgt er, indien we op SCREEN 0 werken, nog een andere kontrole. Adres 2DH bevat namelijk als inhoud de (MSX-VERSIE)-1, m.a.w. 0 als u op een MSX-I werkt en 1 als u op een MSX-II werkt. Met behulp van deze kontrole wordt bepaald of we de 40-koloms mode (MSXI) of de 80-koloms mode (MSXII) moeten inschakelen.

KAR-TEST BAS D2

```
5 '*******************************
6 '* KARAKTERSET-TEST - Voor het *
7 '*  aanroepen van de routines  *
8 '*      beginnend met #B       *
9 '*******************************
10 IFPEEK(&HFACF)=0THENIFPEEK(&H2D)=0THENW
IDTH40:ELSEWIDTH80:VDP(14)=0:ELSEWIDTH32
20 CLS:FORI=0TO31:PRINTCHR$(1)CHR$(I+64);"
 ";:NEXT:FORI=32TO255:PRINTCHR$(I);" ";:NE
XT:GOSUB1000:PRINT:PRINT:PRINT"DRUK EEN TO
ETS"
25 IFINKEY$=""THEN25
30 CLS:PRINT"Voer een tekst in: (MAX. 80 K
ARKATERS)":LOCATE0,4
35 A$=INKEY$:IFA$=""THEN35ELSEA=ASC(A$):IF
A<32ORA>127THENIFA=13THEN37ELSE35ELSEPRINT
CHR$(A);:B$=B$+CHR$(A):GOTO35
37 B$=B$+CHR$(A+96):IFB$=""ORLEN(B$)>128TH
EN30
40 IFLEN(A$)>80ORA$=""THEN30
50 A$="":FORI=1TOLEN(B$):B=ASC(MID$(B$,I,1
)):A$=A$+CHR$(B+96):NEXT:LOCATE0,8:PRINTA$
:A$="":B$="":GOTO25
```

CURSIEF

..Is het eerste voorbeeld dat ik geef. De eerste routine hiervoor is zowel op screen 0 als screen 1 goed bruikbaar.
Hoe maken we een karakter cursief? In feite hoeft alleen het bovenste deel van het karakter maar een beeldpunt naar links of rechts verschoven te worden ten opzichte van het onderste deel. Nu hadden we al gezien hoe een en ander vanuit machinetaal ging met de schuif- en roteerinstrukties, maar iets dergelijks kan in BASIC ook! Bij de behandeling van de bit-opbouw van een karakter in aflevering 2 (nr. 17) kon u middels figuur 2 al zien dat ieder bit de halve waarde vertegenwoordigt van z'n linker buurbit. Als we nu de te schuiven lijn gewoon door 2 delen, krijgt het getal de halve waarde en ieder bit dus ook. Een eventuele rest (het uittredende bit met waarde 2^-1 oftwel 1/2) gaat automatisch verloren bij het inVPOKEn van de nieuwe waarde.
In #B.000 wordt in de FOR-NEXT lus het cursief te maken gebied gedefinieerd als CHR$(32) t/m CHR$(127). De genestelde lus FOR J=0 TO 3 zorgt ervoor dat eerst de bovenste drie lijnen een pixel naar rechts verschoven worden door te delen door twee. Deze moeten 96 karakters verder ingeVPOKEd worden, vandaar dat na de VPOKE +768 (96*8=768) staat. Vervolgens moeten op dezelfde wijze de onderste vier lijnen, nu echter onveranderd, overgezet worden.

```
1000 FOR I=BASE(2)+32*8 TO BASE(2)+127*8 S
TEP 8:FOR J=0 TO 3:VPOKE I+J+768,VPEEK (I+
J)/2:NEXT:FOR J=4 TO 7:VPOKE I+J+768,VPEEK
(I+J):NEXT:NEXT:RETURN
```

Een aardige variatie hierop is, om de lijnen 1-4 i.p.v. 0-3 te schuiven; het resultaat is echter wel minder makkelijk leesbaar.

```
1000 FOR I=BASE(2)+32*8 TO BASE(2)+127*8 S
TEP 8:VPOKE I+768,VPEEK(I):FOR J=1 TO 4:VP
OKE I+J+768,VPEEK (I+J)/2:NEXT:FOR J=5 TO
7:VPOKE I+J+768,VPEEK(I+J):NEXT:NEXT:RETUR
N
```

Cursieve letters naar links zijn, hoewel minder gebruikelijk, even makkelijk te krijgen. In plaats van de bovenste vier lijn, verschuiven we nu de onderste vier (4-7) met het volgende resultaat:

```
1000 FOR I=BASE(2)+32*8 TO BASE(2)+127*8 S
TEP 8:FOR J=0 TO 3:VPOKE I+J+768,VPEEK (I+
J):NEXT:FOR J=4 TO 7:VPOKE I+J+768,VPEEK(I
+J)/2:NEXT:NEXT:RETURN
```

In de volgende aflevering komen nog veel meer mogelijkheden aan bod. Ook de logische bewerkingen van de vorige aflevering zullen hierbij een rol spelen.

M.Hondema

NIEUWE KARAKTERSETS

Ziedaar, nu eindelijk eens een nieuwe titel, na het hele scrolling-gebeuren. En inderdaad zal deze aflevering geheel zijn toegewijd aan deze 'techniek', die ook u zeer goede diensten zal kunnen bewijzen. En als u zelf kreatief bezig wilt zijn, staat niets u in de weg om eigen hersenprodukten het levenslicht te doen aanschouwen. Mocht u zo snel niets kunnen bedenken, dan hoop ik dat de voorbeelden in dit deel enige inspiratie kunnen geven; het is echt de moeite waard te blijven proberen...

ERRORTJE

Maar allereerst nog een klein erratum.

- Ten eerste waren de eerste scrolling-delen van het artikel geschreven op de 40-koloms screen 0 mode (lees: op een MSX 1) waardoor de bepaling van het beginadres van het gebruikte karakter 219 in het VRAM (die 'S=3800' luidde) niet klopte op 80 koloms. Universeler bepaalt u dit met 'S=BASE(2+(PEEK(&HFCAF)*5))+219*8'; dit zal verderop duidelijker gemaakt worden. In elk geval werkt het dan ook op de professionele 80 koloms-mode.
- Verder was het mini-programma KARAKTERSET-TEST uit het vorige nummer overduidelijk niet voldoende getest. Ten eerste stond in regel 10 'PEEK(&HFACF)' i.p.v. 'PEEK(&HFCAF)', maar misschien heeft u dit zelf kunnen korrigeren. Verder kwamen er STRING-problemen. De nieuwe versie werkt door de ingetikte tekst binnen het VRAM te kopieren, zodat er geen ellende met de string-manipulaties kan ontstaan. Ook zult u nu geen last meer hebben met het m-etje aan het einde van de nieuwe zin (overbodigheid in regel 37). Daar komt 'ie:

KARTEST BAS D2

```
 5 '*******************************
 6 '* KARAKTERSET-TEST - Voor het *
 7 '*  aanroepen van de routines  *
 8 '*       beginnend met #B      *
 9 '*******************************
10 IFPEEK(&HFCAF)=0THENIFPEEK(&H2D)=0T
HENWD=40:BA=0:WIDTH40ELSEWD=80:BA=0:WI
DTH80ELSEWD=32:BA=5:WIDTH32
20 CLS:FORI=0TO31:PRINTCHR$(1)CHR$(I+6
4);" ";:NEXT:FORI=32TO255:PRINTCHR$(I)
;" ";:NEXT:GOSUB1000:PRINT:PRINT:PRINT
"DRUK EEN TOETS"
25 IFINKEY$=""THEN25
30 T=0:I=0:CLS:PRINT"Voer een tekst in
: (MAX. 80 KARAKTERS)":LOCATE0,4
40 I$=INKEY$:IFI$=""THEN40ELSET=T+1:GO
SUB60:I=ASC(I$):IFI=13THEN45ELSEIFI<32
ORI>127THEN40
42 PRINTCHR$(I);:GOTO40
45 FORI=BASE(BA)+4*WDTOBASE(BA)+4*WD+8
0:IFVPEEK(I)<>32THENVPOKEI+4*WD,VPEEK(
I)+96:NEXTELSENEXT
50 GOTO25
60 IFT<80THENRETURNELSERETURN45
1000 'ROUTINE #B-00K1 : MERGE REGEL 1-
7 MET VOORGAANDE; CHR$(128)-CHR$(224)
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+127*8 STEP 8:FOR J=0
 TO 6:VPOKE I+J+769,VPEEK(I+J) OR VPEE
K(I+J+1):VPOKE I+768,VPEEK(I):NEXT:NEX
T:RETURN
```

- Verder moet ik een opmerking uit het vorige nummer herstellen. In het vorige nummer kon u lezen dat ik me zou beperken tot het plaatsen van de nieuwe karakterset boven de oude. Wel, de routines daarvoor kunt u blijven verwachten, maar er is toch een irritant nadeel van deze methode en dat is de snelheid. Nu moet namelijk de hele karakterset met aanpassingen voor het nieuwe lettertype hoger in het VRAM in z'n geheel 'overgeschreven' worden, dus 8 lijnen per karakter. Door echter de nieuwe set bovenop de oude te zetten hoeven alleen de veranderingen maar ingeVPOKEd te worden, en als dat er niet veel zijn werkt dit sneller (dus niet in alle gevallen!). Verder kunt u het resultaat onmiddellijk zien in de normale karkaterset, wat natuurlijk zeker in de eksperimenteerfase erg handig is.

TWEE KEER

...zal ik elke routine geven, en wel zoals boven genoemd. Dus: een karakterset zoals de cursieve uit de vorige aflevering, lopend van CHR$(128) t/m CHR$(224), welke u met de KARAKTERSET-TEST kunt bekijken waarbij de normale ASCII-karakters niet aangetast worden; en een die ik van CHR$(32) t/m CHR$(191 geplaatst heb, zodat nu ook de accent-letters e.d. veranderd worden. Maar het te veranderen gedeelte kunt u zelf bepalen in de I-FOR-NEXT lus (dit woord zal wel niet in de VAN DALE staan...!), alsmede de doeladressen in de VPOKE-commando's (hiervoor gebruik ik '+768' om 96 karakters verder op weg te schrijven).

Onderstaande routines hebben nog een extraatje. Dit ziet u in regel 1010. U bent nu langzamerhand doodgegooid met adres FCAFH en dat adres kan zich ook nu er weer niet buitenhouden. Voor SCREEN 0 staat het begin van het karaktersetgeheugen namelijk ALTIJD in BASE(2) (dit is in de 80-kolomsmode hoger gelegen als in de 40-kolomsmode), maar voor SCREEN 1 in BASE(7). Dus met BASE((2+PEEK(&HFCAF)*5) vindt u dit voor elk scherm. Zorgt u er verder wel voor de in de routine gebruikte variabelen tot in integers om te dopen met het DEFINT-commando; dat scheelt weer een grove 10% snelheid!

NOGMAALS CURSIEF

In de vorige aflevering zijn cursief-routines voor CHR$(127) t/m CHR$(224) gegeven; daarvoor hier eerst even de equivalenten voor de laatstgenoemde mogelijkheid (karakter 32 t/m 191). U zult merken dat deze ongeveer twee keer zo snel werken omdat er nu maar 4 lijnen veranderd hoeven te worden.

```
1000 'ROUTINE #B-00A2 : CURSIEF RECHTS
 1; VERANDERT ASCII KARAKTERSET
1010 FOR I=BASE(2)+32*8 TO BASE(2)+191
*8 STEP 8:FOR J=0 TO 3:VPOKE I+J,VPEEK
(I+J)/2:NEXT:NEXT:RETURN


1000 'ROUTINE #B-00B2 : CURSIEF RECHTS
 2; VERANDERT NORMALE KARAKTERSET
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+191*8 STEP 8:FOR J=1
 TO 4:VPOKE I+J,VPEEK (I+J)/2:NEXT:NEX
T:RETURN

1000 'ROUTINE #B-00C2 : CURSIEF LINKS;
 VERANDERT NORMALE KARAKTERSET
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+191*8 STEP 8:FOR J=4
 TO 7:VPOKE I+J,VPEEK (I+J)/2:NEXT:NEX
T:RETURN
```

ONDERSTREPEN

Dit is een uiterst eenvoudige routine. Onderstrepen, immers, kan simpelweg door lijn 7 geheel te setten (waarde 255) op het geselecteerde deel van de karakterset.

```
1000 'ROUTINE #B-00D1 : ONDERSTREPEN C
HR$(128)-CHR$(224)
1010 FOR I=BASE(2)+32*8 TO BASE(2)+127
*8 STEP 8:FOR J=0 TO 7:VPOKE I+J+768,V
PEEK(I+J):NEXT:VPOKE I+7+768,255:NEXT:
RETURN
1000 'ROUTINE #B-00D2 : ONDERSTREPEN C
HR$(32)-CHR$(127)
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
33*8 TO BASE(2+A)+127*8 STEP 8:VPOKE I
+J+7,255:NEXT:RETURN
```

VET / FUTURISTISCH

Schijnbaar is er weinig verband tussen deze twee lettertypes, maar toch is de laatste (in elk geval in onderstaande versie) van het bekende vette lettertje afgeleid. Om een karakter vet te maken, wordt een projectie van het karakter, bestaande uit dit karakter maar dan 1 positie naar rechts verschoven, over het karakter geplaatst. Hoe u een lijn van een karakter 1 positie naar rechts verplaatst, werd in het vorige deel al uitgelegd: de lijnwaarde delen door 2. Dit gebeurde toen om een kursief lettertje te krijgen. Nu moeten we dit beeld nog op het orgineel zetten. Wel, hiervoor leent de OR-funktie zich: op iedere kolom waar of in het orgineel, of in het beeld een punt geset is, zal dit ook in de projektie geset zijn. Dus: LIJNWAARDE OR LIJNWAARDE/2, m.a.w. VPEEK (I+J) OR VPEEK(I+J)/2. Als u deze berekening voor alle lijnen van de karakters uitvoert, krijgt u een vette letter, die trouwens zelfs op SCREEN 0 nog wel vrij goed leesbaar is. En... meerdere aanroepen van deze routine na elkaar geeft een steeds bredere letter! Natuurlijk wordt dit wel op SCREEN 0 al gauw onleesbaar. De listing geef ik hier echter niet, omdat u die gemakkelijk van onderstaande listing (futuristisch) kunt afleiden: i.p.v. 'FOR J=4 TO 7' hoeft u alleen maar 'FOR J=0 TO 7' te plaatsen. Nu zal het u ook duidelijk zijn hoe 'FUTURISTISCH' werkt: nu worden alleen de onderste 4 lijnen van het karakter 'vetgemest'. Hierdoor krijgen de lettertekens 'dikke pootjes'.

```
1000 'Routine #B-00F1 : VET REGEL 4-7
(FUTURISTIC);CHR$(128)-CHR$(224)
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+127*8 STEP 8:FOR J=0
 TO 3:VPOKE I+J+768,VPEEK(I+J):NEXT:FO
R J=4 TO 7:VPOKE I+J+768,VPEEK(I+J) OR
 VPEEK(I+J)/2:NEXT:NEXT:RETURN
1000 'ROUTINE #B-00F2 : VET LIJN 4-7 (
FUTURISTIC); CHR$(32)-CHR$(191)
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+191*8 STEP 8:FOR J=4
 TO 7:VPOKE I+J,VPEEK (I+J) OR VPEEK(I
+J)/2:NEXT:NEXT:RETURN
```

VET LINKS

Dit is weer eens een hele andere karakterset; u krijgt zo wat vierkanter en vette letters. Het principe is echter, zoals zo vaak, erg eenvoudig: overal waar het karakter dicht bij de linkerkant van de matrix komt, wordt het nog wat dikker gemaakt. U ziet VPOKE X,Y OR &B11000000 staan. Zoals u begrijpt, zorgt dit ervoor dat kolom 7 en kolom 6 geset worden; dit geeft een dikkere lijn (twee pixels breed).
Maar als we dit met alle lijnen van alle karakters zouden doen, zou dat alleen een vertikale streep links op het karakter opleveren. Daarom staat de vergelijking 'IF X= 64' er. Als namelijk of kolom 6 of kolom 7 geset is, is de lijnwaarde

in elk geval 64 (dit als alleen kolom 6 geset is, als er meer bits geset zijn kan de waarde alleen maar hoger worden). Nu gaan we dus kolom 6 en 7 setten als aan bovenstaande voorwaarde voldaan wordt. Is dat niet het geval, dan wordt de lijnwaarde niet veranderd.

```
1000 'ROUTINE #B-00G1 : VET KOLOM 7; C
HR$(128)-CHR$(224)
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+127*8:IF VPEEK(I)=>6
4 THEN VPOKE I+768,VPEEK(I) OR &B11000
000:NEXT:RETURN ELSE VPOKE I+768,VPEEK
(I):NEXT:RETURN
1000 'ROUTINE #B-00G2 : VET KOLOM 7; C
HR$(32)-CHR$(127)
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+191*8:IF VPEEK(I)=>6
4 THEN VPOKE I,VPEEK(I) OR &B11000000:
NEXT:RETURN ELSE NEXT:RETURN
```

Een variant hierop is de volgende routine. D.m.v. de vergelijking 'IF X=32' vindt nu een verandering van de lijnwaarde plaats als het bit op kolom 5 (32), 6 (64) of 7 (128) geset is en worden middels de OR-bewerking nu kolom 5 en 6 geset. Dit veroorzaakt een erg vette letter met een duidelijk accent op de linkerkant. Het zal echter geen betoog behoeven dat dit geheel in uw eksperimenterende hand ligt...

```
1000 'ROUTINE #B-00H1 : VET KOLOM 5&6;
 CHR$(128)-CHR$(224)
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+127*8:IF VPEEK(I)=>3
2 THEN VPOKE I+768,VPEEK(I) OR &B01100
000:NEXT:RETURN ELSE VPOKE I+768,VPEEK
(I):NEXT:RETURN
1000 'ROUTINE #B-00H2 : VET KOLOM 5&6;
 CHR$(32)-CHR$(191)
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+191*8:IF VPEEK(I)=>3
2 THEN VPOKE I,VPEEK(I) OR &B01100000:
NEXT:RETURN ELSE NEXT:RETURN
```

VERTIKAAL MERGEn

Hoe u twee lijnwaardes moet MERGEn door ze te OR-ren kon u in de vorige routine al zien, toen ging het echter met 1 variabele en 1 konstante waarde. Hier gebruiken we twee variabelen: lijn 0 en lijn 1 van het karakter mengen we op lijn 1, zoals regel 1000 vermeldt. Dus ook nu geldt dat overal waar of op lijn 0, of op lijn 1 een bitje geset is, dit ook op lijn 1 geset zal zijn. Dit gaat met 'VPOKE X+1,VPEEK (X) OR VPEEK (X+1)', en natuurlijk moeten we hier voor de CHR$(128-224)-versie nog 768+1=769 bij de te VPOKEn waarde op tellen, en eerst met de genestelde 'FOR J=0 TO 7'-lus het karakter hoger in het VRAM in z'n geheel kopieren (ja... dat kost altijd kostbare computertijd!)

```
1000 'ROUTINE #B-00I1 : MERGE REGEL 0
EN 1 OP REGEL 1; CHR$(128)-CHR$(224)
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+127*8 STEP 8:FOR J=0
 TO 7:VPOKE I+J+768,VPEEK(I+J):NEXT:VP
OKE I+769,VPEEK(I) OR VPEEK(I+1):NEXT:
RETURN
1000 'ROUTINE #B-00I2 : MERGE REGEL 0
EN 1 OP REGEL 1; CHR$(32)-CHR$(191)
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+191*8 STEP 8:VPOKE I
+1,VPEEK(I) OR VPEEK(I+1):NEXT:RETURN
```

Dit is leuk voor de hoofdletters en sommige leestekens, maar omdat lijn 0 en 1 bij de kleine letters niet veel gebruikt worden (waarvoor zouden ze anders nog klein zijn?), volgt hier nog een uitvoering die naast 0 en 1, ook lijn 5 en 6 mengt op lijn 5. Nu is ieder karakter zichtbaar veranderd. Overigens zijn deze karaktersets natuurlijk ook goed bruikbaar op screen 0 en (nog meer vanzelfsprekend) in de 40 als zo ook de 80 koloms mode.

```
1000 'ROUTINE #B-00J1 : MERGE REGEL 0+
1->1 & 5+6->5; CHR$(128)-CHR$(224)
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+127*8 STEP 8:FOR J=0
 TO7:VPOKE I+J+768,VPEEK(I+J):NEXT:VPO
KE I+769,VPEEK(I) OR VPEEK(I+1):VPOKE
I+773,VPEEK (I+5) OR VPEEK (I+6):NEXT:
RETURN
1000 'ROUTINE #B-00J2 : MERGE REGEL 0+
1->1 & 5+6->5; CHR$(32)-CHR$(191)
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+191*8 STEP 8:VPOKE I
+1,VPEEK(I) OR VPEEK(I+1):VPOKE I+5,VP
EEK (I+5) OR VPEEK (I+6):NEXT:RETURN
```

We kunnen ook alle lijnen van een karakter mergen met de buurlijnen. Echter, als we dit van boven naar beneden doen, gebeurt het volgende:
1e: LIJN 1=LIJN 0 OR LIJN 1
2e: LIJN 2=LIJN 1 OR LIJN 2 = LIJN 0 OR LIJN 1 OR LIJN 2
3e: LIJN 3=LIJN 3 OR LIJN 2 = LIJN 0 OR LIJN 1 OR LIJN 2 OR LIJN 3
etc.

Dat is dus geen ideale methode; het karakter zal naar onderen toe steeds vetter en onleesbaarder worden. Maar als we nu LIJN 0 en LIJN 1 op lijn 0 combineren, zal lijn 1 onveranderd blijven en kan die weer met LIJN 2 op lijn 1 geset worden enz. Zo krijgen we een vertikaal vet karakter dat natuur-

lijk ook een lijn hoger is als de 'moeder'. Deze dochter is helaas wel wat minder goed leesbaar.

```
1000 'ROUTINE #B-00K1 : MERGE REGEL 1-
7 MET VOORGAANDE; CHR$(128)-CHR$(224)
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+127*8 STEP 8:FOR J=0
 TO 6:VPOKE I+J+769,VPEEK(I+J) OR VPEE
K(I+J+1):VPOKE I+768,VPEEK(I):NEXT:NEX
T:RETURN
```

Andersom kunnen we ook werken; #B-00K2 gaat de lijnen van het karakter volgens bovenstaand principe langs maar nu van onderen naar boven toe.

```
1000 'ROUTINE #B-00K2 : MERGE REGEL 6-
0 MET VOLGENDE; CHR$(32)-CHR$(191)
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+191*8 STEP 8:FOR J=6
 TO 0 STEP-1:VPOKE I+J+1,VPEEK(I+J+1)
OR VPEEK(I+J):NEXT:NEXT:RETURN
```

NOGMAALS LINKS VET

Dit is niet meer dan een variant op de lettertypes G en H. Ook hier worden kolom 7 en 6 geset als een van beide al geset was, maar nu alleen op de onderste vier lijnen. En ziedaar: SCIENCE FICTION (althans zo heb ik hem zelf gedoopt; het lijkt nogal op het lettertype van LEVEL 9's RETURN TO EDEN).

```
1000 'Routine #B-00L1 : VET REGEL 4-7
KOLOM 5-7(SCIENCE F.);CHR$(128)-CHR$(2
24)
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+128*8 STEP 8:FOR J=0
 TO 3:VPOKE I+J+768,VPEEK(I+J):NEXT:FO
R J=4 TO 7:IF VPEEK(I+J)>=32 THEN VPOK
E I+J+768,VPEEK(I+J) OR &B11000000:NEX
T:NEXT:RETURN ELSE VPOKE I+J+768,VPEEK
(I+J):NEXT:NEXT:RETURN
1000 'Routine #B-00L2 : VET REGEL 4-7
KOLOM 5-7(SCIENCE F.);CHR$(32)-CHR$(19
1)
1010 A=PEEK(&HFCAF)*5:FOR I=BASE(2+A)+
32*8 TO BASE(2+A)+191*8 STEP 8:FOR J=4
 TO 7:IF VPEEK(I+J)>=32 THEN VPOKE I+J
,VPEEK(I+J) OR &B11000000:NEXT:NEXT:RE
TURN ELSE VPOKE I+J,VPEEK(I+J):NEXT:NE
XT:RETURN
```

KANTELEN

Wie de volgende routine(s) op het allereerste gezicht doorziet is zeker een vrij doorgewinterd programmeur... Maar met enige uitleg zal het iedereen helder worden. Dit is echter niet de makkelijkste routine! Om een karakter te kantelen, moeten alle vertikale kolommen horizontaal getekend worden, dat spreekt. Dus: kolom 0 wordt lijn 0, kolom 1 lijn 1 etc. Dat lijkt eenvoudig, maar er komt toch wel het een en ander bij kijken. Om de routine overzichtelijk te houden, krijgt u nu geen one-liner. In regel 1010 de bekende FOR-NEXT voor de te veranderen karakters, alsmede een reservering voor variabele A(n) voor 8 lijnwaarden. In A(0) t/m A(7) komen de vertikale kolomwaardes te staan. Nadat we een kolom hebben uitgelezen, kan deze namelijk niet direkt horizontaal ingeVPOKEd worden, omdat dat de andere kolommen weer zou beinvloeden. Daarom worden deze waarden eerst in A(X) opgeslagen voordat het karakter in een keer veranderd wordt wanneer alle kolommen (lijn voor lijn) zijn uitgelezen. Op regel 1020 ziet u twee FOR-NEXT lussen, namelijk een J-lus voor de 7 te setten lijnen, en een daarin genestelde K-lus om voor iedere lijn alle 7 kolommen uit te lezen. Deze kolommen hebben, zoals u weet, de waardes 2 ^ kolomnr.; dat is in dit geval 2 ^ J. De teller J houdt namelijk bij welk bit op elke lijn uitgelezen wordt. Maar dat uitlezen gaat niet direkt.

CHECK BIT

Om van een getal na te gaan of een bepaald bit geset is, is het het gemakkelijkste alle andere bits af te schermen. Afschermen kan met de AND funktie, dat was u al eerder tegengekomen: door een bit te ANDen met 0 is het resultaat ALTIJD 0. Dus nu ANDen we het te onderzoeken bit met 1 en alle andere met 0. Het resultaat is dan dat alleen het te onderzoeken bit in het resultaat komt. In #B-00M: LIJNWAARDE AND 2 ^ J. (Een macht van twee is altijd 1 bit in een binair getal).

Bv: Onderzoek bit 3 (dus AND 00000100)

```
    01001101              10110010
AND 00000100          AND 00000100
    --------              --------
    00000100 = 4 = 2^3 dus BIT3 geset
                          00000000 = 0 dus
                           BIT 3 niet geset
```

Het resultaat is echter variabel zoals u ziet : als het oorspronkelijke (en dus ook het resultante) bit 1 was wordt het resultaat niet 1, maar 2 ^ BITNR.; oftewel 2 ^ J. In 1030 ziet u dit 'checken'. Alleen moeten de gechekte bits (die samen de nieuwe horizontale lijnwaarde gaan vormen) nog worden

opgeslagen in A(J). Dit kan door voor een bit dat geset blijkt te zijn, zijn waarde (2^J) bij de lijnwaarde op te tellen, en anders de lijnwaarde gelijk te houden (dit gaat vanzelf als aan de voorwaarde op regel 1030 niet voldaan wordt). Als deze lussen afgewerkt zijn, staan alle vertikale kolommen van onder elkaar liggende bits (afgewerkt door de K-lus) als horizontale lijnwaardes in A(0) t/m A(7). Die hoeven alleen nog maar ingeVPOKEd te worden, zie regel 1040. Dit gebeurt in #M-00B2 dan als extra nog omgekeerd (A(7);A(6)...A(0)) zodat de letters zowel gekanteld als gespiegeld worden. Dan moeten de A-variabelen nog even leeggemaakt worden voor het volgende karakter behandeld wordt, anders zou A steeds hoger worden door het optellen van de bitwaardes van elk karakter.

MAAR...

...de routine is wel zeer langzaam! De 96 karakters van #B-00M1 kosten 87 seconden en de 159 van #B-00M2 zelfs 148 seconden om op hun zij gezet te worden. De kantelsnelheid hangt overigens af van het aantal gesette (1) bits; hoe meer voorgrondkleur een karakter heeft hoe langzamer het kantelen. Verder is het resultaat op screen 0 wel verre van indrukwekkend omdat de 6 kolommen doorgaans te weinig zijn voor een gekantelde (hoofd-) letter. Maar ook hier is SCREEN 1 de oplossing.

```
1000 'Routine #B-00M1 : LETTERS GEKANT
ELD; CHR$(128)-CHR$(224)
1010 DEFINT A-Z:DIM A(8):A=PEEK(&HFCAF
)*5:FOR I=BASE(2+A)+32*8 TO BASE(2+A)+
127*8 STEP 8
1020 FOR J=7 TO 0 STEP -1:FOR K=0 TO 7
1030 IF (VPEEK(I+K) AND 2^J)=2^J THEN
A(J)=A(J)+2^K
1040 NEXT:NEXT:FOR J=0 TO 7:VPOKE I+J+
768,A(J):NEXT:FORJ=0TO7:A(J)=0:NEXT:NE
XT:RETURN
1000 'Routine #B-00M2 : LETTERS GEKANT
ELD LINKS; CHR$(32)-CHR$(191)
1010 DEFINT A-Z:DIM A(8):A=PEEK(&HFCAF
)*5:FOR I=BASE(2+A)+32*8 TO BASE(2+A)+
191*8 STEP 8
1020 FOR J=7 TO 0 STEP -1:FOR K=0 TO 7
1030 IF (VPEEK(I+K) AND 2^J)=2^J THEN
A(J)=A(J)+2^K
1040 NEXT:NEXT:FOR J=7 TO 0 STEP -1:VP
OKE I+J,A(J):NEXT:FORJ=0TO7:A(J)=0:NEX
T:NEXT:RETURN
```

BESLUIT

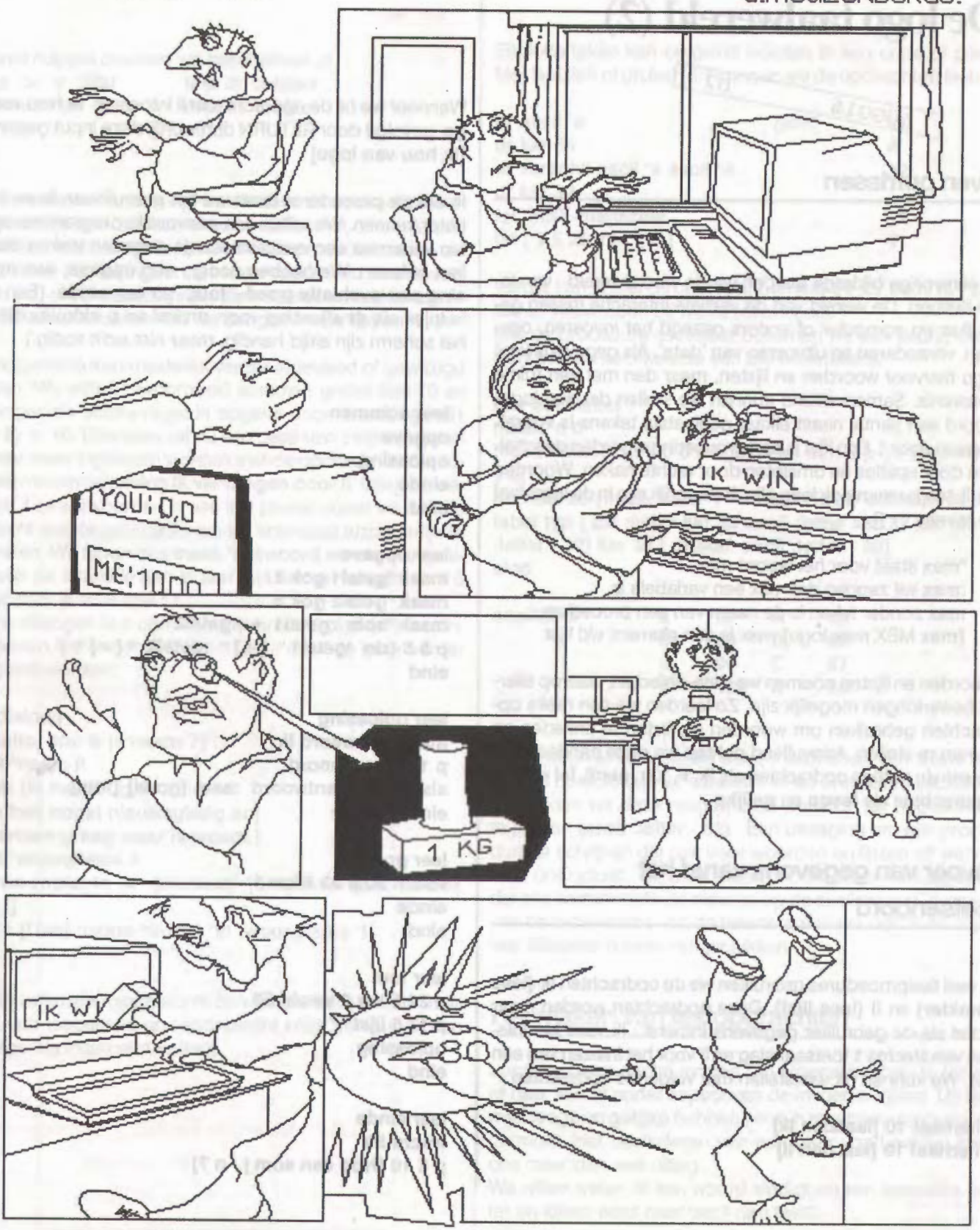
Hiermee besluit ik dit artikel dan maar; anders wordt het wel een erg lange lap om te doorworstelen. Nogmaals wil ik u aanraden zelf ook eens een leuk lettertje te maken op bovenstaande wijze (dus geen ellenlange data-lijsten...) en dat in te sturen. Zelf heb ik er ook al weer verscheidene klaarliggen. Als het aan mij ligt hoeft u nooit meer de afgezaagde MSX-karakterset te gebruiken! Maar nu eerst maar het laatste MSX-nieuws opdoen op de MSX DOE DAG in DEN HAAG en dan tot het volgend nummer...

M.Hondema

Z80 PCT D3

Z80 NEEMT WRAAK

J.MEULENBERGS.



De logo taalwereld (2)

LOGO16 D2

Even opfrissen

In een vorige bijdrage leerden we de woordwereld van logo kennen. De wereld van de verbale interactie tussen gebruiker en computer of anders gezegd het invoeren, opslaan, veranderen en uitvoeren van 'data'. Als gegevens kent logo hiervoor woorden en lijsten, maar dan met een ruime betekenis. Samenvattend kunnen we stellen dat een logowoord een aantal naast elkaar geplaatste tekens is voorafgegaan door ". Een lijst is een groepering woorden gescheiden door spaties en omsloten door rechte haken. Woorden en lijsten kunnen ook leeg zijn. Belangrijk zijn in de logo-taal de tekens " : [].

"**msx** staat voor het woord msx
:msx wil zeggen dat msx een variabele is
msx zonder teken is de naam van een procedure
[msx MSX msx-logo] msx is een element v/d lijst

Woorden en lijsten noemen we data-objecten, waarop allerlei bewerkingen mogelijk zijn. Zo leerden we een reeks opdrachten gebruiken om woorden en lijsten te ontleden en samen te stellen. Aanvullend richten we onze aandacht nu op enkele nuttige opdrachten als **lk, ll, kar, ascii, tel** en testopdrachten als **leegp** en **gelijkp** .

Invoer van gegevens vanaf het toetsenbord

In veel taalprocedures gebruiken we de opdrachten **lk (lees karakter)** en **ll (lees lijst)**. Deze opdrachten worden maar actief als de gebruiker gegevens intoetst : lk voor het inlezen van slechts 1 toetsaanslag en ll voor het inlezen van een lijst. We kunnen dit vaststellen met volgende opdrachten :

```
herhaal 10 [laatzien lk]
herhaal 10 [laatzien ll]
```

Wanneer we bij de opdracht met ll intoetsen **lk hou van logo** gevolgd door RETURN dan wordt deze input geprint als **[lk hou van logo]**

In enkele procedures leren we het gebruik van lk en ll nog beter kennen. We willen een eenvoudig programma opzetten waarmee een eersteklassertje sommen kleiner dan 10 kan oefenen. We hebben nodig : een **opgave**, een **oplossing** met **evaluatie goed / fout**, en een **einde**. (Een paar hulpjes als **dr** afkorting voor drukaf en **p** afdrukpositie op het scherm zijn altijd handig, maar niet echt nodig.)

```
leer sommen
opgave
oplossing
einde
eind

leer opgave
maak "getal1 gok 5
maak "getal2 gok 5
maak "som :getal1 + :getal2
p 5 5 (zin :getal1 " [+] " :getal2 " [=] " )
eind

leer oplossing
maak "antwoord lk
p 17 5 :antwoord
als gelijkp :antwoord :som [goed] [fout]
eind

leer goed
wacht 30 p 23 5 kar 1
einde
eind

leer fout
p 23 5 kar 2 wacht 50
p 17 5 lijst " "
oplossing
eind

leer einde
wacht 50
p 5 10 [Nog een som j / n ?]
```

```
maak "keuze lk
als  :keuze = "j [ss sommen] [p 5 12 [Dan stoppen we.]]
eind
```

de twee hulpjes drukken we naast elkaar af

```
leer p  :x  :y  :lijst          leer dr  :object
zetcursor lijst  :x  :y         drukaf  :object
typ  :lijst                     eind
eind
```

In de procedure oplossing wordt door de opdracht maak "antwoord **lk** aan :antwoord automatisch het teken van de ingedrukte toets toegekend. Ook in de procedure einde zal **lk** het teken van de ingedrukte toets toekennen aan :keuze. Alleen de j-toets start de procedure opnieuw. Het indrukken van alle andere toetsen zal het programma stoppen.

Dit programma kan makkelijk verder ingekleed of gewijzigd worden. We willen bijvoorbeeld sommen groter dan 10 en vervangen de eerste regel in opgave door **maak "getal1 (gok 5) + 10**. Elke som zal nu een getal van 2 cijfers zijn en kan niet meer ingelezen worden met de opdracht lk. We zullen dan ook in oplossing **lk vervangen door ll**. Daar we weten dat ll de antwoorden in een lijst plaatst zullen we ook de opdracht **eerste** gebruiken om elk antwoord afzonderlijk te evalueren. We typen dus **maak "antwoord eerste ll** en passen ook de x-positie aan in leer fout en vervangen p 17 5 kar 32 door **p 16 5 lijst " "**
Ook in dialogen is ll onmisbaar evenals drukaf (dr) en zin. We starten hieronder een dialoog die door de lezer verder kan gezet worden.

```
leer dialoog
dr [Hallo, hoe is je naam ?]
maak "naam ll
dr (zin [Ik heet Logo] ", :naam)
dr [Ik ben nogal nieuwsgierig en]
dr [vernam graag waar je woont.]
maak "woonplaats ll
als :woonplaats = [Herselt] [dr [Daar kom ik ook vaak.]]
dr (zin [Dus] :naam "woont "in :woonplaats "!)
eind
```

Aan de variabele :naam wordt de waarde van een in te toetsen naam toegekend en :woonplaats krijgt de waarde van de volgende ingetoetste naam.

De opdrachten ascii, kar en tel

Elk letterteken kan omgezet worden in een code of getal. Met laatzien of drukaf (dr) kunnen we de opdrachten testen.

```
dr ascii "a                    geeft   97
dr kar 65                              A
dr verschil ascii "a ascii "A          32
dr kar 32
tel "MSX-magazine                      12
tel [ 3 5 28 234 9 0 ]                 6
```

Verder kan je met de opdracht **herhaal 20 [dr ascii lk]** zelf testen welke code bij welke toets hoort.
Met de procedure asciitabel bekomen we een keurig overzicht van de asciiwaarde van alle lettertekens

```
leer asciitabel
maak "n 0
maak "alfabet [a b c d e f g h i j k l m n o p q r s t u v w x y z]
herhaal 26 [maak "n :n + 1 maak "letter element :n :alfabet typ ( zin :letter kar 32 ascii :letter kar 32 kar (ascii :letter - 32) kar 32 ) drukaf  ascii :letter - 32]
eind
```

```
asciitabel      a   97   A   65
                b   98   B   66
                c   99   C   67
                d  100   D   68
                ...
```

We hebben al vastgesteld dat de ascii-code van kleine letters en hoofdletters 32 verschilt. In de procedure asciitabel gebruikten we deze mogelijkheid om toch de hoofdletters te printen **(ascii :letter - 32)**. Een uitdaging om een procedure te schrijven die ook voor woorden en lijsten dit werkje voor ons oplost. Iedereen heeft ondertussen wel begrepen dat alle sorteerroutines steunen op de ascii-code. Op grond van de asciiwaarde van de tekens zullen we ook in de logotaal alfabetisch leren rangschikken.

Het maken van testopdrachten

Logo kent een aantal termen om eigenschappen te testen of naar een of ander aspect van de invoer te kijken. De termen **leegp** en **gelijkp** hebben we al in meerdere procedures ontmoet. Het bestuderen van een paar voorbeelden leert ons meer dan veel uitleg.
We willen weten of een woord eindigt op een bepaalde letter en kijken eerst naar test1 dan test2.

```
leer test1 :el :object
als gelijkp :el laatste :object [gt "waar]   [gt "vals]
eind

dr test1 "d "kind          geeft terug          waar
dr test1 "d "kin                                vals

leer test2 :el :object
als gelijkp :el laatste :object [gt "waar]
[gt waar? :el :object]
eind
```

In beide procedures kijkt gelijkp of twee elementen gelijk zijn. Is dat zo dan wordt altijd de eerste lijst uitgevoerd. Is dat niet zo dan wordt de tweede lijst uitgevoerd. Let op het verschil tussen de tweede lijst van test1 en test2. In test2 is **waar?** een procedure, die eerst gedefinieerd moet worden voor we test2 kunnen testen.

```
leer waar? :el :object
als leegp :object [gt 0]
als gelijkp :el laatste :object [gt tel :object]
gt waar? :el ml :object
eind

dr test2 "d "kind          geeft          waar
dr test2 "d "kin                          0
dr test2 "d "kinderen                     4
```

In waar? vinden we ook de term **leegp**. Als het woord niet eindigt op d kijkt deze fuctie waar er een d gebruikt is en daar de procedure recursief is en telkens de laatste letter met ml wegvalt zal de procedure maar stoppen als het woord 'leeg' is.
Andere termen als **woordp, lijsp** en **getalp** hebben respectievelijk als functie na te gaan of iets gelijk is aan een woord, een lijst of een getal.

Een raadspelletje om af te sluiten

Het klassiek raadspelletje om in zo gering mogelijk aantal beurten een getal onder 100 te raden staat veelal als model voor een ander spel of een quiz.

```
leer raadspel
begin
kern
slot
eind

leer begin
maak "score 0
maak "getal gok 100
eind

leer kern
dr [voer keuze in]
maak "k ll
maak "keuze eerste :k
maak "score :score + 1
als :keuze < :getal [dr [te laag]]
als :keuze > :getal [dr [te hoog]]
als :keuze = :getal [slot]
kern
eind

leer slot
dr (zin [geraden in] :score [beurten])
dr [wil je stoppen toets s]
maak "j lk
als :j = "s [topnivo stop] [raadspel]
eind
```

J.Verwimp

van vrienden moet je 't hebben....

uw beduimeld, bevuild en grijsgelezen exemplaar van MSX CLUB MAGAZINE....

zeg ze toch gewoon dat een eigen abonnement véél leuker is,

uw en ons blad wordt er beter van ...

LOGO18 D2

Logo spelenderwijs

Een goede handleiding

De logo programmeertaal biedt vele mogelijkheden om zelf actief kennis op te bouwen. Seymoer Papert, de geestelijke vader van de computertaal logo, zegt dat de logotaal vooral ontworpen is om al doende te leren programmeren. We willen daarbij aansluiten door bouwstenen en interessante voorbeelden aan te dragen als inspiratiebron voor "doe het zelvers". Deze rubriek staat dan ook open voor alle logogebruikers die iets van hun ontdekkingstochten in de logowerelden willen vertellen of laten zien.
Wie een meer systematische aanpak verkiest kunnen we verwijzen naar het boek van **Albert Sickler MSX-LOGO spelenderwijs** uitgegeven bij Kluwer. De auteur presenteert de taal logo op een doordachte en goed opgebouwde manier. De vele facetten van logo komen aan bod. Na een spannende kennismaking volgen het samenbundelen en herhaald uitvoeren van opdrachten ; dan volgt de plaatjeswereld ; het rekenen met logo ; de woorden en lijsten in de boeiende taalwereld en de krachtige recursieve procedures om zeer compacte programma's te leren schrijven. Het laatste hoofdstuk biedt een interessant totaaloverzicht van alle opdrachten of primitieven. Als we daarbij nog vertellen dat de 176 pagina's gekruid zijn met plezierige en uitdagende voorbeelden, dan kan dit boek voor alle logoliefhebbers eveneens een spannende verkenning worden en zeker een aantrekkelijke handleiding die gemakkelijk kan geraadpleegd worden.

Schildpadden in vele gedaanten

Een vertraagde film

De razendsnelle tekenschildpad volgen in recursieve procedures is niet makkelijk. Met de wacht-opdracht kunnen we de schildpad wel tijdelijk stoppen, maar het is beter de hele beweging te vertragen om een goed inzicht in de werking van een procedure te verwerven. In elke opdracht kunnen we daartoe de opdrachten **vt, at, li** en **re** vervangen door de vertraagde opdrachten **v, a, l** en **r**.

```
leer v :n
herhaal :n [vt 1]
eind
```

Of nog een grotere vertraging door **wacht** eveneens in te schakelen in de herhaalopdracht.

```
leer a :n
herhaal :n [at 1 wacht 3]
eind


leer tragester
ms ss
po zetpos [-25 -50] pn
herhaal 12 [v 110 r 150]
eind
```

Het is wel duidelijk dat de 4 vertragingsprocedures wel moeten gedefinieerd worden. We testen ze daarna ook in de procedure ster.

```
leer ster
v 50 r 150
ster
eind
```

Als we onze voorwaarden niet opleggen met de als-opdracht zal de procedure ster niet stoppen. Met een hulp-procedure kleurster geven we de variabele :stap een beginwaarde.

```
leer kleurster
ms ss
maak "stap 1
drukaf [stap is]
ster
eind

leer ster
als :stap  36 [zetpk 1]
als :stap  72 [zetpk 8]
als :stap  108 [zetpk 10]
als :stap  144 [sw stop]
vt :stap re 150
maak "stap :stap + 1
zetcursor [2 2] drukaf :stap
ster
eind
```

Meer schildpadden, kleuren en vormen

Bij het aanzetten van de computer zien we op het scherm schildpad 0, maar er zijn er 30 genummerd van 0 t/m 29. We kunnen ze samen of afzonderlijk oproepen of opdrachten geven met de primitieven : zeg, elk, wie, allemaal, vraag en kleursp. Een hoofdstuk om veel plezier aan te beleven. Standaard zijn er ook reeds andere vormen voor de schildpad aanwezig. We laten ze zien in een leuk dansje. In **plaats** geven we de schildpadden een kleurtje mee en in **dans :n** zal

de opdracht **zetv :n** al de vormen tonen. **Podium** geeft de start voor de show.

```
leer podium
ms ss
maak "n 0
zetv 0
dans :n
eind

leer plaats
zeg [1 2 3 4 5 6 7 8 9 10 11 12]
elk [kleursp wie] ts po
elk [re wie * 30] v 30
eind

leer dans :n
als :n = 12 [stop ms ss]
kleurscherm 13 ts
plaats
ts re 90 herhaal 60 [vt 5 re 6] li 90
v 10 herhaal 60 [vt 3 li 24] a 40 richt 0
elk [zetv :n]
dans :n + 1
eind
```

Nieuwe vormen ontwerpen

Misschien stoeide je al met de vorm-editor om een persoonlijke sproken-bibliotheek op te bouwen. Indien niet dan willen we graag een eindje mee opstappen. Met de opdracht **ev (edit vorm)** kunnen we de bestaande vormen oproepen als we ook het juiste vormnummer toevoegen. Waarna we met de cursor-besturingstoetsen en de spatiebalk vakjes in het patroon kunnen vullen of leeg maken. De betekenis van de toetscombinaties CTRL/K CTRL/Y en CTRL/STOP leer je best in het gebruik. Met de ESC-toets zet je de nieuwe vorm vast.
We testten 4 nieuwe vormen en zullen de gevolgde weg kort beschrijven. Om het gemakkelijk te maken vertrekken we van een bestaande vorm. Met **ev 0** roepen we de bal op. Omdat we de relatie bal en het vormnummer 0 willen behouden, kopieren we de balvorm naar de door ons gekozen nummers 11, 12, 13 en 14 met de opdracht **kopiev 0 11, kopiev 0 12 ...** Met ev 11 bekomen we nu ook de balvorm, brengen de gewenste wijzigingen aan en fixeren de pac-man-vorm met de ESC-toets, waarna 12, 13 en 14 aan de beurt komen. Even kijken of de nieuwe vormen wel aanwezig zijn.

```
ms ss zetv 0 ts
herhaal 15 [zetv vormn + 1 wacht 50]
```

Sproken bewaren en opnieuw laden

Daar logo getallen gebruikt om de sproken te bewaren, moeten we de goede strategie gebruiken om plezier van onze inspanningen te hebben. Logo heeft hiervoor wel de opdrachten in voorraad.

```
noem pakv 11 "links
noem pakv 12 "rechts
noem pakv 13 "omhoog
noem pakv 14 "omlaag
```

De nummers bij iedere nieuwe vorm kregen zojuist een naam (een variabele) en kunnnen nu in groep bewaard worden onder de naam **hapvormen.** Met de opdracht inhoud kunnen we testen of de vormen weggeschreven zijn. Na reset laden we hapvormen opnieuw in het geheugen van de computer en geven de patronen opnieuw aan de vormnummers door met

```
neemv 11 :links
neemv 12 :rechts
neemv 13 :omhoog
neemv 14 :omlaag
```

Animatie met sproken

De procedure happen toont een eenvoudig voorbeeld. Zoals hierboven beschreven moeten de hapvormen met de opdracht **neemv** eerst aan de variabelen worden toegewezen. Dit kan ook met de procedure **roephapvormen.**

```
leer roephapvormen
neemv 11 :links
neemv 12 :rechts
neemv 13 :omhoog
neemv 14 :omlaag
eind

leer happen
ms ss
roephapvormen
herhaal 10 [zetv 0 wacht 20 zetv 11 wacht 20 zetv 0 wacht
20 zetv 12 wacht 20]
ms ss
eind
```

In **bewegen** wordt de waarde van **de variabele :richting** bepaald door de lk-opdracht, zo kunnen we de 4 hapvormen met de cursortoetsen besturen. (De karakterwaarde van de cursortoetsen vinden we met de opdracht **drukaf ascii lk** in te voeren en dan op de gewenste cursortoets te drukken.)

```
leer bewegen
zetv 0 po ts
maak "richting lk
als :richting = kar 29 [richt 270 maak "n 11]
als :richting = kar 28 [richt 90 maak "n 12]
als :richting = kar 30 [richt 0 maak "n 13]
als :richting = kar 31 [richt 180 maak "n 14]
als :richting = "s [zetvaart 0 sw stop]
zetv 0 wacht 10 zetv :n wacht 20
zetvaart 10
bewegen
eind
```

Daarmee zijn de eerste bouwstenen voor een computerspel aanwezig. Nu moeten de **demonen** nog wakker geschud worden om onder bepaalde voorwaarden een reeks opdrachten af te werken.
We zullen daarvan voorbeelden geven in een volgend artikel. Ondertussen verwachten we ook uw steentje.

J.Verwimp

Het boek van Albert Sickler MSX-LOGO spelenderwijs kan bij de club besteld worden voor de prijs van 575 frank verzending inbegrepen.

Tekenen in logo

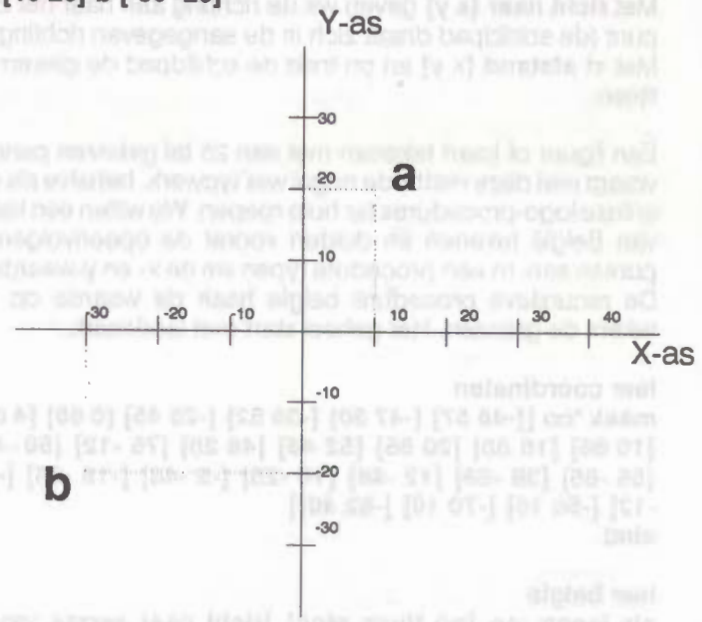
M20LOGO D2

Inleiding

Met de schildpad-opdrachten vt, at, re, li, po, pn en de beschikbare kleuren kunnen we mooie en effectvolle tekeningen op het scherm toveren. Vooral in recursieve procedures kan de schildpad met olympische snelheden de opdrachten uitvoeren. Gelukkig zijn er ook andere mogelijkheden.

Een andere aanpak

Om meer onregelmatige figuren te tekenen kunnen we ruitjespapier gebruiken. Zo kunnen we vakjes tellen en de te volgen weg vooraf bepalen. Indien we plattegronden of kaarten willen tekenen, dan kunnen de x- en y-as onze pen oriënteren. Met het rechthoekig coördinatenstelsel kunnen we van een tekening ieder punt aangeven door de x- en y-waarde te noemen. De punten a en b duiden we aan met : **a [10 20] b [-30 -20]**



Willen we met logohulp onze kennis van aardrijkskunde wat opfrissen dan kunnen we meerdere methodes uittesten om kaarten op het scherm te tekenen.
We kunnen op zicht werken, stap voor stap, of we kunnen de verhoudingen schatten en de opdrachten in een proce-

dure zetten. Zijn we niet sterk in het schatten en willen we de verhoudingen correct weergeven dan werken we veiliger met coördinaten.
Met de procedure hulpscherm kunnen we het beeldscherm vooraf keurig verdelen. De puntjes storen niet en het nulpunt van de x- en y-as vinden we met de opdrachten thuis en ts. De assen laten we weg om de tekeningen niet te ontsieren.

```
leer hulpscherm
tw sw po zetpos [-120 70]
herhaal 14 [puntrij po zetpos lijst -120 ycoor - 10] po thuis
eind

leer puntrij
tw zetpk 15
herhaal 23 [po zetpos lijst xcoor + 10 ycoor pn punt lijst
xcoor ycoor]
eind
```

Tekenen met coordinaten

De opdrachten : zetpos, richt, naar en afstand zullen de schildpad orienteren op het beeldscherm.

Met **zetpos [x-waarde y-waarde]** plaatsen we de schildpad op het 1ste punt (het vertrekpunt van de tekening).
Met **richt naar [x y]** geven we de richting aan naar het 2de punt (de schildpad draait zich in de aangegeven richting).
Met **vt afstand [x y]** en pn trekt de schildpad de gewenste lijnen.

Een figuur of kaart tekenen met een 25 tal gekozen punten vraagt met deze methode nogal wat typwerk, behalve als we enkele logo-procedures ter hulp roepen. We willen een kaart van België tekenen en duiden vooraf de opeenvolgende punten aan. In een procedure typen we de x- en y-waarden. De recursieve procedure belgie haalt de waarde op en tekent de grenzen. Het geheel start met landkaart.

```
leer coordinaten
maak "co [[-48 57] [-47 50] [-39 52] [-25 45] [0 65] [4 60]
[10 66] [16 60] [20 65] [52 43] [48 20] [75 -12] [50 -40]
[55 -65] [38 -68] [12 -48] [10 -25] [-5 -40] [-18 -35] [-25
-12] [-56 15] [-70 10] [-82 40]]
eind

leer belgie
als leegp :co [po thuis stop] [richt naar eerste :co vt
afstand eerste :co]
maak "co me :co
belgie
eind

leer landkaart
hulpscherm
tw po zetpos [-82 40] pn kleursp 15 ts
coordinaten belgie sw
eind
```

Tekenen met toetsbesturing

Naast het tekenen met de grafische kommando's of met hulpschermen en coördinaten kan je de tekenschildpad ook besturen met de cursortoetsen.
Het principe is eenvoudig. We maken een variabele **maak "toets lk**, en leggen onze voorwaarden op met de als-opdracht. In de recursieve procedure **cursorstuur** zie je enkele voorbeelden. We weten reeds dat elk letterteken kan omgezet worden in een code of getal. Met de opdracht **drukaf ascii lk** kan je testen welke code bij welke toets hoort. (Zie ook het hooofdstuk over ascii- en kar-waarden in magazine 16 blz. 21)

```
leer cursorstuur
maak "toets lk
als :toets = kar 30 [grens richt 0 vt 8]
als :toets = kar 31 [grens richt 180 vt 8 richt 0]
als :toets = kar 28 [grens richt 90 vt 8 richt 0]
als :toets = kar 29 [grens richt 270 vt 8 richt 0]
als :toets = kar 118 [vulvierkant]
als :toets = kar 119 [wisvierkant]
als :toets = kar 115 [stop]
cursorstuur
eind
```

In **grens** bakenen we het terreln af waarbinnen we de schildpad laten rondhuppelen (telkens 8 stappen).

```
leer grens
als ycoor  54 [zetpos lijst xcoor ycoor - 8]
als ycoor  -62  [zetpos lijst xcoor ycoor + 8]
als xcoor  -104 [zetpos lijst xcoor + 8 ycoor]
als xcoor  96 [zetpos lijst xcoor - 8 ycoor]
eind
```

Vul- en wisvlerkant spreken voor zichzelf en verschillen alleen in zetpk. We tekenen met deze vierkantjes alleen in zwart/wit. Om in kleur te werken moet je na **zetpk** het passende getal meegeven, nieuwe vierkantjes ontwerpen of de procedure aanpassen zoals in **cursortekenen.**

```
leer vulvierkant
pn zetpk 1
herhaal 4 [vt 8 re 90]
po re 45 vt 3 pn vul po at 3 li 45
eind

leer wisvierkant
pn zetpk 15
herhaal 4 [vt 8 re 90]
po re 45 vt 3 pn vul po at 3 li 45
eind
```

```
leer rooster
sw po richt 90 zetpk 7 zetpos [-112 72]
herhaal 17 [po zetx -112 zety ycoor - 8 pn vt 224]
po zetpos [-120 64] re 90
herhaal 29 [po zetx xcoor + 8 zety 64 pn vt 128]
po thuis ts
eind
```

Vierkanttekenen is de startprocedure. Wens je toch op een ruitjesscherm te tekenen dan moet je op de voorlaatste regel **rooster** toevoegen.

```
leer vierkanttekenen
ss sw po zetpos [-112 64]
pn zetpk 15 herhaal 2 [re 90 vt 224 re 90 vt 128] richt 120
po vt 5 zetpk 15 pn vul po thuis kleursp 7 ts
cursorstuur
eind
```

En nu gewoon lijnen tekenen met de cursortoetsen.

```
leer cursortekenen :v
tw pn maak "toets lk
als :toets = kar 30 [richt 0 zetvaart :v]
als :toets = kar 31 [richt 180 zetvaart :v]
als :toets = kar 28 [richt 90 zetvaart :v]
als :toets = kar 29 [richt 270 zetvaart :v]
hulpje
cursortekenen :v
eind
```

In **hulpje** kunnen we met **s** de schildpad stoppen en met **k** een kleur kiezen, waarna de cursortoetsen weer verder tekenen. Je kan hulpje met vele mogelijkheden uitbreiden.

```
leer hulpje
als :toets = kar 115 [zetvaart 0]
als :toets = kar 107 [typ [kleur ?] zetpk eerste ll]
eind
```

Met joystick tekenen

Bezitters van een joystick kunnen de tekenschildpad over het scherm laten bewegen en ondertussen opdrachten laten uitvoeren. Vooraf kunnen we kennis maken met de nieuwe opdrachten : **joy, knopp** en **joyp.** Zo kunnen we de stand van de joystick uitlezen met behulp van een kleine procedure.

```
leer joystick
drukaf joy 1
joystick
eind
```

Als we de stick bewegen zal bij elke stand een cijfer afgedrukt worden. In **joysticktekenen** kan je nu begrijpen hoe de schildpad de juiste richting kan nemen met **richt (-45 + 45 * joy 1).** Door de knop in te drukken stopt de schildpad.

```
leer joysticktekenen :vaart
tw ts pn
als joyp 1 [richt ( -45 + 45 *joy 1 ) zetvaart :vaart]
als knopp 1 [zetvaart 0]
als toetsp [joyhulp]
teken :vaart
eind
```

In deze recursieve procedure is de regel **als toetsp [joyhulp]** belangrijk. Bij het gebruiken van de procedure zullen we immers vlug vaststellen dat we een beetje hulp goed kunnen gebruiken. Het kommando **toetsp** zal voor ons nakijken of er een toets ingedrukt wordt en dan de procedure **joyhulp** aanspreken.
De procedure **joyhulp** kan iedereen zelf opzetten. **Hulpje** kan als voorbeeld dienen. En nu maar uitproberen en nieuwe mogelijkheden ontdekken om uw tekeningen en landkaarten te versieren.

Vergeet niet uw mooie tekeningen te bewaren met de opdracht **bewaarpl "naam**, zo kan je ze weer vlug op uw scherm laten verschijnen met **laadpl "naam.**

Veel logo-joy !

jaargang '85, '86 of '87 gemist ?

Breidt nu uw bibliotheek uit met de compilatieuitgaven van MSX CLUB MAGAZINE :

JAARGANG 85
(meer dan 50 programma's)

jaarboek 85	200 fr /fl 10
verzamelcassette '85	750 fr /fl 40
jaarboek '85 + cassette	750 fr / fl 40
disk ' 85	750 fr / fl 40
jaarboek '85 + disk	900 fr / fl 49

JAARGANG 86
(meer dan 60 programma's)

verzamelcassette '86	750 fr /fl 40
verzameldisk '86	750 fr /fl 40

JAARGANG 87
(meer dan 80 programma's)

verzamelcassette 87	750 fr /fl 40
verzameldisk 87	750 fr /fl 40

Het grote MSX Peeks, Pokes & Truuks boek deel II 295 fr / fl 14.75

bespaar Uzelf uren tikwerk en frustratie : de verzamelcassette/disk bevat alle programma's welke in de jaargang gepubliceerd zijn.

SUPERFONT

SUPERF BAS D2
SUPERF PIC D2

Superfont is een grafisch programma dat toelaat, op een eenvoudige wijze, tekst en tekeningen naar het MSX **scherm** of naar de **printer** te sturen. (MSX-printer of EPSON & compatibelen)

Superfont werkt met karaktersets (fonts) die zowel **tekst** als **tekeningen** kunnen bevatten.

Superfont maakt het mogelijk om uw programma's op een eenvoudige manier van animatie te voorzien.

Welke mogelijkheden biedt Superfont ?

* 40 karaktersets (zie voorbeelden op omslag p.2)

- normale karakters
- dubbel brede karakters
- dubbel hoge karakters
- inverse karakters
- bepalen voor- en achtergrondkleur van de karakters
- schrijfrichting veranderen
- proportioneel schrijven
- vertikale schrijfrichting
- roteren rond X-as of Y-as
- vergrotingsfaktor

* meer dan 1200 kant en klare hoge-resolutie tekeningen

(zie voorbeelden op omslag p.3)

- MSX 1 en MSX 2
- alle grafische schermmodes
- werken in windows : het grafisch scherm kan worden opgedeeld in 8 verschillende vensters met elk hun eigen karakteristieken
- sprites definiëren : met de honderden ikonen en tekeningen uit de superfont patroontabellen (140) is het heel eenvoudig om bewegende figuurtjes te definiëren

Wat kost Superfont ?

Superfont wordt geleverd op 3 schijven, met 3 duidelijke, geillustreerde handleidingen:

- SCHERMROUTINES
- PRINTERROUTINES
- SUPERFONTTABELLEN.

(Superfont bevat ook een schijf waarop de 1234 tekeningen kant-en-klaar zitten als stempels voor DYNAMIC PUBLISHER)

prijs : 3000 Bfr / Hfl 160

KERSTSPE D2

Kerstballenspel

Uit het boek **50 msx-logo projecten** kozen we voor de Kersttijd een leuk spel. Je moet proberen om 9 kerstballen vanuit de verte op bepaalde plekken in de kerstboom te gooien. Dat lukt als je ze met de goede snelheid en onder de goede hoek gooit. De plekken in de kerstboom kun je vantevoren zelf aangeven met de pijltjestoetsenbesturing. Sommige procedures zijn soms met opzet niet helemaal kompleet, maar wel met voldoende uitleg. We vinden het belangrijk dat je niet zomaar wat overtypt, maar dat je ook begrijpt wat je aan het programmeren bent.

Laten we dit kerstballenspel eerst in stukken verdelen :

```
leer kerstspel
kerstboom
cijfers
uitleg
gooien
afsluiting
eind
```

Nu eerst een procedure om de kerstboom te tekenen. Heb je al een idee hoe ?
Tip : begin vanuit de top van de boom eerst de rechterkant (met een zigzaglijn) dan de linkerzigzaglijn en tot slot de onderkant.

commentaar

```
leer kerstboom
ss ms kleurscherm 1          scherm helemaal zwart
zeg 0 sw zetpk 12            teken met groene pen
po zetpos [40 70]            top van de boom
pn richt 135                 naar rechts-beneden
herhaal 7 [vt 20 re 135 vt 8 li 135]
                             teken rechter zigzaglijn
po zetpos [.. ..]            terug naar de top
pn richt -135                naar links-beneden
herhaal ..[vt .. li ...
                             teken linker zigzaglijn
vt . re ...]
richt 90 vt 90 at.....
                             onderkant boom met stam
eind
```

Is de kerstboom gelukt ? Dan moeten nu de cijfers 1 t/m 9 in de boom gezet worden met een sterretje erbij (daar moeten straks de ballen komen). De plaats van de sterretjes kun je besturen met de pijltjestoetsen. Met een druk op de spatiebalk komt het volgende sterretje met cijfer op die plek. De xy-positie van elke ster moet je bewaren in een lijst "plaatsen. Dit alles kun je bereiken met de volgende procedure :

```
leer cijfers
zetcursor [0 21]
drukaf [zet 9 cijfers in de boom met]
drukaf [cursortoetsen en spatiebalk]
maak "cijfer 1
maak "plaatsen []
zeg 0 ts po zetvaart 8
sturen
zetvaart 0 sw
eind

leer sturen
maak "toets ".
als toetsp [maak "toets lk]
                             lees toets
als :toets = kar 28 [richt 90]
                             pijl-naar-rechts
als ......   kar 29 [..... ...]
                             pijl-naar-links
als ......   kar 30 [..... ...]
                             pijl-omhoog toets
als ......   kar 31 [..... ...]
                             pijl-omlaag toets
als ......   kar 32 [zetcijfer]
                             met spatiebalk
als :cijfer > 9 [stop]
eind

leer zetcijfer
zetvaart 0 sw pn             stop schildpad
herhaal 8[vt 3 at .. re ..]
                             teken een sterretje
po ts zetvaart ..            schildpad gaat verder
zetcursor cursorxy pos
                             zie hulpkommando's
typ :cijfer                  zet cijfer in de boom
maak "cijfer :cijfer .....
                             verhoog cijfer
maak "plaatsen zetl pos :plaatsen
                             zet xy-pos in de lijst
eind
```

Ziezo. Probeer nu maar eens uit of deze procedures werken. Zoja, dan komen nu de kerstballen.
Neem daarvoor bijv. de sproken 1 t/m 9. Je hebt daarvoor maar 1 vorm nodig, die je maakt met ev 1.

Deze kerstbal-vorm moet je daarna vastleggen met :

```
maak "kerstbal pakv 1
```

zodat de vorm straks ook op de diskette bewaard kan worden, samen met de procedures. Vervolgens ga je in de uitleg-procedure vertellen hoe je het spel moet spelen en ook enkele variabelen goedzetten :
- variabele "bal geeft aan met de hoeveelste bal je bezig bent in het spel
- variabel "beurt : de hoeveelste gooi met deze bal.

```
leer uitleg
drukaf [.................]
maak "beurt 1
maak "bal 1
neemv 1 :kerstbal
eind
```

En nu komt de **gooien**-procedure. Die is recursief. Telkens zet je de zoveelste bal in startpositie en vraag je de speler de gewenste richting en snelheid.
In de lijst "plaatsen staat de positie waar deze bal moet terechtkomen.

```
leer gooien
zeg :bal zetv 1 po
zetpos [-110 -30] ts zetvaart 0
zetcursor [0 19]
drukaf zin [MIK DEZE BAL OP POSITIE] :bal
zetcursor [0 21]
drukaf [in welke richting gooi je ?]
typ [( 0 = horiz,90 = vert )]
maak "richting eerste ll
richt 90 - :richting
zetcursor [0 21]
drukaf [en met welke snelheid ?]
typ [( mximaal 40) ]
maak "snelheid eerste ll
zetvaart :snelheid
maak "plaats element :bal :plaatsen
                        (dit is de plaats
                        in de boom waar
                        de bal heen moet)
maak "x eerste :plaats
maak "y laatste :plaats
gooi
als :bal > 9 [stop]
maak "beurt :beurt + 1
gooien
eind
```

Met de procedure **gooi** laat je de kerstbal door de lucht vliegen met een boogje en kijk je of hij goed terechtkomt. Het is raak, indien de afstand tussen de bewegende kerstbal en de plaats van het betreffende cijfer heel klein is.
Is het mis, dan stop je de bal wanneer hij buiten het schermbeeld dreigt te komen.

```
leer gooi
re 4
als 50 > (xcoor - :x) * (xcoor - :x) +
(ycoor - :y) * (ycoor - :y) [raak stop]
als of (xcoor > 120) (ycoor < -30) [stop]
gooi
eind

leer raak
zetvaart 0                  stopt de bal en zet hem
zetpos :plaats              precies op z'n plaats
zetcursor cursorxy :plaats
                            zet een spatie op de
typ spatie 1                plaats van het cijfer
maak "bal :bal + 1
zeg :bal                    neem volgende bal en
kleursp eenuit [2 5 8 10 11 13 ]
                            geef hem een kleur
eind
```

Ziezo, nu hoef je alleen nog de **afsluiting**-procedure te schrijven. Daarin zeg je in hoeveel beurten het gelukt is alle 9 kerstballen op te hangen en om te vragen of de speler nog eens wil spelen.

Logo-hulpkommando's

```
leer cursorxy :xy
geefterug lijst 28/240 * (120 + eerste
:xy) (-23/140 * (-70 + laatste :xy)
eind

leer eenuit :iets
geefterug element (1 + gok tel :iets)
:iets
eind

leer spatie :n
als :n = 0 [ gt kar 32]
gt woord kar 32 spatie :n-1
eind
```

Tip : Je kunt het spel verder uitbreiden door bijvoorbeeld een engeltje weer alle kerstballen uit de boom te laten schieten ! Daarvoor kun je bijna dezelfde procedures gebruiken als hierboven. Succes !

uit 50 MSX LOGO projecten
J. Verwimp

GRAPHIC MACRO LANGUAGE

Wanneer je op PC of op MSX een **GRAFISCH SCHERM** vraagt kan je tussen de aanhalingstekens van het bevel **DRAW "...enz..."** tal van korte en krachtige bevelen geven. Deze mogelijkheid noemt men de **GRAPHIC MACRO LANGUAGE**.

De bevelen zijn voor **PC** (in GW-BASIC) en voor **MSX** (MSX1 EN MSX2 BASIC) vrijwel gelijk, maar de grafische eigenschappen van de schermen zullen voor beide toestellen verschillen vertonen.

Vragen we **SCREEN 2** op **PC**, dan beschikken we horizontaal over 640 beeldpunten (genummerd van 0 tot 639) en vertikaal over 200 beeldpunten of **PIXELS** (genummerd van 0 tot 199). Deze **640 OP 200 PUNTEN** kan men apart aan of uit zetten maar veel meer zit er niet in. (het grafisch **SCHERM 2** heet **MONOCHROOM**).
Willen we op de PC toch meerdere (hoewel fel beperkte) kleurmogelijkheden, dan moet **SCREEN 1** gekozen worden met een **BEELDOPPERVLAK** of **PIXELOPPERVLAK** of **GRAFISCHE RESOLUTIE** van **320 OP 200**. (Aantal punten horizontaal is gehalveerd t.o.v. **SCREEN 2**).

Bij **MSX** zal **SCREEN 2** toelaten **256 OP 192** aan of uit te zetten in 16 kleuren genummerd van 0 tot 15. (normaal beperking tot 2 kleuren per vakje van 8 op 8 beeldpunten, maar na een kunstgreep om te zetten tot 2 kleuren per 8 beeldpunten horizontaal naast mekaar).

Beschikt men over een **MSX2-TOESTEL**, dan kan men na keuze van **SCREEN 7** beschikken over **512 op 212** pixels of beeldpunten, die men elk afzonderlijk **16** verschillende kleuren kan geven, genummerd van 0 tot 15. Via het **COLOR=(NR,R,G,B)** kan elk van die 16 kleuren (NR) veranderd worden door aan ROOD, GROEN en BLAUW (R, G en B) RGB-waarden van 0 tot 7 toe te kennen.
Bij **MSX2** kan ook **SCREEN 8** gekozen worden waar **ELK** beeldpunt rechtstreeks **256** verschillende kleuren kan aannemen (vb. te bereiken via **COLOR V,A,R** met Voorgrond, Achtergrond en Randkleur tussen 0 en 255).

Tussen de **AANHALINGSTEKENS VAN DRAW** worden bevelen gegeven die toelaten 1 of meerdere beeldpunten tegelijk aan of uit te zetten in een kleur die eventueel kan gekozen worden.

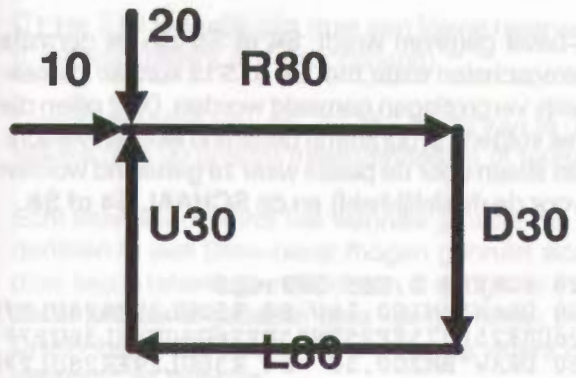
We gaan uit van het feit dat op een grafisch scherm een **GRAFISCHE CURSOR** bestaat waar de **X- en Y-WAARDE** wordt onthouden van de plaats waar de laatste tekenfunctie werd beëindigd. Het nulpunt van een grafisch scherm ligt **IN DE LINKER BOVENHOEK** van het beeldscherm.

De zuiverste manier om aan deze **GRAFISCHE CURSORPOSITIE** bijvoorbeeld de X-waarde 100 en de Y-waarde 60 te geven is : **DRAW"BM100,60"**. De BM staat hier voor **"BLANK MOVE"** wat zoveel wil zeggen als **"GA NAAR ... ZONDER IETS TE TEKENEN"** of eenvoudiger **"ZET JE KLAAR OP DE GRAFISCHE POSITIE ..."** of **"GEEF DE GRAFISCHE CURSOR VOLGENDE X- EN Y-WAARDEN"**.
De grafische cursor kan ook door bijvoorbeeld **PSET(X,Y)** of ook **PRESET(X,Y)** geplaatst worden maar dan moet je voortdurend bezorgd zijn om de kleur waarin deze punten gezet worden, dus gemakkelijker is **DRAW"BM..,.."** te gebruiken.

Vanaf deze grafische cursorwaarde kan je binnen de DRAW-aanhalingstekens met het bevel **U** een punt hoger komen op het scherm. Met **D** kom je 1 punt lager, terwijl **L** je 1 punt naar links brengt en **R** 1 punt naar rechts. (Zie engelse termen Up, Down, Left, Right). Na deze letter kan nog een cijfer aangegeven worden om de aktie een aantal maal te herhalen.
Zo zal **SCREEN 2:DRAW"BM10,20 R80 D30 L80 U40"** een rechthoek op het grafisch scherm brengen. Zowel voor PC als voor MSX is het nuttig een zogenaamde **"HOLD SCREEN"**-lijn (zie lijn 40 in volgend programma) in te bouwen, die ons toelaat het grafisch resultaat ongestoord te bekijken tot we een toets aanraken.

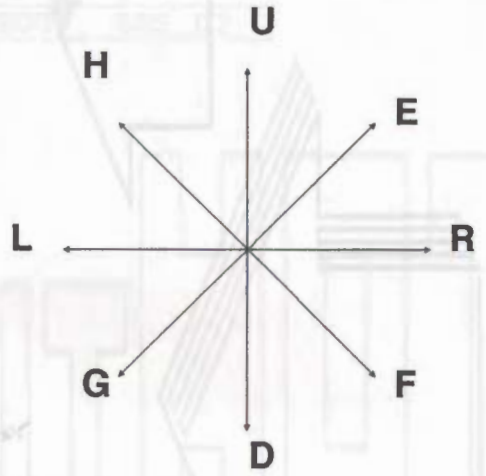
```
10 REM ### VOOR MSX EN PC ##
20 SCREEN 2:KEY OFF:CLS
30 DRAW"BM10,20 R80 D30 L80 U40"
40 I$=INKEY$:IF I$="" THEN 40
50 SCREEN 0:CLS:LIST
```

De spaties in het DRAW-bevel zijn niet noodzakelijk maar maken de zaak wel duidelijker.



FIGUUR 1

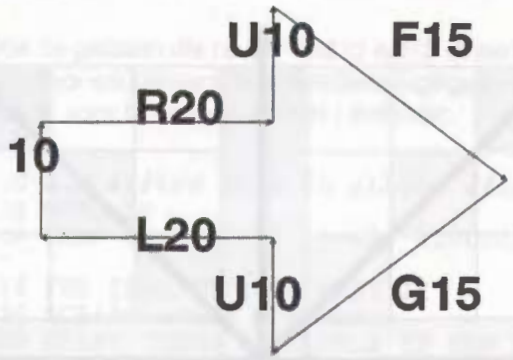
Met de letter **E** kan men schuin rechts omhoog, met **F** kan men schuin rechts naar beneden, met **G** schuin links naar beneden en met **H** schuin links naar boven op het scherm. Wordt **E, F, G of H** (vgl. alfabet !!) gevolgd door een cijfer of getal, dan wordt de verplaatsing voor zoveel punten herhaald als aangegeven in de geschreven getalwaarde.



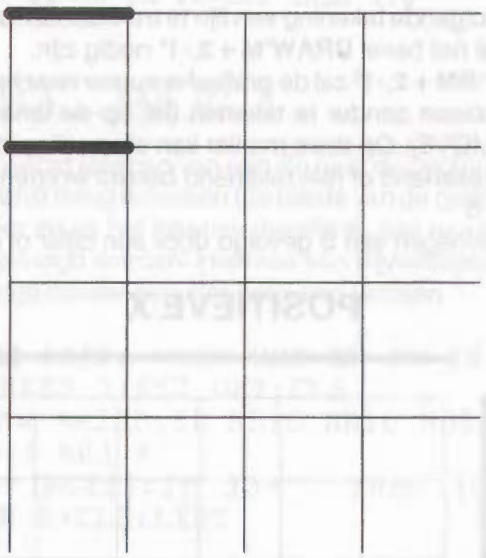
FIGUUR 2

```
10 REM ### VOOR MSX EN PC ###
20 REM ### TEKEN EEN PIJL ###
30 SCREEN 2:KEY OFF:CLS
40 DRAW"100,70 R20 U5 F15 G15 U10 L20 U10"
50 I$=INKEY$:IF I$="" THEN 50
60 SCREEN 0:CLS:LIST
```

FIGUUR 3

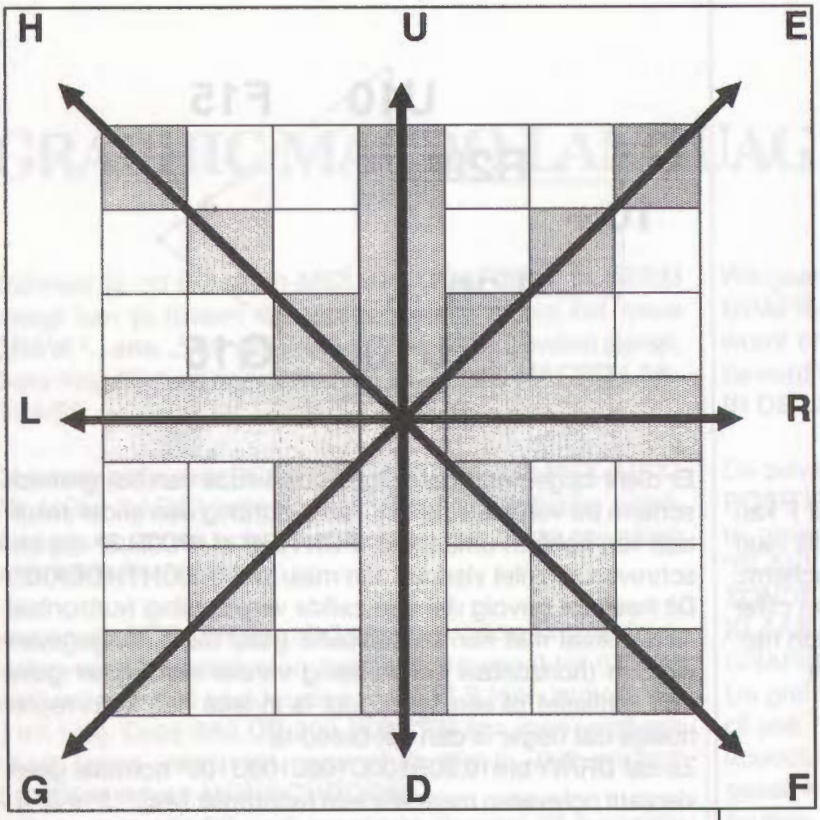


Er dient opgemerkt dat volgens de keuze van het grafisch scherm de verplaatsing in X- en Y-richting een ander resultaat kan hebben omdat die **"PUNTEN"** of **"PIXELS"** die beschreven zijn niet vierkant zijn maar wel **"RECHTHOEKIG"**. Dit heeft tot gevolg dat een zelfde verplaatsing horizontaal en vertikaal met een verschillend getal moet aangegeven worden (horizontale verplaatsing vereist een groter getal dan vertikale) of een beeldpunt is in feite een klein rechthoekje dat hoger is dan het breed is.
Zo zal DRAW"bm10,20R100D100L100U100" normaal geen vierkant opleveren maar wel een rechthoek (#@!?$*#@?). Hierdoor zal ook een volle lijn horizontaal er **VETTER** uitzien dan een volle lijn vertikaal.



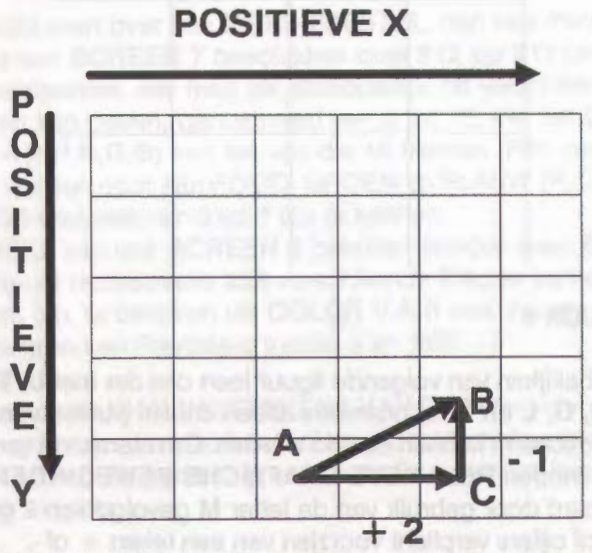
FIGUUR 4

Het bekijken van volgende figuur leert ons dat met **U, E, R, F, D, G, L en H** rechtstreeks alleen enkele punten van het beeldscherm kunnen bereikt worden. Om hieraan te verhelpen werden **RELATIEVE GRAFISCHE BEWEGINGEN** ingevoerd door gebruik van de letter **M** gevolgd van 2 getallen of cijfers verplicht voorzien van een teken + of -.



FIGUUR 5

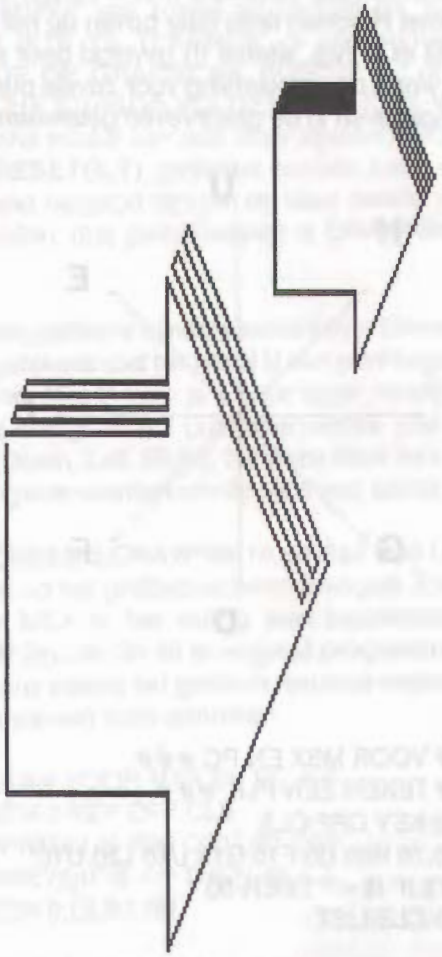
Om op volgende tekening een lijn te trekken van **vak A** naar **vak B** zal het bevel **DRAW"M+2,-1"** nodig zijn.
DRAW "BM+2,-1" zal de grafische cursor naar het beoogde vak sturen zonder te tekenen (let op de letter B in de BLANK MOVE). Op deze manier kan elk punt op het beeldscherm tekenend of niet tekenend bereikt worden.
FIGUUR 6
Door toevoegen van **S** gevolgd door een cijfer of getal kan



de schaal van het tekenen aangepast worden tot een nieuw S-bevel gegeven wordt. **S4 of S0** zijn de normale tekenschalen maar met **S8 of S12** kunnen gemakkelijk vergrotingen gemaakt worden. De 2 pijlen die met volgend programma getekend worden verschillen alleen door de plaats waar ze getekend worden (voor de duidelijkheid) en de **SCHAAL S4 of S8**.

```
10 SCREEN 2:KEY OFF:CLS
20 DRAW"BM100,100 S8 R30UL29ER28UL27ER
26U8F25EH25EF25EH25EF26G30U12L30U27"
30 DRAW"BM200,50  S4 R30UL29ER28UL27ER
26U8F25EH25EF25EH25EF26G30U12L30U27"
40 I$=INKEY$:IF I$="" THEN 40 ELSE SCR
EEN 0:CLS:LIST
```

PIJLEN BAS D2



Door toevoegen van **C** gevolgd door een cijfer of getal (dat binnen de heersende beperkingen in kleur ligt) kan het tekenen gebeuren in een andere kleur). Op **PC** zal **SCREEN 2** alleen **C1** (pixel aan) en **C0** (punt uit) kunnen gebruikt worden, maar op **MSX** zal in **SCREEN 2** de mogelijkheid van

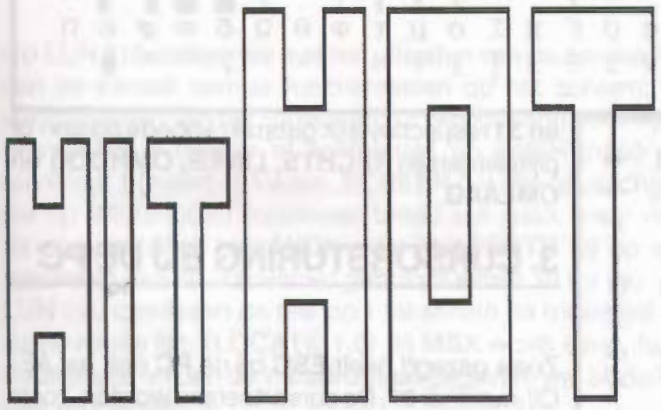
C1 tot **C15** mogelijk zijn (met een kleine reserve voor kleur 0, het transparant of doorschijnend).

Door toevoegen van **A0, A1, A2** of **A3** kan je de tekening telkens over 90 graden meer verdraaien of roteren.

Echt interessant wordt het wanneer je weet dat ook veranderlijken in een Draw-bevel mogen gebruikt worden. Hierdoor kan je tekeningen definieren of aangeven in een string, zodat je deze te gepaster tijde kan gebruiken in gelijk welk DRAW-bevel. Als voorbeeld worden de letters H, I, O en T getekend en gebruikt.

```
10 SCREEN 2:KEY OFF:CLS
20 H$="U20R5D5R5U5R5D20L5U5L5D5L5BR18"
30 I$="U20R5D20L5BR8"
40 O$="U20R15D20L15BE5U10R5D10L5BF5BR8
"
50 T$="BR5U15L5U5R15D5L5D15L5BR13"
60 DRAW"BM10,100 S8 XH$; XI$; XT$; S12
 XH$; XO$; XT$; S4"
70 I$=INKEY$:IF I$="" THEN 70 ELSE SCR
EEN 0:CLS:LIST
```

HITHOT BAS D2



Wordt een te tekenen string **BUITEN** de **DRAW-AANHALINGSTEKENS** aangegeven of gedefinieerd, dan kan die "**VARIABELE**" of "**VERANDERLIJKE**" ingelast worden door in de DRAW-aanhalingstekens na een **X** de naam van de string te schrijven, gevolgd door een ; of **puntkomma**.
In vorig programma werden de letters gedefinieerd in H$, I$, O$ en T$, om ze nadien in de DRAW te gebruiken. Je moet er alleen een **X** voorplaatsen en een ; achter.
In de definities van de letters merken we ook dat een **BE5** in een draw de grafische cursor 5 punten meer naar rechts en 5 punten meer naar boven brengt zonder te tekenen (zie de definitie van de O$).
Als men niet wenst voor verrassingen te staan is het altijd best bij het verlaten van een string, waarin de schaal werd veranderd met het **S-bevel**, de standaardwaarde **S4** aan te brengen, zodat alles weer is zoals normaal.

Ook de getallen die na een **BM** of een **S** of een **C** staan kunnen door middel van veranderlijken ingegeven worden door een = voor te voegen en een ; erachter.

```
10 REM ###### VOOR PC ALLEEN #####
20 GOTO 50
30 DRAW"BM=X%;,=Y%; C=C%; R20D20L20U20
"
40 FOR I%=1 TO 4000:NEXT I%
50 SCREEN 1:KEY OFF:CLS
60 PRINT "DEMO VARIABELE IN EEN DRAW [
ESC]=END"
70 PRINT "
        "
80 PRINT
90 PRINT "DUW 1, 2 OF 3  (VOOR DE KLEU
R) ? ";
100 I$=INKEY$:IF (I$<"1" AND I$<>CHR$(
27)) OR I$>"3" THEN 100
110 IF I$=CHR$(27) THEN SCREEN 2:CLS:L
IST:END
120 PRINT I$
130 C%=VAL(I$)
140 PRINT
150 INPUT "GEEF X + [ENTER] (0<X<200)
    ";X%
160 IF X%<0 OR X%>200 THEN 140
180 INPUT "GEEF Y + [ENTER] (60<Y<170)
    ";Y%
190 IF Y%<60 OR Y%>170 THEN 170
200 GOTO 30
```

COLSCA BAS D2

Als men na het tekenen van een lijn naar de beginplaats van die lijn wenst terug te komen (de plaats van de grafische cursor is voor en na het tekenen dezelfde), dan moet de letter **N** voorgevoegd worden. Hiermee kan bijvoorbeeld op een eenvoudige manier een ster getekend worden.

```
10 REM ##### VOOR MSX EN PC #####
20 SCREEN 2:KEY OFF:CLS
30 DRAW"BM100,80 NL10 NR10 NU10 ND10
NE3 NG3 NH3 F3"
40 I$=INKEY$:IF I$="" THEN 40 ELSE
SCREEN 0:CLS:LIST
```

Om dit stukje een beetje kracht bij te zetten en onderwijl de verbeelding aan te sporen werd een toepassing voor **PC** en **MSX1** en **MSX2** toegevoegd.

WOESTIJN MX1 D2

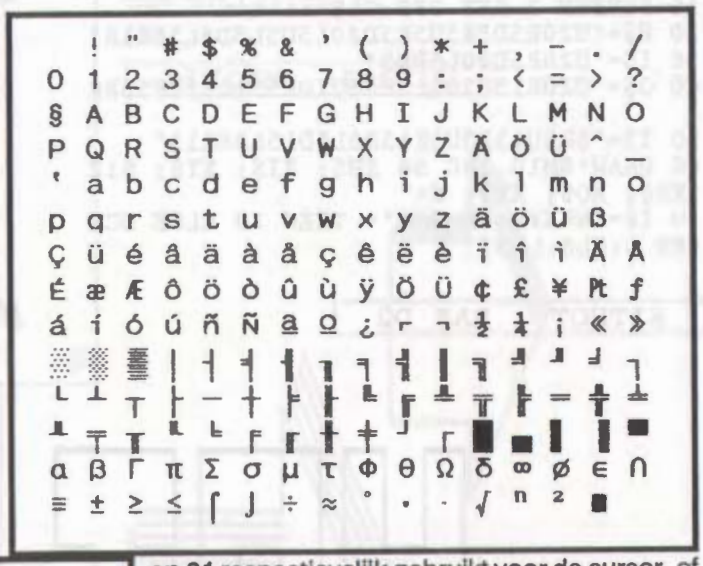
WOESTIJN MX2 D2

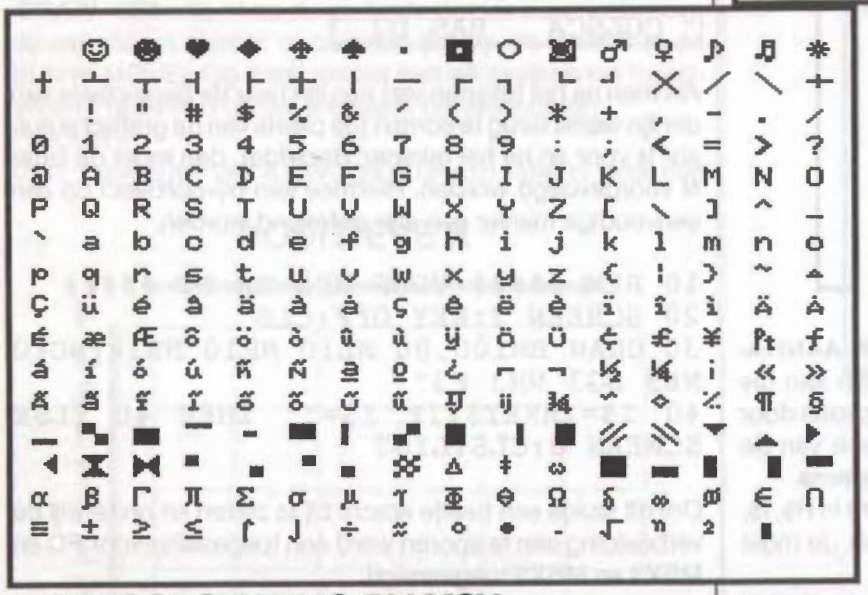
VERSCHILLEN IN CURSORSTURING TUSSEN MSX1/2 EN PC

1. ALGEMEEN

Wanneer we de verschillen bekijken tussen de sturing van de cursor bij **MSX** en de **PC of PC-COMPATIBLE**, dan valt de eenvoud van de programmering bij MSX op. Deze machines zijn dan ook voor de huiskamer gebouwd, voorzien van een uitgebreide BASIC interpreter, die de gebruiker zelf eenvoudig kan aanspreken.

De **ASCII-tabel** was oorspronkelijk bedoeld om hoofdletters en kleine letters, cijfers en leestekens door getallen weer te geven. Vooraan in de tabel staan stuurtekens die gebruikt worden om de communicatie tussen computers te vergemakkelijken. Slechts vanaf teken nummer 32 tot teken nummer 127 is er grote overeenkomst tussen de verschillende computermerken.





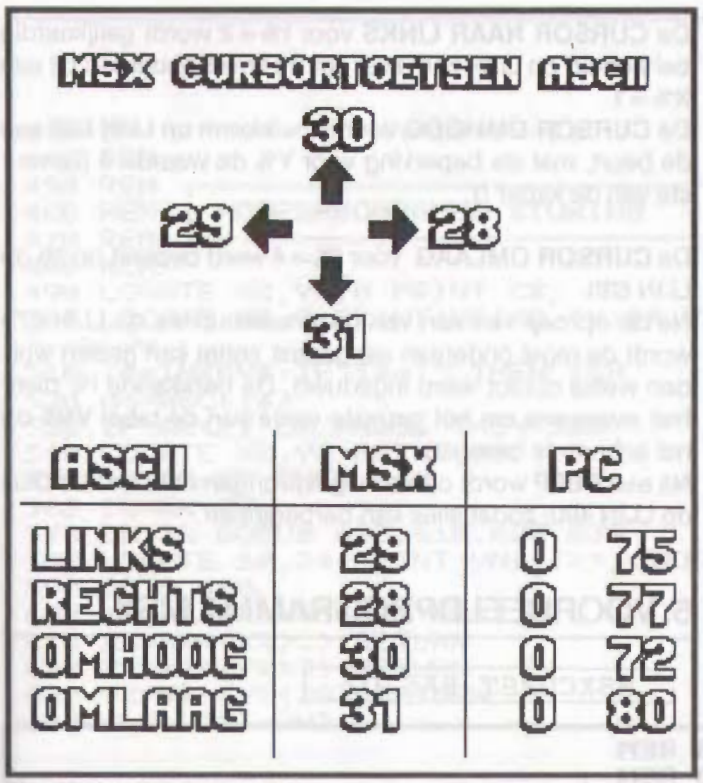
2. CURSORSTURING BIJ MSX

De escape-toets (**ESC**) kreeg bij de meeste toestellen het ASCII-nummer 27. Bij **MSX** werden de nummers **28, 29, 30 en 31** respectievelijk gebruikt voor de cursor- of pijltjestoetsen **RECHTS, LINKS, OMHOOG en OMLAAG.**

3. CURSORSTURING BIJ DE PC

Zoals gezegd heeft **ESC** bij de **PC** ook het ASCII-nummer 27. De cursortoetsen worden, zoals ook de functietoetsen, ontvangen onder de vorm van telkens **twee afzonderlijke getallen**, waarvan het eerste steeds het getal **NUL** is. Voor de cursortoetsen wordt dit getal **NUL** telkens gevolgd door het getal **72, 75, 77 of 80** voor respectievelijk **OMHOOG, LINKS, RECHTS en OMLAAG.**

Voorbeeld : Bij het intoetsen van **CURSOR LINKS** ontvangt **MSX** het **ASCII-nummer 29**, terwijl de **PC** achtereenvolgens het getal **NUL** en het getal **75** ontvangt.



ASCII	MSX	PC
LINKS	29	0 75
RECHTS	28	0 77
OMHOOG	30	0 72
OMLAAG	31	0 80

4. BESCHRIJVING VOORBEELDPROGRAMMA MSX1/2

Op LIJN 210 starten we met het uitzetten van de aanduiding van de inhoud van de functietoetsen op het scherm. De voorbereiding van het scherm wordt vervolledigd door voorgrondkleur 7 (cyaan of lichtblauw) en achtergrond- en randkleur 1 (zwart) te kiezen. **SCREEN 0** is een tekstscherm dat op **MSX2** tot 80 kolommen breed kan gaan, maar voor de compatibiliteit met **MSX1**, hier met **WIDTH 40** op een maximum van 40 kolommen gebracht wordt. (0 tot 39)
LIJN 220 localiseert de titel op 1 teken van de linkerkant op de bovenste lijn. (**LOCATE 1,0**) Bij **MSX** wordt eerst de **X-coördinaat** en dan de **Y-coördinaat** gegeven, wat bij de **PC** juist omgekeerd is.
De minimumwaarde is bij **MSX** zowel voor **X** als voor **Y** op **NUL** vastgesteld. Voor de **PC** zijn die waarden **1**.

Bij **LOCATE**-omzettingen van **MSX** naar **PC** moeten we letten én op het wisselen van **X** en **Y** én op het feit dat beide coördinaten met 1 moeten worden vermeerderd.
Op REGELNUMMER 280 wordt er op 2 lijnen van de bovenkant, geheel tegen de linker rand, begonnen met het kaderhoekje linksboven (**GRAPH** met **r**). Daar zowel bovenaan als onderaan de kader een horizontale lijn getrokken wordt, construeren we hiervoor een subroutine (LIJN 320) die 38 horizontale streepjes (**GRAPH** met **-**) na mekaar print (zie **;** na tweede aanhalingsteken). Na het oproepen van deze subroutine in LIJN 280 moeten we alleen nog het rechter bovenhoekje aangeven (**GRAPH** met **y**).
Van 3 tot 20 lijnen van de bovenkant brengen we de zijstreepjes (**SHIFT** met **GRAPH** en **|**) aan gescheiden door 38 spaties. (zie LIJN 290) **Opmerking :** Deze grafische codes zijn bestemd voor een **QWERTY-klavier**.
LIJN 300 begint met de linker onderhoek (**GRAPH** met **v**), gevolgd door 38 horizontale lijntjes die op het scherm gebracht worden in subroutine 320. De kader wordt op LIJN 300 afgesloten met de rechter onderhoek (**GRAPH** met **n**).
Op LIJN 310 springen we over de genoemde subroutine en de DATA van LIJN 380.

LIJN 390 zegt aan de BASIC-interpreter dat de **DATA** moeten gelezen worden vanaf lijn 380. (Deze lijn mag eventueel weggelaten worden).
Deze DATA zullen in een tabel ingelezen worden. Door lijn 400 wordt de grootte van die tabel vastgelegd op 4 vakjes. (Ook deze lijn is strikt genomen niet nodig daar de afmetingen van een tabel slechts moeten aangegeven worden als ze groter is dan 10 vakjes).
LIJN 410 zal deze DATA 1 voor 1 inlezen in de tabel **VN$** (van het woord VoetNota). In vakje 1 zit "**NAAR RECHTS**", in vakje 2 zit "**NAAR LINKS** ", in vakje 3 zit "**NAAR BOVEN** " en in vakje 4 "**NAAR ONDER** ".
LIJN 420 brengt de originele inhoud van de voetnota op het scherm op 1 teken van de linker kant en 24 lijnen van de bovenkant. Het teken **;** na het sluiten van de aanhalingstekens is nodig om het scherm niet te laten "**scrollen**" of omhoogwippen.

LIJN 430 geeft de coördinaten aan voor de beginstand van de cursor nl. op 19 tekens van de linker rand en 11 lijnen van de bovenkant (ongeveer in het midden van het scherm). De procenttekens wijzen op het feit dat het gehele getallen zijn (geen kommagetallen).

Op LIJN 490 begint pas de hoofdlus van het programma. De **LOCATE X%,Y%,0** plaatst de cursor op de zojuist aangegeven plaats. De **0** die als derde "parameter" in LOCATE staat zal de cursor op het scherm verborgen houden. Op die plaats brengen we dan met een **PRINT "C";** een hoofdletter C op het scherm. (kunstmatige cursor of kunstcursor)
LIJN 500 zal achteraan de eerste regel de waarden van X% en Y% aangeven. **PRINT USING "X:## Y:##";X%,Y%** laat toe voor het printen van de X- en de Y-waarde telkens 2 plaatsen te voorzien op het scherm (zie **##** na **X:** en **Y:**).

LIJN 510 laat toe de laatste intoetsing (**INKEY$**) te bewaren in de variabele **I$**. Zolang de I$ leeg is (**I$=""**) blijven we op lijn 510. Is er wel een intoetsing dan wordt op LIJN 520 de ASCII-waarde van de intoetsing bewaard in A% (ASCII-waarden zijn gehele getallen, dus % erachter).

Ons interesseren alleen de intoetsingen **ESC (ASCII = 27)** en de **CURSORTOETSEN (ASCII = 28, 29, 30 en 31)** zodat alles onder ASCII 27 en boven ASCII 32 mag als een "foute" intoetsing aanzien worden. LIJN 530 verlegt de aktie opnieuw naar LIJN 510 als zo'n foute intoetsing ontdekt werd.

Op LIJN 540 komen alleen de gezochte intoetsingen, zodat waarschijnlijk de cursor zal moeten verplaatst worden. We nemen onze voorzorgen door de geplaatste letter **C** te verwijderen met het printen van een **SPATIE** of **BLANCO** op dezelfde plaats, zodat op dit ogenblik de kader leeg is.

Was die intoetsing een **ESC** dan wordt op lijn 550 verwezen naar het einde van het programma op LIJN 640. Daar wordt het scherm geveegd, waarna de **LIST** van het programma automatisch op het scherm wordt gebracht.

De enige intoetsingen die LIJN 560 bereiken zijn de **CURSORINTOETSINGEN** met ASCII-waarde **28 tot 31**. Deze waarden die in A% werden gespaard zullen verminderd worden met 27. Dit resultaat wordt in I% opgeborgen.

A%	I%
28	1
29	2
30	3
31	4

De berekening leert dat alleen **1, 2, 3 en 4** mogelijk zijn. Volgens de cursortoets die werd ingeduwd zal op lijn 570 verwezen worden naar een andere subroutine. Hierbij helpt de berekende I%-waarde. I% is 1 als A% gelijk was aan 28, wat betekent dat **CURSOR RECHTS** werd ingeduwd. Hier moeten we de **X%-waarde** met 1 vergroten om de letter **C** naar rechts te brengen op het scherm. De eerste verwijzing op lijn 570 gebeurt naar de subroutine op lijn 600 (als I% = 1).

Op LIJN 600 mag de vermeerdering maar gebeuren tot X% de waarde 38 bereikt, daar anders een gedeelte van de kader zou overschreven worden. Om dit te voorkomen wordt op LIJN 600 gezegd dat X% slechts mag toenemen zolang X% kleiner is dan 38. Dit gebeurt door gebruik te maken van een logische vergelijking (**X%&**). Deze waarde tussen haakjes kan **WAAR** zijn of **NIET WAAR**.
Bij **NIET WAAR** wordt hiervoor **0** bekomen, terwijl bij **WAAR** de waarde **-1** verschijnt. Is LIJN 600 de X% kleiner dan 38, dan staat er in feite -1 tussen de haakjes. Het minteken zet deze bekomen waarde -1 dan om in een +1, zodat X% toch met 1 verhoogd wordt. Is X% = 38 dan komt er een **0** tussen de haakjes, zodat de X% niet kan veranderen.

De **CURSOR NAAR LINKS** voor I% = 2 wordt gelijkaardig behandeld op LIJN 610. Hier ligt de limiet onderaan bij een X% = 1.
De **CURSOR OMHOOG** voor I% = 3 komt op LIJN 620 aan de beurt, met als beperking voor Y% de waarde 3 (bovenste van de kader !).

De **CURSOR OMLAAG** voor I% = 4 werd beperkt op 20 op LIJN 630.
Na de oproep van één van deze subroutines op LIJN 570 wordt de regel onderaan aangepast zodat kan gezien worden welke cursor werd ingeduwd. De berekende I% dient hier eveneens om het gepaste vakje van de tabel **VN$** op het scherm te brengen.
Na een **BEEP** wordt opnieuw gesprongen naar de hoofdlus op LIJN 490, zodat alles kan herbeginnen.

5. VOORBEELDPROGRAMMA MSX

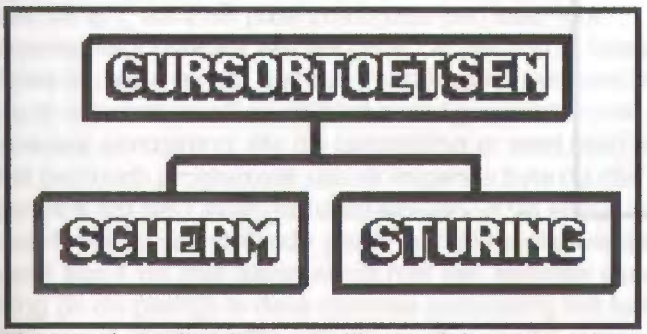
MSXCURST BAS D2

```
100 REM
110 REM
120 REM | CURSORSTURING MSX1 EN MSX2 |
130 REM | DE SPIEGELEER MARO 10-6-88 |
140 REM | ERPE-MERE        053/705162 |
150 REM
160 REM
170 REM
180 REM | HOOFDPROGRAMMA INLEIDING   |
190 REM
200 REM
210 KEY OFF:CLS:COLOR 7,1,1:SCREEN 0:W
IDTH 40
220 LOCATE 1,0:PRINT "CURSORSTURING MS
X"
230 REM
240 REM
250 REM | KADER                       |
260 REM
270 REM
280 LOCATE 0,2:PRINT "┌";:GOSUB 320:PR
INT "┐";
290 FOR I%=3 TO 20:PRINT "|";SPACE$(38
);"|";:NEXT I%
300 PRINT "└";:GOSUB 320:PRINT "┘"
310 GOTO 390
320 FOR I%=1 TO 38:PRINT "─";:NEXT I%:
RETURN
330 REM
340 REM
350 REM | VOETNOTA                    |
360 REM
370 REM
380 DATA "NAAR RECHTS","NAAR LINKS ","
NAAR BOVEN ","NAAR ONDER "
390 RESTORE 380
400 DIM VN$(4)
410 FOR I%=1 TO 4:READ VN$(I%):NEXT I%
420 LOCATE 1,24:PRINT "LAATSTE STURING
: GEEN          ESC=LIST";
```

```
430 B$="  ":C$="C":X%=19:Y%=11
440 REM
450 REM ┌──────────────────────┐
460 REM │ HOOFDPROGRAMMA STURING │
470 REM └──────────────────────┘
480 REM
490 LOCATE X%,Y%,0:PRINT C$;
500 LOCATE 30,0:PRINT USING "X:## Y:##
";X%,Y%
510 I$=INKEY$:IF I$="" THEN 510
520 A%=ASC(I$)
530 IF A%<27 OR A%>31 THEN 510
540 LOCATE X%,Y%:PRINT B$;
550 IF A%=27 THEN 640
560 I%=A%-27
570 ON I% GOSUB 600,610,620,630
580 LOCATE 18,24:PRINT VN$(I%);:BEEP
590 GOTO 490
600 X%=X%-(X%<38):RETURN
610 X%=X%+(X%>1):RETURN
620 Y%=Y%+(Y%>3):RETURN
630 Y%=Y%-(Y%<20):RETURN
640 CLS:LIST:END
```

6. BESCHRIJVING VOORBEELDPROGRAMMA VOOR PC IN GWBASIC

Als we nadien nog willen overgaan naar een blokgestructureerde taal zoals **TURBO PASCAL** dan structureren we het programma door vooraf een tekening te maken met **PST** (Programma Structuur Technieken) hierin wordt een groter programma in enkele stukken onderverdeeld.



Het totaal wordt gelezen per lijn van links naar rechts.

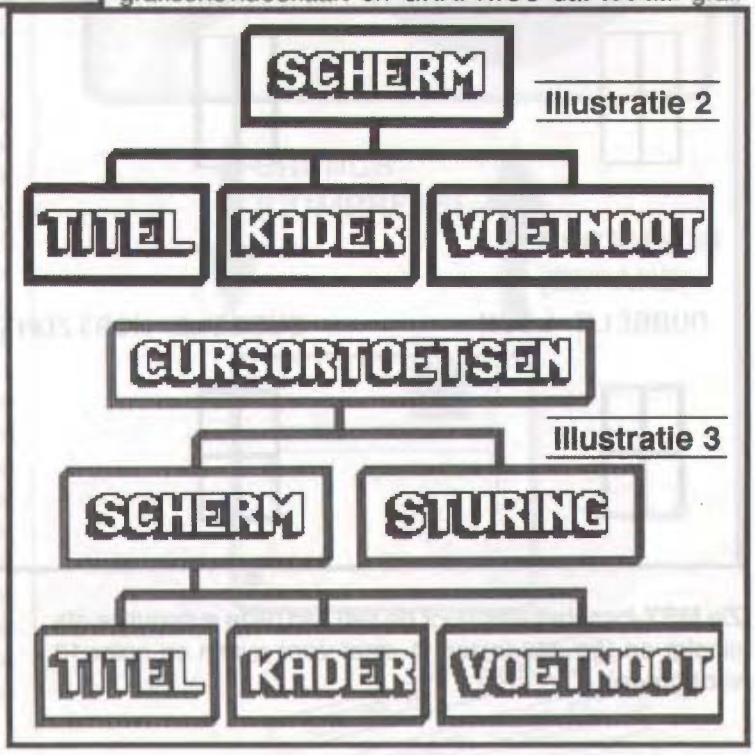
Laatstgenoemde onderverdelingen kunnen nog verder verfijnd worden.

(zie illustratie 2)

Zodat het totale **PST** er als volgt zal uitzien na 1 verfijning van de schermconstructie.

(zie illustratie 3)

Verfijningen gebeuren door inbouwen van subroutines waar we op **regel 170** overspringen om op **regel 680** te starten met het oproepen van de subroutine van de schermbouw. Deze gebeurt zoals bij het **MSX-programma**. Op **lijn 220** wordt het tekstscherm (screen 0) met een 80-koloms breedte en een groene kleur (10) op zwarte achtergrond (0) gevraagd.
Op **regel 230** werd rekening gehouden met de verwisseling van **X** en **Y** in de **LOCATE-opdracht**.
De grafische tekens (Hoekjes en streepjes) kunnen, hoewel ze niet op de **PC-toetsen** voorkomen, gevormd worden door de **ALT-toets** ingedrukt te houden en onderwijl het **ASCII-nummer** voor het grafische teken te tijpen. Pas na het loslaten van de **ALT-toets** verschijnt het beoogde teken. Op **regel 330** wordt het horizontale grafische streepje (gevormd door **ALT** in te drukken en onderwijl 196 te tijpen waarna de **ALT-toets** losgelaten wordt).
Opmerking : Bij het opstarten van de computer geeft men best volgende **MS-dos** opdrachten in de **AUTOEXEC.BAT** : **GRAFTABL** die de **PC-ASCII-tekens** van 128 tot 255 toegankelijk maakt voor de kleuren- of grafischevideokaart en **GRAPHICS** dat toelaat grafi-



Illustratie 2

Illustratie 3

sche afbeeldingen op een eventuele printer af te drukken.

```
ENKELVOUDIGE LIJN      DUBBELE VERTIKALE
┌ ─ ┬ ─ ┐              ╓ ─ ╥ ─ ╖
218 196 194 196 191    214 196 210 196 183
│   │   │              ║   ║   ║
179  32 179  32 179    186  32 186  32 186
├ ─ ┼ ─ ┤              ╟ ─ ╫ ─ ╢
195 196 197 196 180    199 196 215 196 182
│   │   │              ║   ║   ║
179  32 179  32 179    186  32 186  32 186
└ ─ ┴ ─ ┘              ╙ ─ ╨ ─ ╜
192 196 193 196 217    211 196 208 196 189

DUBBELE LIJN           DUBBELE HORIZONTALE
╔ ═ ╦ ═ ╗              ╒ ═ ╤ ═ ╕
201 205 203 205 187    213 205 209 205 184
│   │   │              │   │   │
186  32 186  32 186    179  32 179  32 179
╠ ═ ╬ ═ ╣              ╞ ═ ╪ ═ ╡
204 205 206 205 185    198 205 216 205 181
│   │   │              │   │   │
186  32 186  32 186    179  32 179  32 179
╚ ═ ╩ ═ ╝              ╘ ═ ╧ ═ ╛
200 205 202 205 188    212 205 207 205 190

DUW EEN TOETS ...

KADERS OP DE PC
```

ENKELVOUDIGE LIJN

DUBBELE VERTIKALE

DUBBELE LIJN

DUBBELE HORIZONTALE

Zie **MSX-beschrijving** voor de rest van deze subroutine die eindigt op **lijn 440**.(color 11 staat voor cyaan en color 10 voor groen)

De return brengt ons terug naar regel 690, waar de sturingsroutine op **lijn 500** geroepen wordt. Hieruit kunnen we slechts terugkeren nadat op **lijn 530** de **ESC-toets** ingedrukt werd.

De detectie van deze **ASCII 27** stuurt ons op **lijn 540** naar de **RETURN**, die de enige uitgang is van deze subroutine. Hiermee komen we aan het klassieke einde van het programma op **lijn 700**.

We weten dat de cursortoetsen op de **PC** elk het getal **NUL** sturen gevolgd door een tweede getal. Als geen twee opeenvolgende getallen ontvangen worden dan kan de intoetsing geen cursortoets zijn (**LEN(I$)0**) en wordt er terug gesprongen naar **lijn 530**. Op **lijn 560** wordt met **MID$(I$,2,1)** het teken gevormd dat overeenkomt met de **ASCII-waarde** van het tweede ontvangen getal. Met het **INSTR-bevel** wordt gekeken of dit getal **72 (ASCII voor H), 75 (ASCII voor K), 77 (ASCII voor M) en 80 (ASCII voor P)**. In **I%** zit dus de eventuele volgordeplaats in de reeks "**HKMP**". Komt dit tweede getal in deze reeks niet voor dan wordt **I%=0** en vragen we op **lijn 530** een nieuwe intoetsing. De rest van de routine verloopt parallel met de **MSX-routine** op het **SOUND-bevel** na. **SOUND** verwacht twee parameters of gegevens nl. de frequentie (hier 2500 Hz) en de lengte (hier 0,1 seconden). In het programma **PCURSOR2.BAS** werden **LIJN 550 en 560** vervangen. Achteraan **LIJN 560** wordt met een spitsvondige, zeer vergezochte kunstgreep 72 naar 1, 75 naar 2, 77 naar 3 en 80 naar 4 omgezet door gebruik te maken van de afronding bij berekening met gehele getallen (controleer zelf **I%**)

Crunch-algoritme voor grafische schermen

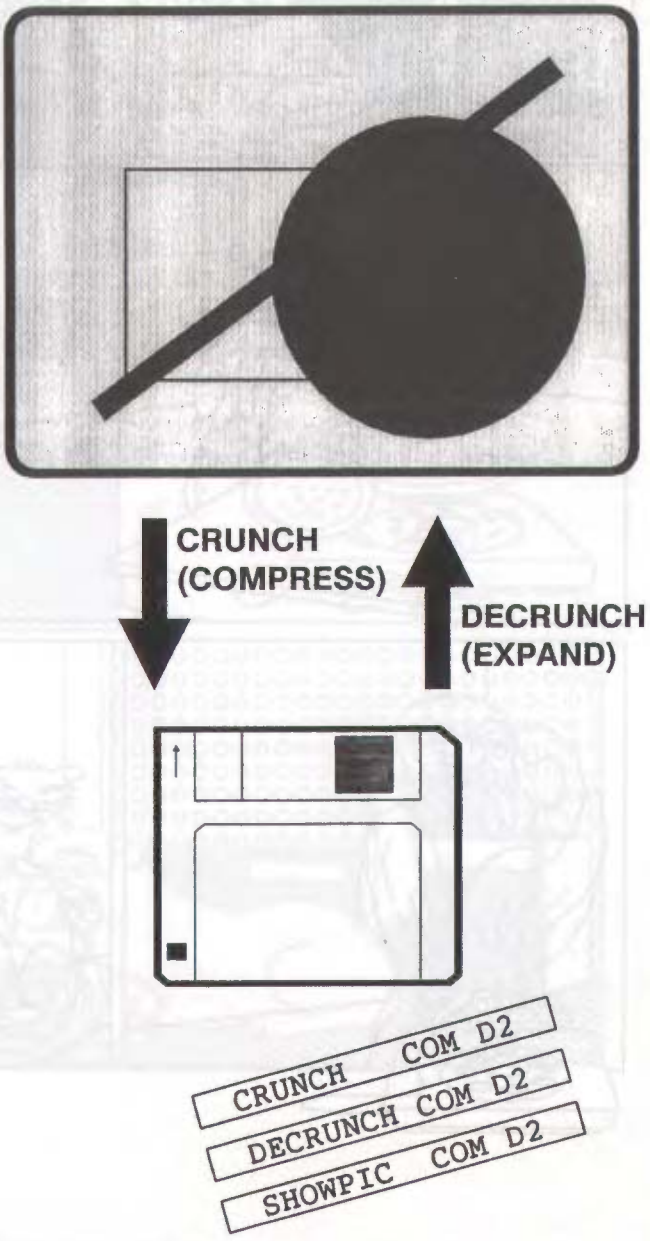
Wat is crunchen ?

Als je graag gedigitaliseerde beelden bijhoudt en niet teveel disketjes wilt verspelen, dan kunnen de volgende kleine programma's zeer nuttig zijn. Het programma 'crunch' zal proberen een grafisch plaatje op een zeer compacte manier op disk te zetten. Stel bijvoorbeeld dat je scherm in de file PROEF.PIC staat (dit scherm mag van eender welke grafische mode zijn). Als je dan crunch opstart via MSX-DOS, zal hij eerst naar de naam van de file vragen. Tik PROEF.PIC in. Dan vraagt hij naar de nieuwe naam. Bijvoorbeeld PROEF.CRU. Het programma begint nu te crunchen en in bijna alle gevallen is het resultaat een kortere PROEF.CRU. Als je nu een gecrunchd scherm wilt bekijken, moet je het programma 'showpic' opstarten. Dit vraagt naar de nieuwe filenaam. Tik PROEF.CRU in en daarna de schermmode van het grafisch plaatje. Na een tijdje verschijnt het plaatje op het scherm.
Als je echter het scherm opnieuw wilt omzetten naar een originele file (om bijvoorbeeld opnieuw in basic te kunnen gebruiken), dan moet je 'decrunch' opstarten. Dit zal de gecrunchte file omzetten naar een normale file.

Hoe werkt crunch ?

Het programma gaat uit van het principe dat er in een grafisch scherm veel gelijke stukken zijn. Stel bijvoorbeeld dat er 20 pixels naast elkaar dezelfde kleur hebben. Dan gaat het crunch programma niet het pixel 20 keer herhalen, maar het getal 20 op disk zetten met daarna 1 keer het pixel. Dit zijn slechts 2 bytes in plaats van 20. Er is echter een probleem. Stel dat er 20 opeenvolgende verschillende pixels zijn. Dan zou het programma per pixel het aantal op disk zetten. Dit is 1, want elk pixel komt maar een keer voor.
Daarna volgt de kleur van het pixel. Dit zijn dan in totaal 40 bytes in plaats van 20, wat een verdubbeling betekent ! Om dit op te lossen wordt op disk het aantal geschreven met een speciale aanduiding. Als die aanduiding er staat, dan weet het decrunch programma dat de volgende byte op disk het aantal is dat die 1 moet herhaald worden. In het vorige voorbeeld met 20 verschillende pixels zou dit betekenen dat er eerst een 1 op disk gezet wordt met een speciale aanduiding (In de praktijk is deze speciale aanduiding het laatste bitje van 1, dus eigenlijk 128+1) en dan een 20 om aan te duiden dat er 20 pixels volgen die allemaal maar 1 keer herhaald moeten worden. In totaal zijn dit nu nog maar 22 bytes. Dit is maar 2 meer dan het oorspronkelijke.
Ondanks het feit dat het programma soms meer bytes nodig heeft, is het eindresultaat in bijna alle gevallen toch korter dan het origineel. En dit soms tot meer dan 50 % korter!
Het programma is in HISOFT-C geschreven en moet, misschien met enige aanpassing, ook werken op andere C-compilers. Het is niet erg snel omdat HISOFT-C niet erg snel is. Maar ik ben aan het werken aan verbeterde versies, dus... even geduld!

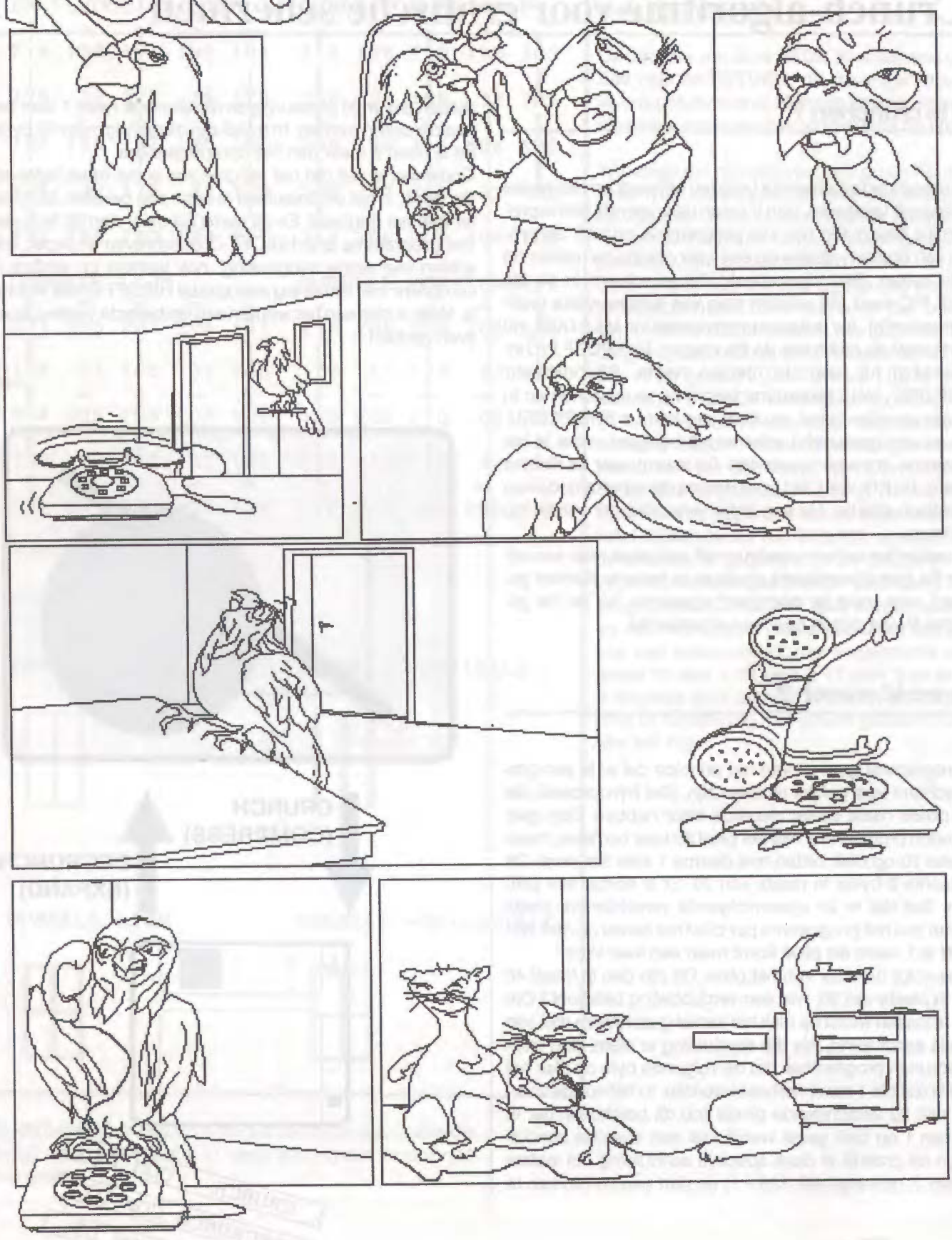
Jorrit

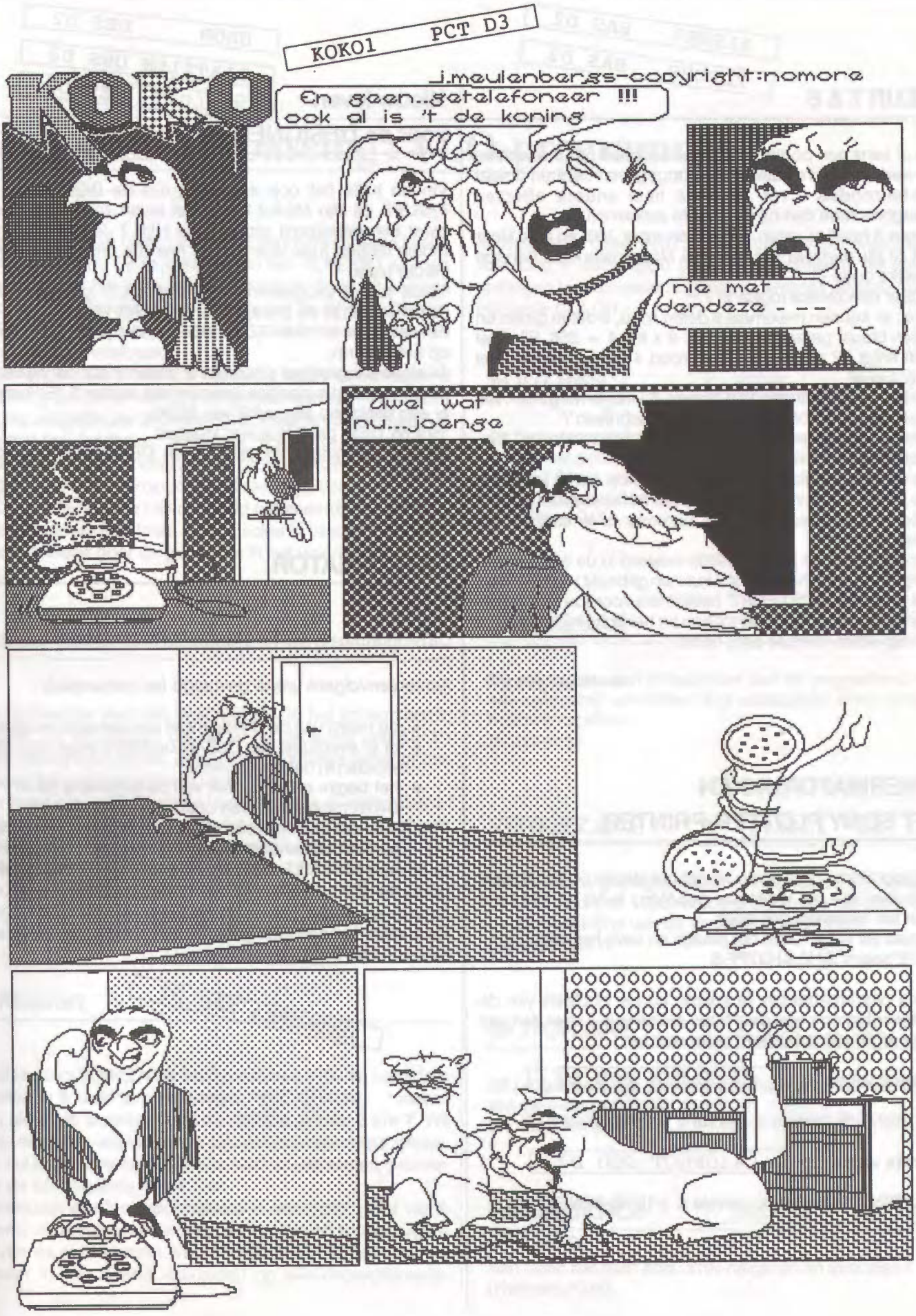


KOKO PCT D3

KOKO

Tekening: j.meulenbergs




KOKO1 PCT D3
J.meulenbergs-copyright:nomore
KOKO
En geen..getelefoneer !!!
ook al is 't de koning
nie met
dendeze - - -
Awel wat
nu...joenge

KLEUR7 BAS D2
KLEUR8 BAS D2

KLEUR 7 & 8

Wie al eens een poging ondernomen heeft om met screen 8 te werken , zal het allicht vlug opgegeven hebben omdat de kleurcodes 0 tot 15 plots heel andere effecten teweegbrachten dan bij de andere schermen.
Screen 8 heeft nl. geen 16 kleuren maar 256, en elke kleur heeft zo zijn nummer van 0 tot 255. Maar welke kleur behoort tot welke code ?
Zit daar een zekere logica in ?
Jawel: er kunnen maximaal 8 delen rood, 8 delen groen en 4 delen blauw gemengd worden: 8 x 8 x 4 = 256. Elk deel groen krijgt 32 punten, elk deel rood 4 punten en elk deel blauw 1 punt .
De kleurbeschrijvingen van screen 8 vind je nergens. Hoe zou je trouwens 256 kleurnuances beschrijven ?
Nochtans, er zijn er zo'n mooie bij dat het zonde zou zijn, ze nooit gezien te hebben.
Daarom het programma "KLEUR 8". Stop het in je MSX2, zet je kleur- monitor aan en geniet van het spektakel: je krijgt de kleuren gepresenteerd per nummer of je stelt ze zelf samen. Alles kan.
En misschien ben je ook geïnteresseerd in de 512 kleuren die bij MSX2 op scherm 0 tot 7 kunnen gebruikt worden. Dan is het programma "KLEUR7" beslist iets voor jou . Je kan er alle kleurcombinaties mee maken en het scherm verklapt je ook nog welke formule erbij hoort.

Veel kleurgenot !!

SCHERMAFDRUKKEN MET SONY PLOTTER-PRINTER

Het programma gaat ervan uit dat het plaatje op schijf staat en als deel van het Videoram geSAVEd is.Als dit niet zo is kun je het makkelijk zelf doen.
Het gaat als volgt. Laad het plaatje en save het met :
BSAVE"naam",&H0,&H37FF,S

Om de plotterpennetjes te sparen kun je in plaats van de hele tekening ook een deel laten afdrukken. Let er dan wel op dat regel 460 veranderd moet worden:

de **kleine** versie : `C41KLEIN SCD D2`

460 LPRINT "R (laatste x - eerste x +1)*-2,-2":NEXT Y

de **grote** versie : `C41GROOT SCD D2`

460 LPRINT "R (laatste x - eerste x +1)*-3,-3":NEXT Y

G. de Boom

SHOW DES D2
LAATZIEN DES D2
HAALUIT DES D2

Nieuw leven voor de DESIGNER van PHILIPS

Vinden jullie het ook zo spijtig dat de DESIGNER, die PHILIPS bij zijn MSX-2 toestellen levert (of leverde) toch maar een geïsoleerd programma blijft ? Je kan daar hele mooie dingen mee doen, maar met die ontwerpen ben je verder niets.
Tenzij je het programma "HAALUIT.DES" gebruikt, want daarmee kan je elk plaatje apart op schijf zetten.
En wat meer is : "LAATZIEN.DES" tovert je tekening terug op het scherm.
Je eigen programma's kunnen er alleen maar wel bij varen.
Of wil je met die plaatjes gewoon wat spelen ? (Er kunnen er een tiental op een schijf van 360K).
"SHOW.DES" geeft je al wat ideetjes.
De twee tekeningen "POJE.DES" en 'DRAAK.DES" staan op de schijf.

J.Simai

DATACREATOR

Dit programma is een luxe-versie DATACREATOR voor disk.

Achtereenvolgens wordt gevraagd (en behandeld) :

- de naam van de file die moet aangemaakt worden
- of er eventueel een CLEAR-opdracht moet voorzien worden in het programma
- het begin- en eindadres van de gegevens die in DATA moeten worden ondergebracht .(het zal meestal gaan om een stukje machinetaal, maar het kunnen ook andere gegevens zijn)
- eventuele DEF-USR-statements die ook in program-malijnen moeten worden ondergebracht.

Als al deze handelingen gebeurd zijn, vindt U op schijf een kant-en-klaar BASIC programma.

DATACREA BAS D2

een leuke toepassing : VLO-programma

VLO	BAS	D1

Spelletjes worden haast uitsluitend geprogrammeerd in een grafisch scherm. De mogelijkheden van SCREEN1 worden echter zelden ten volle benut. Nochtans drukt dit scherm ook reeds alle karakters in hun geheel af (in tegenstelling tot SCREEN0) en zijn er bovendien 16 verschillende kleuren door mekaar instelbaar.
Beperkingen zijn er wel. Zo kan je de kleuren slechts per 8 karakters definiëren en zijn verschuivingen op het scherm slechts mogelijk per 8 pixels horizontaal en/of vertikaal. Je kan dus wel "springen" maar niet "schuiven". Binnen deze beperkingen geldt anderzijds het grote voordeel dat het programmeren even eenvoudig is als in SCREEN0.
Het spelletje "VLO" is bij voorbeeld gebaseerd op 2 eenvoudige principes: karakters op het scherm brengen en kleur geven (telkens door de gegevens in het video-geheugen te poken).

1. Karakters op het scherm

Het basisadres voor wat in SCREEN1 op het scherm komt, is 6144. Dit wil zeggen, dat als er op dat adres een ASCII-code geschreven wordt, het overeenkomstige karakter bovenaan links op het scherm verschijnt.
Per regel zijn er 32 plaatsen voorzien (ook al is een kleinere breedte ingesteld). Om dus vlot een plaats op het scherm te berekenen, gebruiken we de formule: 6144 + X-as + (Y-as * 32). Vergeleken met de instructie LOCATE, ziet dat er dan zo uit: LOCATE 7,8 wordt VPOKE 6144+7+8*32. Daarachter zetten we dan een komma en de ASCII-code van het gewenste karakter.

2. Kleuren

Voor kleur in SCREEN1 is het basisadres 8192 en per adres worden er 8 karakters ingekleurd. "VPOKE 8192, ..." verandert aldus de karakters met de ASCII-waarden 0 t/m 7. Wil je bij voorbeeld alle letters en cijfers een andere kleur geven, dan vul je de adressen 8198 t/m 8207 in: die komen overeen met de ASCII-codes 48 tot 125.
Die kleuren worden geformuleerd in een binair getal van 8 tekens: de eerste vier slaan op de voorgrond en de volgende vier de achtergrond. Vb. VPOKE 8192,&B00010000 resulteert in een zwarte voorgrond op een doorschijnende achtergrond : binair 0001 = decimaal 1 (kleur zwart) en binair 0000 = decimaal 0 (kleur doorschijnend). De kleurinstellingen kunnen weer ongedaan gemaakt worden met de instructie COLOR.

J. Simal

FOUT.BAS

FOUT	BAS	D1

Dit basicprogramma moet éénmaal worden 'gerund' en mag dan uit het geheugen worden gewist. Hierna zullen bijna alle fouten, d.w.z. de fouten die via de 'hook' gaan, in het Nederlands worden aangeduid.
(Schepers Rohnny)

CHESS.BAS

CHESS	BAS	D1

Bij gebrek aan een schaakbord kan dit programma voor u een oplossing betekenen. Het schaakspel wordt gespeeld met twee spelers.
(Roels Guy)

VERKIEZ.BAS

VERKIEZ	BAS	D1

De Belgische gemeenteraadsverkiezingen staan weer voor de deur (oktober 1988) en dit programma berekent voor u de zetelverdeling van de verschillende partijen.
(Dieltjens Willy)

MOTOR.BAS

MOTOR	BAS	D1

Dit programma laat op een grafische manier de werking zien van de 4-slag en 2-slag motor.
(Moreau Eddy)

PACMAN.HOF

PACMAN	HOF	D1

Een basic pacman spel. Snel reageren en veel eten !
(Hofmans Kurt)

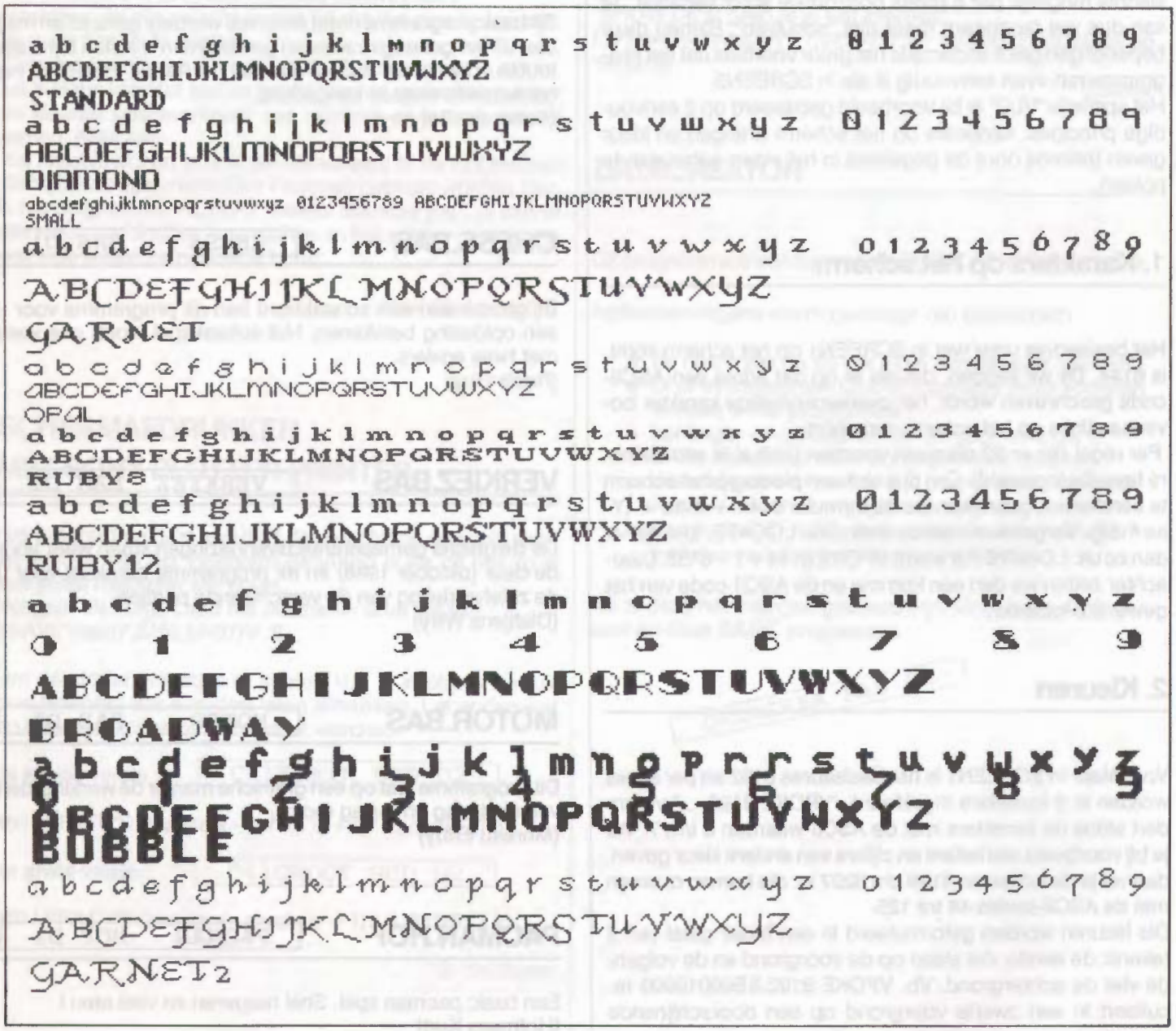
EXTRA FONTS voor DYNAMIC PUBLISHER

DYNAMIC PUBLISHER is bij de MSX-2 gebruikers een veel gebruikt programma. Blader maar even door dit tijdschrift en je kan merken dat verschillende bijdragen (artikelen, schema's en advertenties) hiermee ontworpen zijn.

Voor de redactie een grote luxe : even de LASER-printer aan de MSX en een paar minuutjes later rolt een knalzwart plaatje uit de schuif...

Deze keer hebben we voor onze diskabonnees (naast **MSX-NEWS**) nog een leuke verrassing : schijf nummer 20 bevat de files met onderstaande fonts. U vindt ze terug met de extensie .FNT en ze kunnen op de gebruikelijke manier geladen worden. Ze worden U vriendelijk aangeboden door onze COCON-vrienden (zie ook de adressentabel in dit nummer)

Veel grafisch plezier !



a b c d e f g h i j k l m n o p q r s t u v w x y z
0 1 2 3 4 5 6 7 8 9
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
GOTH I

a b c d e f g h i j k l m n o p q r s t u v w x y z
0 1 2 3 4 5 6 7 8 9
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
I T A L I C

a b c d e f g h i j k l m n o p q r s t u v w x y z
0 1 2 3 4 5 6 7 8 9
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
L I T A L I C

a b c d e f g h i j k l m n o p q r s t u v w x y z 0 1 2 3 4 5 6 7 8 9
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
S C H R I F T

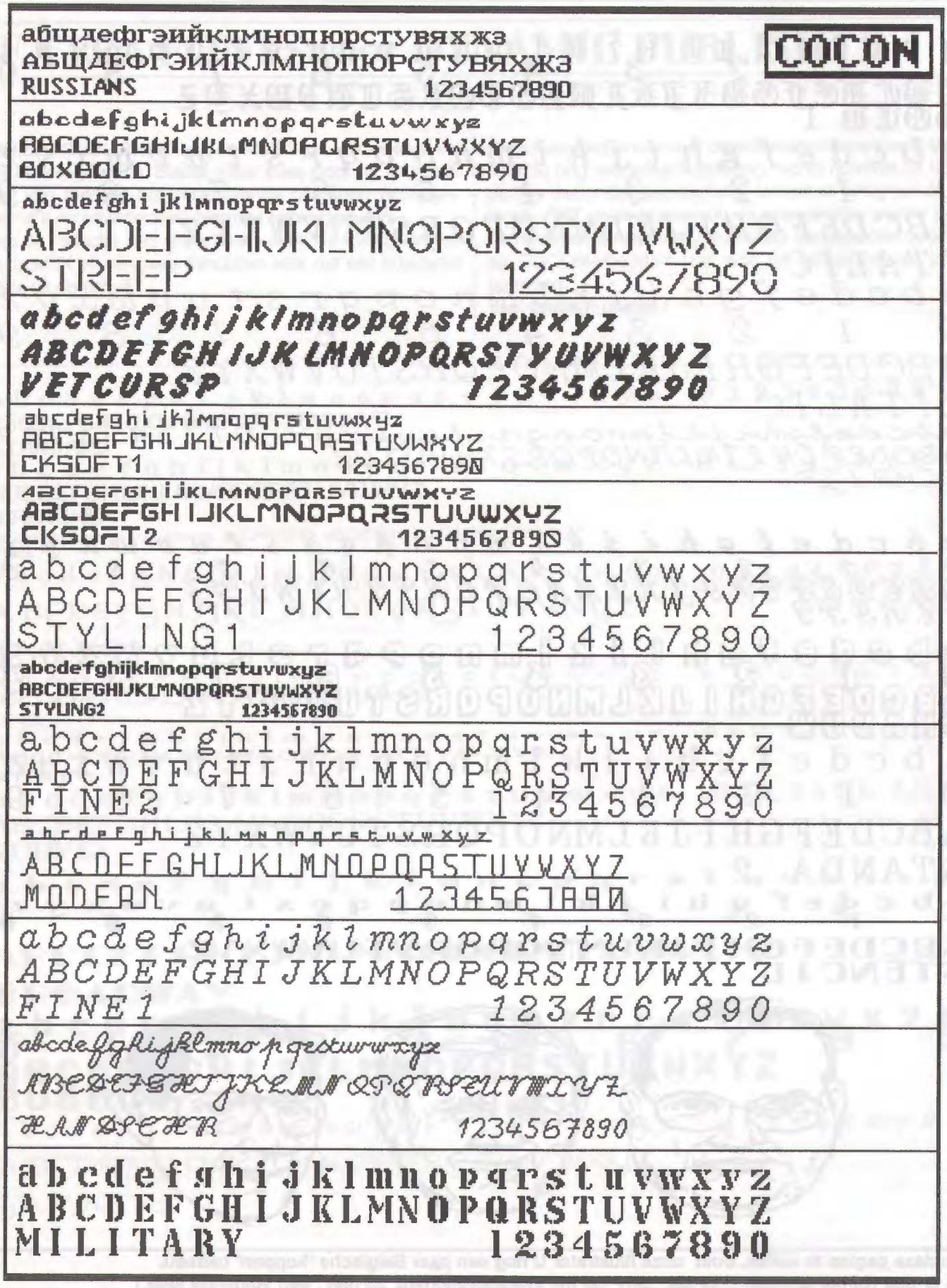
a b c d e f g h i j k l m n o p q r s t u v w x y z
0 1 2 3 4 5 6 7 8 9
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

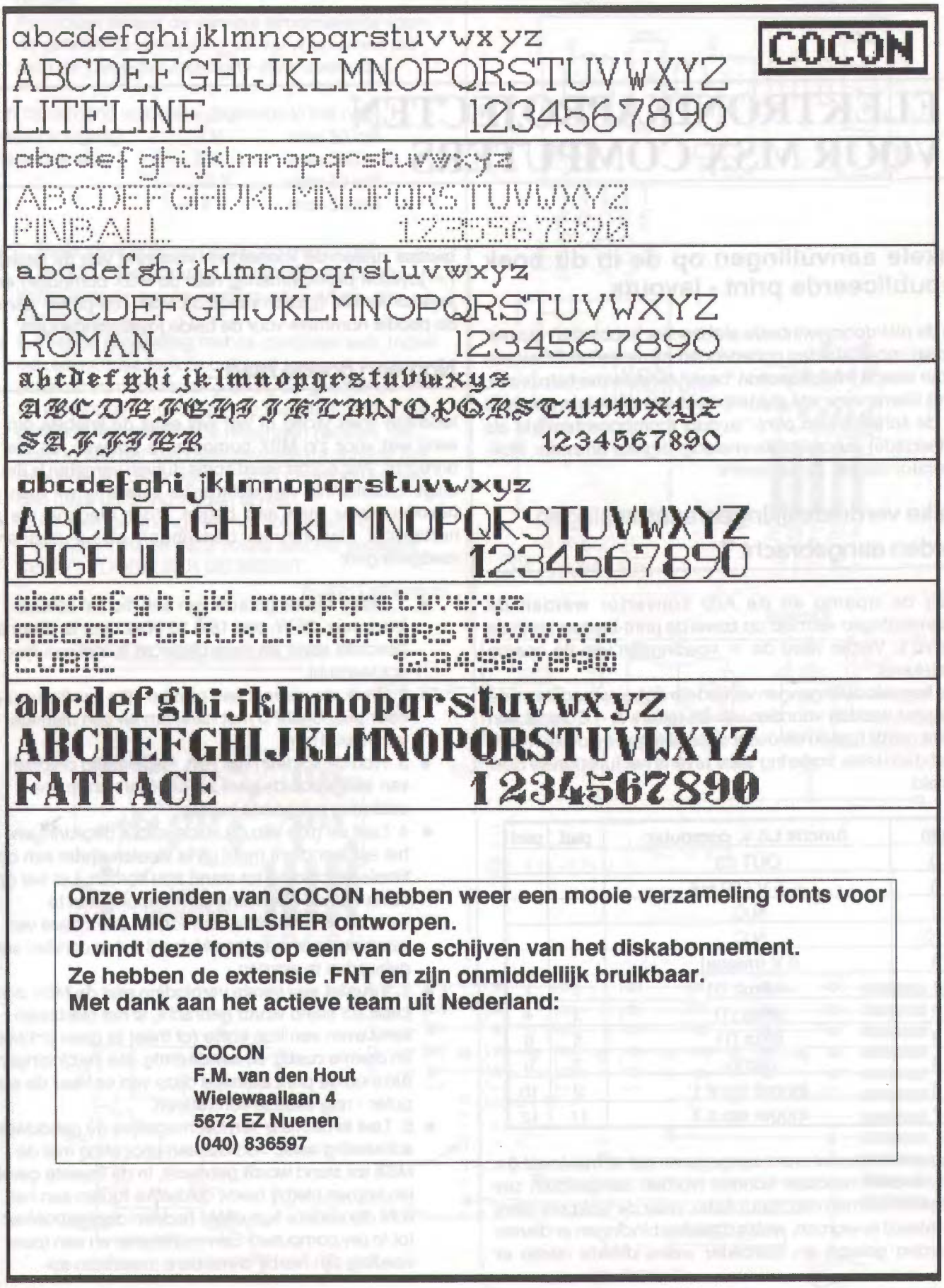
a b c d e f g h i j k l m n o p q r s t u v w x y z
0 1 2 3 4 5 6 7 8 9
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
S H A D O W

a b c d e f g h i j k l m n o p q r s t u v w x y z
0 1 2 3 4 5 6 7 8 9
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
S T A N D A 2

a b c d e f g h i j k l m n o p q r s t u v w x y z
0 1 2 3 4 5 6 7 8 9
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
S T E N C I L

Om deze pagina te vullen, doet onze illustrator U nog een paar Belgische "koppen" cadeau.
Als er nog ruimte is, vindt U ze als *.STP op het diskabonnement, zo niet : een volgende keer !


абщдефгэийклмнопюрстувяхжз
АБЩДЕФГЭИЙКЛМНОПЮРСТУВЯХЖЗ
RUSSIANS 1234567890
COCON
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
BOXBOLD 1234567890
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
STREEP 1234567890
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
VETCURSP 1234567890
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
CKSOFT1 1234567890
ABCDEFGHIJKLMNOPQRSTUVWXYZ
ABCDEFGHIJKLMNOPQRSTUVWXYZ
CKSOFT2 1234567890
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
STYLING1 1234567890
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
STYLING2 1234567890
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
FINE2 1234567890
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
MODERN 1234567890
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
FINE1 1234567890
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
SCRIPT 1234567890
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
MILITARY 1234567890



Onze vrienden van COCON hebben weer een mooie verzameling fonts voor DYNAMIC PUBLILSHER ontworpen.
U vindt deze fonts op een van de schijven van het diskabonnement.
Ze hebben de extensie .FNT en zijn onmiddellijk bruikbaar.
Met dank aan het actieve team uit Nederland:

COCON
F.M. van den Hout
Wielewaallaan 4
5672 EZ Nuenen
(040) 836597

ELEKTRONIKAPROJECTEN VOOR MSX COMPUTERS

Enkele aanvullingen op de in dit boek gepubliceerde print - layouts

Om de niet doorgewinterde elektronika hobbyisten die hier en daar moeilijkheden ondervinden bij de samenbouw van elk der steeds wederkerende "basis" modules ter hulp te zijn, wordt hierna voor alle duidelijkheid een overzicht gegeven van de schema's en print- layouts (componentenzijde als soldeerzijde) van zowel de analoog- digitaal omzetter, klokgenerator als van de basisprint.

Welke verduidelijkingen/aanpassingen werden aangebracht ?

1. Bij de opamp en de A/D konvertor werden de pinaansluitingen vermeld op zowel de print-layouts als in de schema's. Verder werd de + voedingslijn van de opamp bijgetekend.
2. De konnektor-uitgangen van iedere opbouwmodule en de basisprint werden voorzien van de letters A -- E zodat een direkte relatie tussen elk dezer aansluitingen duidelijk wordt. Naast deze letter kodering staat tevens het funktionele doel vermeld.

pin	functie t.o.v. computer	pad	pad
8	OUT (0)		
5	+ 5 V / 50 mA		
	N.C.		
	N.C.		
9	0 V (massa)		
1	vooruit (1)	1	1
2	terug (1)	3	4
3	links (1)	5	6
4	rechts	7	8
6	trigger input 1	9	10
7	trigger input 2	11	12

3. Voor de basisprint werd aangegeven dat er maximaal 6 x A/D konvertor modules kunnen worden aangesloten per basisprint tesamen met hun lokatie, waar de klokprint dient gemonteerd te worden, welke draadverbindingen er dienen te worden gelegd, en hieronder welke direkte relatie er bestaat tussen de konnektornummering van de basisprint (= joystick pin-nummering naar de MSX computer) en de joystick funktie (gezien vanuit de MSX computer) alsmede de paddle nummers voor de beide joystickingangen.

Nog een goede raad....

Iedereen voelt vroeg of laat wel eens de kriebels om ook eens wat voor z'n MSX computer te bouwen, en niet ten onrechte. Wat echter velen soms durven vergeten is dat wel enige basiskennis van elektronika vereist is wil men nog beseffen waar men aan begint. Voor diegene die hun hartstocht niet kunnen bedwingen alvast nog enige raadgevingen:

- 1. Maak steeds gebruik van een "lichte" soldeerbout:max. 25 W. een 15 à 20 Watt bout is uitermate geschikt zeker als deze uitgerust is met een fijne soldeerpunt.
- 2. Werk steeds met een goed en fijn kwaliteitssoldeer bvb. Sn60Pb met harskern en een diameter van max. 1mm
- 3. Hou de soldeerpunt rein. Regelmatig ontdoen van aangebrande hars zal de kwaliteit van de soldering ten goede komen.
- 4. Laat de hitte van de soldeerbout doordringen: het soldeer dient mooi uit te vloeien opdat een optimale verbinding tot stand zou komen. Let wel op, het is niet de bedoeling de komponenten te verbranden.Om dit te voorkomen dient deze verwarming voor max. een kleine 3 à 4 seconden aangehouden te worden.
- 5. Voordat een eerste verbinding met de MSX computer tot stand wordt gebracht, is het raadzaam eerst even een kop koffie (of thee) te gaan drinken en daarna **rustig** en **aandachtig** alle verbindingen - deze op de print alsmede deze van en naar de computer - nog even te kontroleren.
- 6. Test in de mate van het mogelijke de gebouwde schakeling eerst, voordat een koppeling met de MSX tot stand wordt gebracht. In de meeste gevallen komen hierbij reeds duidelijke fouten aan het licht die anders hun effekt hadden doorgetrokken tot in uw computer ! Een multimeter en een losse voeding zijn hierbij onmisbare meet/labo-ap-

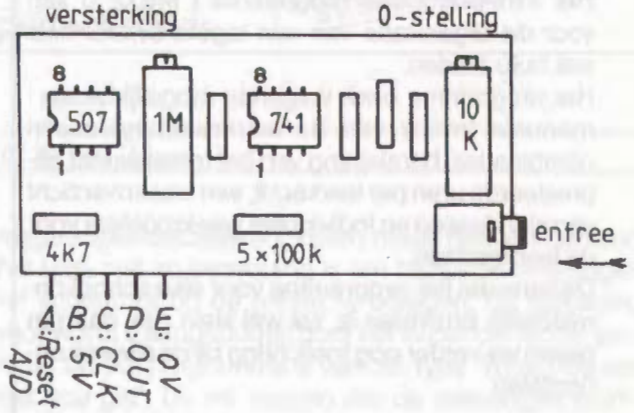
paraten.
Kontroleer tevens de vereiste stroomsterkte voor de gebouwde schakeling; dit zal voorkomen dat men de komputervoeding te sterk overbelast.

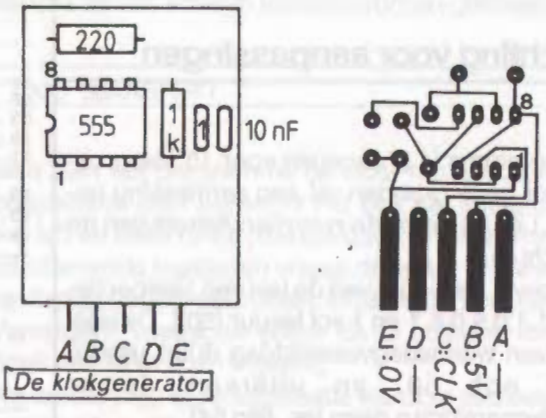
Hou in dit verband volgende gegevens in het oog:

Per joystick ingang:	5 V	max 50 mA
Per cartridge slot :	5 V	max 300 mA
	12 V	max 50 mA
	- 12 V	max 50 mA

- 7. Maak steeds de verbindingen met UITGESCHAKELDE computer, dit vermijdt eventuele kortsluitingen van de voeding en/of adres- en datalijnen.
- 8. Test de schakeling met de computer aan. Indien een abnormale funktie optreedt, schakel de computer uit en onderzoek grondig de gebouwde schakeling(en) op fouten. De meeste MSX2 computers zullen gebufferde uitgangen hebben, hetgeen in één der slechtste gevallen de schade kan beperken tot deze buffers; MSX1 computers hebben deze buffering zelden of niet .
- 9. Als slotopmerking nog het volgende: WEES VOORZICHTIG en denk vooral aan het gezegde BEZINT VOORALEER GE BEGINT...

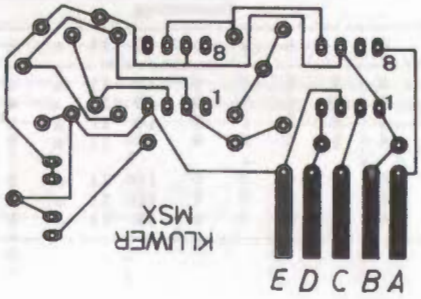
Guido Lallemand

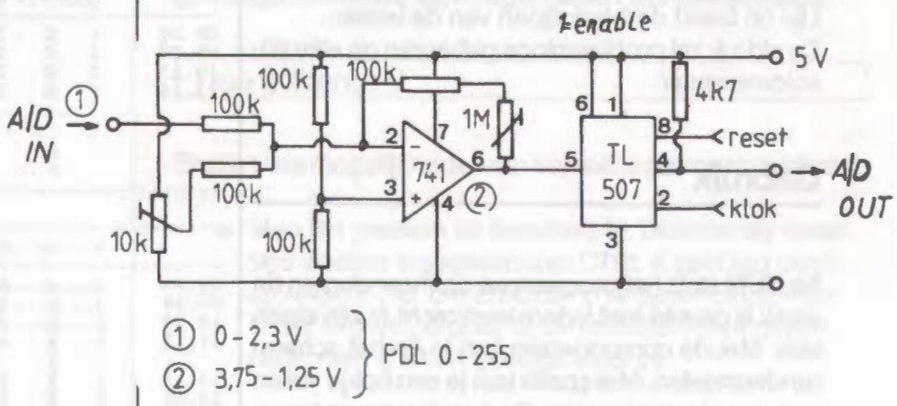


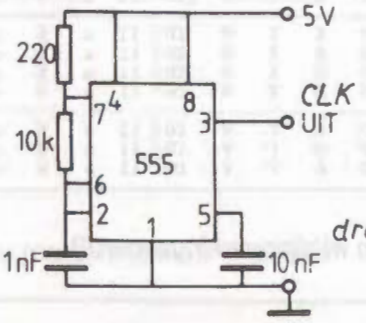


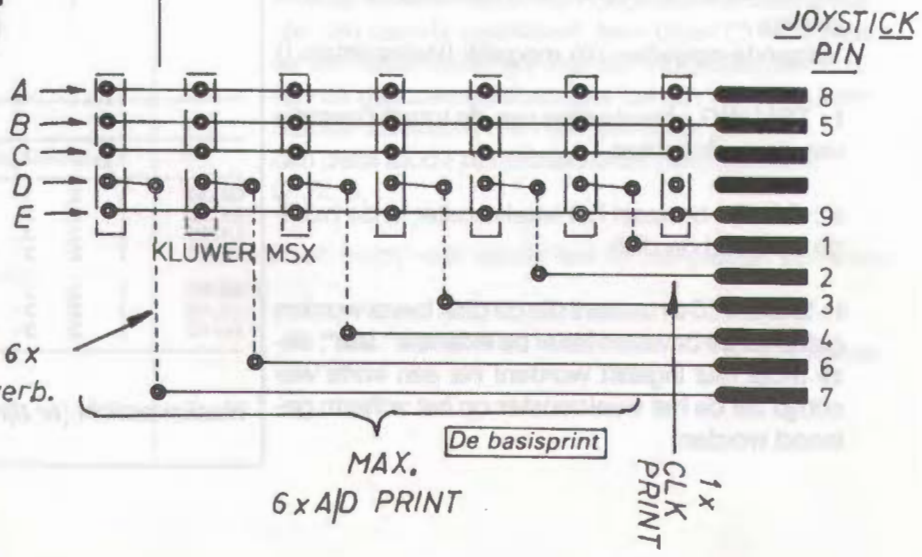
De klokgenerator

De analoog/digitaal-omzetter









UURROOS BAS D1

Het mini-uurrooster programma (MSX2 !) kan voor de organisatie van een lagere school heel wat hulp bieden.
Het programma biedt volgende mogelijkheden : manuele invoer van de leerkrachten/klassen combinaties, berekening van het totaal aantal gepresteerde uren per leerkacht, een weekoverzicht van alle klassen en individuele weekroosters voor de leerkrachten.
De kans dat het programma voor elke school onmiddellijk bruikbaar is, zal wel klein zijn, daarom geven we verder nog toelichting bij de diverse onderdelen.

Toelichting voor aanpassingen

Het programma is ontworpen voor 15 klasgroepen, voor meer groepen zal een aanpassing nodig zijn. Lijn 50 bevat de nummers/letters van de leerkrachten.
Lijn 53 bevat de lengte van de lesuren : respectievelijk 1,1,1,0.5,0.5,1 en 1 vol lesuur (50'). De laatste les van woensdagvoormiddag duurt uitzonderlijk ook 50' en uiteraard is er woensdagnamiddag geen les. (lijn 54)
Lijn 56 bevat de begintijden van de lessen.
De afdruk zal probleemloos gebeuren op elke 80-kolomsprinter.

Gebruik

Na RUN start het programma op maandag en de week is gevuld met iedere leerkracht in zijn eigen klas. Met de cursortoetsen kan je op het scherm rondwandelen. Met spatie kan je een hokje vullen met een nieuwe waarde. De functietoetsen tonen telkens de DAG-rooster van maandag tot en met vrijdag.
Volgende operaties zijn mogelijk (kleine letters !)

t : TELLING : berekening van de totaal-prestatie van de leerkrachten

s : SAVE : bewaart het weekrooster in de huidige toestand op disk.

l : LADEN : De roosters die de disk bevat worden getoond, ze bevatten ieder de extensie **".uur"**, deze moet niet ingetikt worden! Na een korte wachttijd zal de het weekrooster op het scherm getoond worden.

uurrooster 1987 - 1988

maandag

	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
08.30	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
09.20	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
10.25	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
11.15	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
13.15	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
13.40	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
14.45	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e

dinsdag

	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
08.30	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
09.20	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
10.25	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
11.15	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
13.15	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
13.40	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
14.45	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e

woensdag

	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
08.30	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
09.20	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
10.25	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
11.15	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e

donderdag

	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
08.30	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
09.20	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
10.25	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
11.15	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
13.15	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
13.40	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
14.45	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e

vrijdag

	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
08.30	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
09.20	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
10.25	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
11.15	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
13.15	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
13.40	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e
14.45	1	2	3	4	5	6	7	9	10	11	a	b	c	d	e

Weekoverzicht (er zijn geen wijzigingen ingevoerd!)

i : Individueel overzicht : afdruk van het weekrooster voor iedere leerkracht.

o : Overzicht : afdruk van het totale weekrooster.

```
-------------------------------------------------------
                    uurrooster 1
-------------------------------------------------------
        maandag   dinsdag   woensdag  donderdag vrijdag
-------------------------------------------------------
08.30       1         1         1         1         1
09.20       1         1         1         1         1
10.25       1         1         1         1         1
11.15       1         1         1         1         1
-------------------------------------------------------
13.15       1         1                   1         1
13.40       1         1                   1         1
14.45       1         1                   1         1

-------------------------------------------------------
                    uurrooster 2
-------------------------------------------------------
        maandag   dinsdag   woensdag  donderdag vrijdag
-------------------------------------------------------
08.30       2         2         2         2         2
09.20       2         2         2         2         2
10.25       2         2         2         2         2
11.15       2         2         2         2         2
-------------------------------------------------------
13.15       2         2                   2         2
13.40       2         2                   2         2
14.45       2         2                   2         2


-------------------------------------------------------
                    uurrooster 3
-------------------------------------------------------
        maandag   dinsdag   woensdag  donderdag vrijdag
-------------------------------------------------------
08.30       3         3         3         3         3
09.20       3         3         3         3         3
10.25       3         3         3         3         3
11.15       3         3         3         3         3
-------------------------------------------------------
13.15       3         3                   3         3
13.40       3         3                   3         3
14.45       3         3                   3         3


-------------------------------------------------------
                    uurrooster 4
-------------------------------------------------------
        maandag   dinsdag   woensdag  donderdag vrijdag
-------------------------------------------------------
08.30       4         4         4         4         4
09.20       4         4         4         4         4
10.25       4         4         4         4         4
11.15       4         4         4         4         4
-------------------------------------------------------
13.15       4         4                   4         4
13.40       4         4                   4         4
14.45       4         4                   4         4


-------------------------------------------------------
                    uurrooster 5
-------------------------------------------------------
        maandag   dinsdag   woensdag  donderdag vrijdag
-------------------------------------------------------
08.30       5         5         5         5         5
09.20       5         5         5         5         5
10.25       5         5         5         5         5
11.15       5         5         5         5         5
-------------------------------------------------------
13.15       5         5                   5         5
13.40       5         5                   5         5
14.45       5         5                   5         5
```

Individuele roosters (default waarden)

Superfont editor

SFEDIT BAS D2

Wat is het ?

Sommige superfontbezitters zullen reeds hebben gemerkt dat het toch niet zo eenvoudig is om bepaalde tekeningen op het scherm of op de printer te brengen. De oplossing hiervoor wordt u aangeboden door het Superfont editor programma. Dit editprogramma is van het type 'What you see is what you get'. Dit wil zeggen dat de tekeningen rechtstreeks op het scherm kunnen worden getekend.

Hoe opstarten ?

Men start het programma op door run "sfedit.bas". Als het programma start verwacht het ook het superfontprogramma (sf) en twee fonts (standard.fon en body.fon). Wanneer dit allemaal is ingeladen vraagt de editor of u een alternatieve display font wil gebruiken. Indien niet dan drukt u [return]. Vervolgens vraagt de editor de te editeren font (de '.fon' hoeft niet te worden ingetikt).
Als laatste stap in de initialisatie wordt er een bestandsnaam gevraagd waarin de tekeningen zullen worden bewaard.

Hoe editeren ?

Er zijn twee mogelijkheden om bepaalde patronen te klezen.

- Men tikt gewoon de karakters in, de kontrole karakters worden ingegeven met CTRL A gevolgd door @,A-Z,1-5 (vb. om chr$(03) in te geven drukt men CTRL A gevolgd door C, om chr$(27) in te geven drukt men CTRL A gevolgd door 1).

- Indien het ingeven van kontrolekarakters u nogal moeilijk lijkt bestaat er een gebruiksvriendelijke manier om deze te selekteren. Men drukt CTRL B waarna een tabel verschijnt met alle mogelijke patronen van de gekozen editfont. Met behulp van de cursortoetsen selekteert men het gewenste patroon. Om deze mode te verlaten drukt men opnieuw CTRL B.

We wensen u nog veel plezier toe bij het gebruik van deze editor.

W. Coremans

De programma's bij het ETIKETTEN-programma vindt U op disk 1

ETIKETTENLABELS PRINTEN

Diskettes vol spelletjes plaatsen is meestal voor ons gebruikers niet zo'n geweldig probleem. Als je dan echter een hele serie van diskettes hebt, dan wordt het vaak wat moeilijk om uit te vinden op welke diskette juist dat ene leuke utility staat wat we willen hebben.
Voelt u zich aangesproken? Dan is hier een programma wat bij u past. Enig probleem daarbij is dat u dan wel een echte MSX-printer dient te bezitten, anders gaat het niet, want het programma is geheel ingericht voor de Philips serie vanaf VW0030 en verder tot de NMS 1431.
Diskettenlabels zijn er in allerlei uitvoeringen. De hier bedoelde labels zijn in de diverse computershops op banderol te verkrijgen. Meestal zijn dit er 100. Natuurlijk kunnen ook de labels gebruikt worden welke bij de aangekochte diskettes worden meegeleverd. Dit is dan weliswaar iets lastiger, maar het gaat ook. De labels op banderol worden op kettingrol geleverd en dat werkt toch gemakkelijker.

HET PROGRAMMA bestaat uit 3 gedeelten.
1e deel is AUTOEXEC.BAS(staat als ETIKET.LDR op het diskabonnement)
2e deel is ETIKET.PRG
3e deel is PRINT.BIN

Beginnen we met deel 3. Dit gedeelte wordt genoemd INST-HEX.BAS. Met INST-HEX.BAS wordt uiteindelijk het machinetaal programma PRINT.BIN gemaakt. Het programma INST-HEX.bas intikken en daarbij vooral goed opletten op tikfouten en let daarbij vooral op de dataregels. Als dit gedeelte is ingetikt dan vooral NIET RUNnen. Eerst saven op disk. Vervolgens kan het programma geRUNned worden. Gebruik in ieder geval een nieuwe diskette om de en als het programma na run goed zonder fouten doorgaat, dan kan men veronderstellen dat een en ander juist is. Dit kunnen we kontroleren op de volgende manier.
DEFUSR = &HF501:X = usr(0) Wel dient de printer ingeschakeld te zijn. Er wordt nu een afdruk gemaakt. Ook al hebt u schermmode 80 ingeschakeld, toch wordt op papier slechts 40 kolommen afgedrukt.
Probeer het volgende :
POKE&HF511,(N) Voor N mag het getal 1 tot 21 ingegeven worden. Daarmee wordt het aantal regels geprint.
POKE&HF514,&H50 Nu wordt op papier ook 80 koloms afgedrukt. Om af te drukken dient u uiteraard X = USR(0) In te typen en dan return te geven.

ETIKET.PRG

Dit gedeelte van het programma is het besturingsprogramma. Hier wordt steeds vanuit een MENU gewerkt. Dit gedeelte is ingericht voor gebruik met een MSX 2. Hebt u mogelijk nog een MSX 1 dan is het noodzakelijk om de regels 120 tot en met 290 aan te passen. Regel 480 kan dan vervallen. Verander in regel 1230 mijn naam in uw eigen naam dan komt uw naam op uw diskette staan zodat u die altijd kunt herkennen.

AUTOEXEC.BAS is volkomen duidelijk voor wat de funktie betreft. Dit gedeelte zorgt ervoor dat het programma automatisch kan opstarten na bijvoorbeeld een reset van de computer, of bij het opnieuw inschakelen.

DE WERKING VAN HET PROGRAMMA

Zoals reeds gezegd werkt u steeds vanuit een menu. Als eerste kiest u voor de optie 1 (regels tellen). Er verschijnt nu een tekst die een uitleg geeft over het gebruik van de toevoeging in de naam die we SOORTNAAM noemen. Bijvoorbeeld .LDR of .BAS alle soortnamen kunnen gebruikt worden, maar u kunt steeds slechts een soortnaam invoeren. Het is dus verstandig om bij het inrichten van uw diskettes hier rekening mee te houden. Nu verschijnen de files met de gewenste soortnaam op het scherm. Tel deze files en deel het gevonden aantal door twee. Rond het gevonden getal, indien nodig, naar boven af. Bijvoorbeeld 7.5 wordt 8. Druk op return en u gaat terug naar het menu. Nu kiest u voor de optie 3. Er wordt gevraagd om ingave van een diskette nummer. Dit getal mag uit drie cijfers bestaan. Bijvoorbeeld 001 voor diskette 1. Nu wordt gevraagd naar het aantal te printen regels. Vul hier het gevonden aantal regels in wat we straks bij optie 1 hebben gevonden. LET OP!!!!! Dit dient HEXADECIMAAL te gebeuren. 1 t/m 9 kan gewoon normaal; 10 wordt echter A, 11 wordt B, 12 wordt C, 13 wordt D, 14 wordt E, 15 wordt F, 16 wordt 11 enzovoorts. Bij optie 2 wordt alleen het MSX-logo plus diskette nummer en de naam geprint. Daardoor blijft het mogelijk om de diskinhoud op een andere manier in te vullen. Optie 4 hoeft verder geen betoog.

TENSLOTTE : Boven is gezegd dat het programma speciaal is ingericht voor de Philips MSX printer. Mocht u met een andere printer werken en u wilt het programma daarvoor ombouwen dan is voorzien in uitleg bij de verschillende printerinstrukties. U zult dan in uw eigen handleiding moeten zoeken naar een vervangende instruktie. Hebt u een aanpassing gevonden die bij u ook werkt, stuur ons hiervan dan een copy. Wij kunnen daarmee dan ook weer andere gebruikers helpen. Daarvoor alvast bij voorbaat dank.

AANPASSING ETIKETTENLABELS Voor STAR NL-10

In grote lijnen is de werking van het programma hetzelfde gebleven. In de 2 eerste autostart programma's die ik "**ETIKET.BAS**" en "**ETIKET1.BAS**" heb genoemd, zitten slechts minieme verschillen. Het program "**ETIKET.BAS**" is voor normale afdruk.
Op regel 150 wordt het andere deel ingeladen dat ik "**ETIKET.PRG**" heb genoemd, verder heb ik er mijn naam ed. aan toegevoegd (regels 121-122).

Het programma "**ETIKET.PRG**" heeft meerdere aanpassingen. Allereerst heb ik er een optie aan toegevoegd zodat men meerdere bestanden kan afdrukken op dezelfde label. Dit zonder het terug afdrukken van het MSX-embleem of diskette nummer, dit om plaats in te winnen op de label, en omdat er volgens mij op de disketjes die bij de mede hobbyisten ronddwalen er meestal een heleboel bestanden of bestandjes een plaatsje hebben gevonden. Het komt er slechts op aan dat de gebruiker zelf een logische indeling en hoeveelheid van bestanden op zijn schijf bewaart.
In de plaats van het logo grafisch te laten afdrukken, heb ik een gemakkelijkere weg gekozen.
Mede doordat de **STAR NL-10** op eenvoudige wijze dubbele hoogte tot zelfs vierdubbele hoogten kan afdrukken, met behulp van een paar simpele kodes.
Daardoor zijn de grafische regels, de data regels, en routines verdwenen uit het oorspronkelijke programma. Het is misschien niet zo mooi als het officiële logo, maar het is in elk geval aanvaardbaar.
Op regel 1200 wordt een kode gebruikt om de printer een regel te laten terugnemen. Dit is misschien niet de beste oplossing, maar voor mij was dit het simpelste. Het voornaamste doel is een programma te verkrijgen dat werkt. Ook geldt dit voor de door mij aangemaakte optie, ze zal zeker geen voorbeeld zijn van goede programmering maar het werkt zonder gebreken, dit geldt uiteraard ook voor de 2de aanpassing.
In elk geval hoop ik er meerdere mensen een dienst mee te bewijzen. Het programma "**PRINT.BIN**" is ongewijzigd gebleven !
Het aanpas-programma heeft dezelfde werking als het voorgaande, behalve dat er nu voor een nog kleiner lettertype is gekozen, om nog meerdere files te kunnen afdrukken op de label. Het MSX-embleem en diskette nummer is hetzelfde gebleven. Opstarten met "**ETIKET1.BAS**", het 2de deel dat zichzelf laadt noemt "**ETIKETKL.EIN**".

MASTERMI.M16

`MASTERMI M16 D1`

Een prachtige mastermind voor de MSX 2 gebruiker. De nodige uitleg wordt in het programma gegeven en er is zelfs een demo aanwezig.
(Coremans Raf)

FANTASIE.M16

`FANTASIE M16 D1`

Dit is een grafisch programma dat heel wat fantasietjes op het scherm tovert.
(Vandermeulen Kris)

UURROOS.BAS

`UURROOS BAS D1`

Een programma dat vooral is geschikt voor de leraars onder onze leden. U kan uurroosters samenstellen en deze laten afdrukken met de printer.
De mogelijkheid wordt geboden om de gegevens te bewaren en te laden.
(Hermans Wilfried)

WO3.BAS

`WO3 BAS D1`

De uiteindelijke bedoelingen van de auteur van het programma wereldoorlog 3 zijn heel wat vredelievender dan de titel laat vermoeden. Uw taak in dit spelletje, dat wordt gespeeld met behulp van een joystick, is namelijk het beletten van een derde wereldoorlog.
(Roels Guy)

BANNERB.BAS

`BANNERB BAS D1`

Met behulp van dit programma, dat samen met de printer moet worden gebruikt, kan u een sliert van grote letters (woorden, zinnen ....) afdrukken.
Een meterslange slogan kan zo uit uw printer rollen.
(Hermans Wilfried)

BELGIANS.BAS

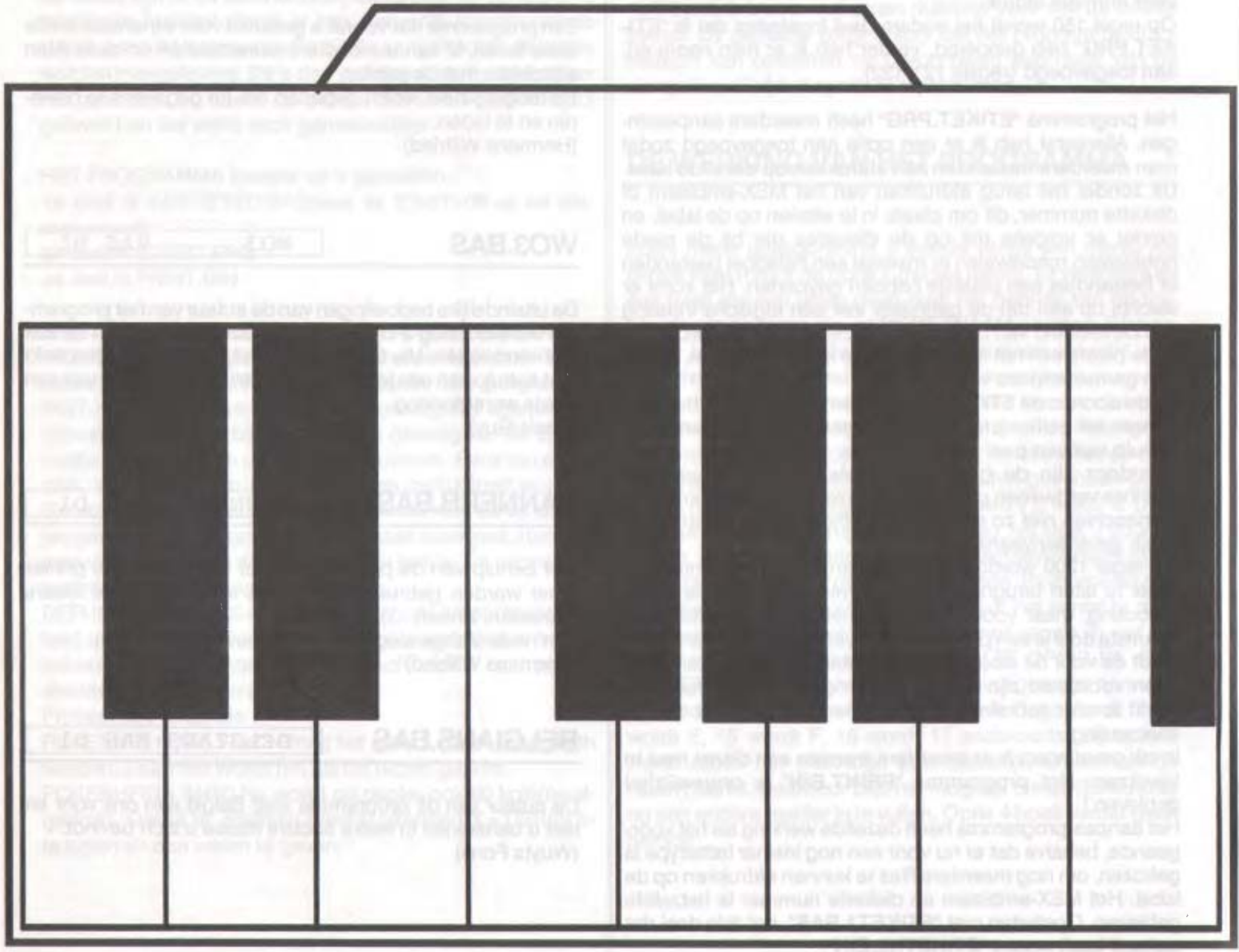
`BELGIANS BAS D1`

De auteur van dit programma stelt België aan ons voor en laat u berekenen in welke sociale klasse u zich bevindt.
(Wuyts Fons)

NOTENLES (MSX-2)

NOTENLES BAS D2

Dit programma is gegroeid uit bittere noodzaak : de juf muziek had me vriendelijk opgedragen om dagelijks met de kinderen een kwartiertje "melodisch dictee" te oefenen.
Met het grafisch tablet en dit programma dacht ik hier aan te ontsnappen.
Eerst moeten we wel het grafisch tablet transformeren in een eenvoudig klavier. Onderstaande tekening kan U overnemen op een velletje karton, netjes uitknippen langs de rand en de constructie past precies als inlegvel op het G.T.
De juf was wel zo vriendelijk om enige aanwijzingen te geven : het eerste menu geeft U de keuze uit de volgende type oefeningen : **E-G** (mi-sol), **E-G-A** (mi-sol-la), **C-E-G-A** (do-mi-sol-la) en **C-D-E-F-G-A** (van do tot en met la).
Dan wordt er gevraagd naar de lengte van de reeks : hoeveel noten mag 1 reeks aanbieden (kies met CURSOR-toetsen en bevestig met spatiebalk).
Daarna kiest U op dezelfde manier het aantal oefeningen.
Draai even aan de volumeknop van de monitor en de eerste reeks wordt aangeboden.
Daarna dient deze melodie in **hetzelfde tempo** op het klavier nagespeeld te worden.
Bij een foute poging wordt het melodietje nogmaals voorgespeeld en als de tweede poging nog niet lukt, dan wordt het melodietje afgebeeld op een heuse notenbalk.
Zelfs met een minimale kennis van notenleer moet het dan wel lukken ! *Succes met de studie !*



De MSX-computer biedt vele editing mogelijkheden en het valt mij op dat maar weinigen de geboden mogelijkheden gebruiken. Aan dit laatste wil ik graag wat veranderen, want meestal is het uit onwetendheid dat we de mogelijkheden van onze MSX-computer niet ten volle benutten.

Vele standaarden

Het is niet zo simpel om alle edit mogelijkheden die de MSX standaard biedt even op een rijtje te zetten. De MSX is een computer die aan vele standaarden wil voldoen en dus ook van al die standaarden iets meegekregen heeft. We kunnen dan ook de edit mogelijkheden op verschillende niveaus bekijken. Ik laat in willekeurige volgorde passeren :

- De screen editor die we aantreffen als we de machine aanzetten.

- De editor die alleen in programma's gebruikt kan worden volgens de zogenaamde VT-52 protocollen. Dit zijn afspraken over de betekenis van de diverse 'ASCII' waarden zoals gebruikt in terminals.

- Verder hebben we nog de edit mogelijkheden die in de verschillende bestuurssystemen gebruikt worden. De bekendste hiervan is natuurlijk het MSX-DOS, maar ook CP/M kan op MSX gedraaid worden.

- Tot slot hebben we dan nog de programmeerbare toetsen. In een programma kunnen dit alle toetsen zijn en in direct mode alleen de functietoetsen.

Afdwalen of rondzwerven ?

En duizelt het u al ? Laten we maar beginnen en zien waar we eindigen. Meestal dwaal ik in mijn omzwervingen behoorlijk af van mijn zelf gekozen onderwerp, maar dat is juist het leuke. (in ieder geval voor mij !) Als je eenmaal op ontdekkingstocht bent in de machine ontdek je steeds weer meer bedoelde en onbedoelde bijzonderheden. Maar ik kom wel altijd weer terug bij mijn onderwerp en eventueel een of meer nummers later zijn alle zaken die het onderwerp betreffen toch wel de revue gepasseerd.

Eenvoudige start

Het meest voor de hand liggend is om te beginnen met de direct mode. We zetten de machine aan en demonstreren in basic een aantal zaken die bijna iedereen al bekend zullen zijn.

Ik geef eerst een simpel voorbeeld :
Op school zie ik in de lessen computerkunde vaak dat leerlingen een programma starten door 'run' in te tikken gevolgd door [RETURN]; terwijl de functietoets [F5] standaard hetzelfde effect heeft. Dit geldt ook voor 'list' en [F4].

[F5] versus [F10]

Wel heel simpel zullen velen zeggen, maar dat wist ik al lang. Toch denk ik dat een aantal onder de lezers nog niet weten dat ook de [F10] een 'run' opdracht heeft. En onder degenen die dat wel wisten zijn er velen die denken dat die extra 'run' er alleen is omdat de 'run' zo belangrijk is, dat de ontwerpers van de MSX die opdracht zowel zonder als met [SHIFT] onder de meest rechter functietoets wilden hebben. In vele gevallen is het inderdaad zo dat het gebruik van de [F5] hetzelfde resultaat heeft met en zonder [SHIFT], maar in bepaalde gevallen kan de [F10] handiger zijn dan de [F5]. Als we namelijk in een tekst staan met de cursor kunnen we een in de computer aanwezig basic programma met [F10] opstarten terwijl de [F5] dan waarschijnlijk een syntax error zal opleveren. In de [F5] / [F10] staat namelijk niet in beide gevallen alleen het woordje 'run' geprogrammeerd. Het staat trouwens standaard met kleine letters geschreven. Er is in zowel [F5] als [F10] tevens een [RETURN] mee ingeprogrammeerd. Het verschil tussen de twee ligt echter in het feit dat bij de [F10] er eerst een 'clear screen' code (= 12) gegeven wordt. Door nauwkeurige bestudering van de werking kunnen we dit achterhalen, maar ook door de inhoud van de functietoetsen in het geheugen op te zoeken kunnen we dit achterhalen. Maar hoe vind je nu de inhoud van de functietoetsen ?

Inhoud functietoetsen

De meest simpele manier om aan de inhoud te komen is de opdracht 'KEY LIST' te geven. We krijgen dan netjes onder elkaar de inhouden van de tien functietoetsen.

```
KEY LIST
color
auto
goto
list
run
color 15,4,4
cload"
cont
list.
 run
```

Wat ons hieraan opvalt is dat de 'run' die bij [F10] hoort een positie naar rechts is opgeschoven. We vermoeden dat er op deze plaats een of andere code staat om er voor te zorgen dat we de [F10] ook midden in een regel kunnen gebruiken. Maar wat is die code en wat doet hij precies ? We bestuderen de werking nog eens wat nader en zien dan dat bij [F10] altijd eerst het scherm wordt schoon gemaakt. We zullen ongetwijfeld in een of ander handboek exact kunnen vinden waar de inhoud van de functietoetsen wordt

opgeslagen en wat er standaard in wordt gezet bij aanzetten van de machine. Maar dat is niet interressant; ik wil er zelf achter komen dan kan ik met deze techniek ook andere soortgelijke problemen te lijf.

Plaatsbepaling

We denken aan de geheugenopbouw van de MSX-computer en zien dan dat ergens boven in het geheugen de RAM staat die het systeem gebruikt. Ik denk hierbij aan de adressen van F000 en hoger. Zoeken we een en ander precieser na dan blijkt dat we slechts vanaf adres F380 hoeven te zoeken. Om nu al de adressen van F380 tot en met FFFF na te zoeken schrijven we een programma. De eerste versie van dit programma is erg simpel. We lezen met de PEEK een inhoud en zetten de inhoud dan als karakter op scherm. Zodra we menen gevonden te hebben wat we zoeken onderbreken we het programma.
Dit programma ziet er als volgt uit :

```
10 FOR I=&HF380 TO &HFFFF
20 PRINT CHR$(PEEK(I));
30 NEXT
```

Het blijkt al snel dat dit programma iets te simpel is. Aangezien we elke waarde normaal gebruiken zullen we ook een backspace (I = 8) een clear screen (I = 12) en een cursor up (I = 30) om er eens enkele te noemen niet alleen afdrukken maar ook uitvoeren in de betekenis dat het scherm bijvoorbeeld ook daadwerkelijk wordt schoongemaakt. We kunnen dan echter niet meer zien wat er stond.

Onderscheppen

We gaan in een verbeterde versie van dit programma de ongewenste codes afvangen. We doen dit niet door een lange rij IF's maar door een lijstje van codes in een tekstvariabele (ik noem hem zo direct T$) te zetten en dan te kijken of de ge'peek'te waarde daarbij zit. Is dit zo dan druk ik alleen een punt in plaats van de echte waarde. In basic is dit zeer simpel met de INSTR op te lossen.

```
10 READ A:IF A <=THEN T$=T$+CHR$(A):GOTO
10
20 DATA 8,10,12,13,28,29,30,31,127,999
30 FOR I=&HF380 TO &HFFFF
40 A$=CHR$(PEEK(I)):IF INSTR(T$,A$) THEN
A$="."
50 PRINT A$;PEEK(I),I+32768
60 NEXT
```

In regel 10 wordt de variabele T$ gevuld met de ongewenste codes die in regel 20 genoemd worden. Deze regel met bijvoorbeeld de waarde 999 afsluiten. Ik heb er maar enkele genomen maar iedereen kan er zelf bijzetten wat hem of haar nuttig lijkt. Als een teken, dat genoemd wordt in de lijst van regel 20 wordt ingelezen zorgt de test in regel 40 voor vervanging van het betreffende teken. In regel 50 worden nu het teken, dat eventueel vervangen is door een punt, de waarde en de plaats afgedrukt. Over deze plaats nog het volgende. Er wordt 32768 bij opgeteld omdat er anders negatieve waarden uitkomen. Op zich is dat niet erg maar het staat zo stom. Laat dit programma lopen en na enig getuur op voorbijschietende getallen en tekens vinden we dat de gezochte functietoetsen staan vanaf adres &HF87F of decimaal 30847.

beep,beep,beep

Als we geen zin hebben in dit getuur kunnen we er ook heel simpel een audio waarschuwing van maken. We zetten in functietoets 1 in plaats van de tekst 'color ' een aantal beepies. De code voor beep is 7. We doen dit met de volgende regels in basic :

```
A$="":FOR I=1 TO 12:A$=A$+CHR$(7):NEXT
```

waarmee we in de tekstvariabele A$ een dozijn beepies zetten en dan

```
KEY 1,A$
```

om deze beepies in functietoets 1 te plaatsen.
Als we nu het zoek programma starten zullen we niet de standaardvulling van [F1] meer kunnen vaststellen. Maar de positie waar de inhoud van [F1] staat zal zich luid beepend aan ons bekend maken. Weten we die plaats (F87F) eenmaal kunnen we na een herstart altijd de inhoud van [F1] vinden. Ik geef de gevonden lijst waarbij ik de tekst tussen quotes (') plaats. Let ook goed op spaties :

standaardvulling functietoetsen

key 1 vanaf F87F `'color '`
key 2 vanaf F88F `'auto '`
key 3 vanaf F89F `'goto '`
key 4 vanaf F8AF `'list '`
key 5 vanaf F8BF `'run*'`
* betekent hier de code voor return = 13

key 6 vanaf F8CF `'color 15,4,4*'`
* betekent hier de code voor return = 13
key 7 vanaf F8DF `'cload"'`
vreemd dat er geen spatie afkon,
bij key 1 tot en met 4 was dat zo keurig gedaan.
key 8 vanaf F8EF `'cont*'`
* betekent hier de code voor return = 13
key 9 vanaf F8FF `'list.*^^'`
* betekent hier de code voor return = 13
^ betekent hier de code voor cursor up = 30

key 10 vanaf F90F '@run*'
@ betekent hier de code voor clear screen = 12
* betekent hier de code voor return = 13
Een oplettend lezer zal opgevallen zijn dat de plaats gereserveerd voor inhoud van de functietoetsen 16 karakters groot is, terwijl volgens de handboeken er slechts 15 karakters in geplaatst kunnen worden. In onze beperkte memorydump zagen we al dat er allemaal nullen achter de tekst komen. Het ligt voor de hand te veronderstellen dat, bij het indrukken van een functietoets, de computer vanaf het beginadres leest en het tot een code 0 naar het beeldscherm plaatst.

Extra lange functietoetsinhoud

Ik denk dan onmiddellijk, dat ik door het vervangen van die nul door iets anders ik extra lange teksten kan maken. Even proberen :

```
KEY 1,"12345678901234567
```

GEEN eindquotes en een tekst die langer dan 15 karakters is. MIS ! Na [F1] krijg ik alleen 123456789012345. We zetten een code voor '+' op de plaats van de nul. Door ?asc("+") komen we te weten dat de code 43 is. Niet deze waarde uit het hoofd leren ! Methode onthouden en de waarden die echt belangrijk zijn onthouden we vanzelf, omdat die zo vaak voorkomen.

```
POKE &HF88E,43
```

BINGO ! Met het indrukken van [F2] nog steeds de oude tekst maar met [F1] krijgen we nu de tekst van beide functietoetsen. Hele lange teksten met meer dan honderd karakters, een compleet brievenhoofd of iets dergelijks zie ik al zitten. Jammer, maar na 39 karakters blijkt het toch niet te lukken. Weet iemand een oplossing laat het dan weten.

Fout in regel

Een programma dat we laten werken waar een fout in zit leidt tot een onderbreking van het programma. Zonder speciale maatregelen krijgen we een foutmelding op het scherm. Bijvoorbeeld :

```
Syntax error in 50
```

Als we nu op [F9] drukken krijgen we door de 'list.' opdracht direct de foutieve regel 50 op het scherm. De cursor wordt zelfs ook nog keurig aan het begin van de foutieve regel geplaatst. De punt achter de 'list' is te beschouwen als een afkorting van de laatst gebruikte regel. Als we een programma 'list'en en die listing onderbreken zal de [F9] ofwel de 'list.'opdracht de laatste gelistte regel tonen. De cursor zal ook dan aan het begin van de regel staan. Als we even nagaan wat er gebeurt is het simpel : eerst wordt de tekst 'list.' neergezet. Dan komt de return waardoor de opdracht uitgevoerd wordt, dat wil zeggen de betrokken regel wordt ge'list'. De prompt wordt gezet en de cursor gaat naar de regel eronder. En tot slot wordt de cursor twee regels omhoog geplaatst (tweemaal code 30 dus cursor up) zodat hij aan het begin van de regel staat. Dit laatste met de kanttekening dat de cursor dus twee regels omhoog wordt gezet. Is de regel dus niet langer dan de schermbreedte komt de cursor op het begin van de regel. In het andere geval, met langere regels dus, zal de cursor ergens in het midden van de regel komen maar wel aan de linkerkant van het scherm.

Extra verplaatsing

Even vooruitlopen op een fraaie editzaak. Het regelnummer zal bijna nooit fout zijn dus is het logischer de cursor niet aan het begin van de regel op het regel nummer te plaatsen maar direct aan het begin van de in die regel gegeven opdrachten. Om de cursor een 'woord' naar rechts te plaatsen dient de code 6. Als we dus bij het vullen van [F9] er ook een 6 in zetten zal de cursor zelfs naar de eerste letter van de regel gaan. De volgende regel vult [F9] met deze code :

```
KEY 9,"list."+ CHR$(13)+ CHR$(30)+
CHR$(30) + CHR$(6)
```

Wel ook hier de beperking dat de regel niet langer mag zijn dan scherm breed. Er is hiervoor wel een oplossing door mij gevonden maar echt goed vind ik die niet. Ik houd mij aanbevolen voor inzendingen.

```
KEY 9,CHR$(12) +"list."+ CHR$(13)+
CHR$(11)+ CHR$(31)+CHR$(6)
```

Bij deze [F9] wordt eerst het beeldscherm schoongemaakt (met 12) dan de 'list.' gegeven en vervolgens wordt de cursor op zijn 'home' (met 11) linksboven gezet, dan een regel naar beneden (met 31) en op de eerste letter gezet (met 6).

Nooit meer fouten

Ik heb ook nog gecontroleerd of je de instructie 'run.' kan geven maar dat lukt niet. Wel lukt het echter om 'delete.' als opdracht te geven. Met enig sarcasme zou je dus als tip kunnen geven dat slechte programmeurs hun programma's maar moeten uitbreiden met de volgende twee regels :

```
1 ON ERROR GOTO 60000
.....
oorspronkelijk programma
.....
60000 DELETE.
```

Als ze het programma maar vaak genoeg starten staat er geen enkele regel meer in het programma die niet kan werken. Sterker : bij zeer slechte programmeurs staat er behalve deze twee voorgestelde regels helemaal geen regel meer.
Grappig is ook dat de 'list.' ook gecombineerd kan worden met de '-'. Ik bedoel nu het kleine minstreepje, want ik kan mij voorstellen dat een argeloze lezer er door alle leestekens er nu niet meer uitkomt. Typ in :

```
KEY 9,"list-."+CHR$(13)
```

De [F9] zal er nu voor zorgen dat alle regels tot en met de regel waar het fout ging laten zien. En

```
KEY 9,"list.-"+CHR$(13)
```

zorgt er voor dat [F9] alle regels vanaf de fout laat zien.
Van de overige functietoetsen valt althans in de standaardvulling voor editwerk weinig over te zeggen. Voor ik echter aan speciale vullingen toekom wil ik eerst de rest van het toetsenbord doornemen.

Directe edittoetsen

Op het toetsenbord van de MSX zitten een groot aantal toetsen die bedoeld zijn voor de opmaak. De meest in het oog lopende zijn natuurlijk de cursortoetsen. Met deze toetsen kun je de cursor vrij over het scherm bewegen. Besef echter wel dat je niet over het programma kan bewegen maar alleen over het scherm. Opschuiven van de tekst op het scherm, het zogenaamde 'scrollen' kan dus niet met de cursortoetsen.

Teken weghalen

Om een enkel teken weg te halen hebben we meerdere mogelijkheden. De oplossing, die nog uit de tijd van de schrijfmachines stamt, is de backspace. Als we bedenken dat er bij een ouderwetse schrijfmachine geen cursortoetsen zaten is de werking van de backspace duidelijk. De letter voor de typpositie, nu dus voor de cursor, wordt verwijderd. De naam is nu eigszins misleidend geworden omdat er niet een spatie gezet wordt maar het teken verwijderd wordt. Toen de term ontstond bedoelde men dat er zonder dat er een teken gezet werd (dus space) een positie terug (dus back) gegaan werd. Wel was het zo dat alleen naar voren gegaan kon worden en dus niet meer naar een vorige regel. Op de computer heeft men dit overgenomen en met de [BS] kun je dus niet naar een vorige regel. Pas op dit geldt voor programmaregels en niet voor schermregels. Op het scherm kan een basicregel best drie regels lang zijn. Met de [BS] zijn we dan best in staan de gehele regel te verwijderen. Inderdaad de gehele regel want zodra we in de meest linkerpositie zijn aangekomen gaat de [BS] als [DEL] werken en wordt bij ontstentenis van een er voor geplaatst teken maar het teken waar de cursor opstaat genomen. De gehele regel zal verdwijnen.

Delete = verwijderen

Met de [DEL] kunnen we ook een teken verwijderen. Nu zal dat het teken zijn waar de cursor opstaat. In de beide gevallen, zowel met [BS] als [DEL], zal de rest van de regel opschuiven.
Tik in 123456789 en zet de cursor op de 5 drukken we op [DEL] zal de 5 verwijderd worden en de 6789 een plaats naar links opschuiven. Bij [BS] zal de 4 verwijderd worden en schuiven de 56789 een plaats naar links op. Blijven we de toets ingedrukt houden zal bij [DEL] de 6,7,8 en 9 in die volgorde verwijderd worden en blijven 1234 staan met de cursor daarachter. Bij [BS] zal echter niets overblijven en zal in volgorde 3,2,1,5,6,7,8 en 9 verwijderd worden.

Insert = invoegen

De [INS] werkt omgekeerd aan de [DEL]. Na indrukken van de [INS] wordt het cursorsymbool gehalveerd. Tikken we nu een of ander teken in zal dit op de plaats van de cursor komen, maar anders dan normaal zal daarmee niet het reeds aanwezige teken overschreven worden. De tekst rechts en onder zal opzij schuiven om zo plaats te maken voor het nieuw ingevoerde teken. In eerste instantie schuift alleen de tekst in de regel maar zou de regel op een nieuwe schermregel komen en daarmee andere teksten overschrijven schuiven ook die regel een positie naar beneden. Zou bij dit laatste tekst onder uit het beeld geschoven worden dan zal de gehele tekst een regel naar boven scrollen. Met [INS] kun je dus in tegestelling met [DEL] wel enig scrollwerk forceren.

CLS / HOME

Met de [HOME] toets zonder de [SHIFT] kunnen we de cursor linksboven op het scherm geplaats krijgen. De uitdrukking komt nog uit de printertijd. Van een beeldscherm bij een computer had nog bijna niemand gehoord en alle uitvoer geschiedde via de printer. Home was in die tijd gewoon naar linksboven op een vel en doe alsof dit de bovenkant van een nieuw vel is. De code is 11. Met code 12 werd aangegeven dat er ook papier doorgedraaid moest worden, dus met een schoon vel beginnen. Met code 13 [RETURN] alleen terug naar linkerkant regel en met 10 [LINE FEED] naar een nieuwe regel. Toen het beeldscherm zijn intrede had gedaan werd 10 overbodig en 13 werd de [RETURN] inclusief naar de nieuwe regel. Ook home werd overbodig maar beide bleven gehandhaafd in vele machines. Code 12 werd algemeen (niet unaniem, commodore heeft 147) voor een schoon scherm aanvaard.

Tabulator

Met de tabulator [TAB] kunnen we een aantal spaties neerzetten tot het eerst komende achtvoud. Tel wel met de meest linker positie is 0. Dit achtvoud is niet in te stellen, wel kunnen we in de PRINT de tab als basic instructie gebruiken. Er komt dan een waarde achter te staan die verteld tot welke positie er spaties moeten komen.

```
PRINT TAB(25);"PIET"
```

zal in de volgende regel 25 spaties zetten en dan de tekst "PIET".

```
PRINT "HET IS";TAB(25);"PIET"
```

zal eerst de tekst "HET IS" neerzetten en dan nog 25-6=19 spaties en net als daarnet de tekst "PIET".
Dat uitwissen van tekst met TAB is eigenlijk niet zoals het hoort en vele tekstverwerkingsprogramma's laten de tabulator dan ook wel instellen door de gebruiker en de cursor zonder vernielende werking over de tekst gaan.
Zoals reeds in het begin gesteld, is er zoveel te vertellen dat een artikel echt te groot zou worden. Maar niet getreurd in het volgende nummer ga ik verder. Om in mijn stijl afscheid te nemen; volgende keer ga ik schrijven over de [CTRL] codes en [ESC] sequences. Probeer eens wat en vind zelf de leuke dingen.

Succes op uw ontdekkingsreizen
Frank H. Druijff

Turbo Pascal, Bios Pascal, Bastaard Pascal?

Diegenen die reeds ervaring hebben in het programmeren in Turbo Pascal zullen waarschijnlijk wel al lang hebben gemerkt dat de uitvoer van Pascalprogramma's naar het beeldscherm eerder traag verloopt. Zelfs trager dan in Basic.
Wel hier is een mouw aan te passen. Een beetje kennis van machinetaal, en een beetje kennis van pascal en hop.... **machinetaal pascal.**. Dit programma laat duidelijk het verschil zien tussen pascal, pascal met bios, en pascal met machinetaal. Tevens zitten in dit programma een aantal handige procedures die je eruit kan halen om eventueel in je eigen programma's te gebruiken.

Het programma wordt ingetypt in de turbo pascal editor.
De remarks kan je eventueel weglaten. Je compileert het programma en je zal versteld staan van het snelheidsverschil!

TESTGR3 COM D2
TESTGR3 PAS D2

REAKTIE.BAS

REAKTIE BAS D1

Met behulp van dit programma kan u uw reaktiesnelheid meten en dit tot op een duizendste van een seconde nauwkeurig.
(Van Wulpen Henk)

LIJNEN.BAS - BOL1.BAS - BOL2.BAS - CILIND1.BAS - CILIND2.BAS

disk 1

Dit is een verzameling van vijf grafische programma's die de illusie van beweging op het scherm op een werkelijk prachtige manier overduidelijk illustreren.
Er wordt een driedimensionale figuur getekend, bestaande uit een zelf aantal te kiezen puntjes. Hoe meer puntjes er moeten worden getekend, hoe meer tijd dit in beslag neemt. De figuur wordt in 48 standen getekend, steeds onder een verschillende hoek. Als dit is gedaan, drukt u spatie en de 48 standen worden vlug na elkaar getoond; zo lijkt het alsof het voorwerp draait. Het tijdsinterval tussen twee opeenvolgende beelden wordt door uzelf bepaald.
Als u tijdens het draaien op spatie drukt, kan u zelf de figuur laten draaien met behulp van de cursortoetsen. Het draaien wordt gestopt met CTRL-STOP. U kan dezelfde figuur nogmaals zien bewegen (eventueel langzamer of sneller) door het programma terug te starten; zo hoeft u dus niet op het tekenen te wachten.
(Coremans Raf)

DATATIME.BAS

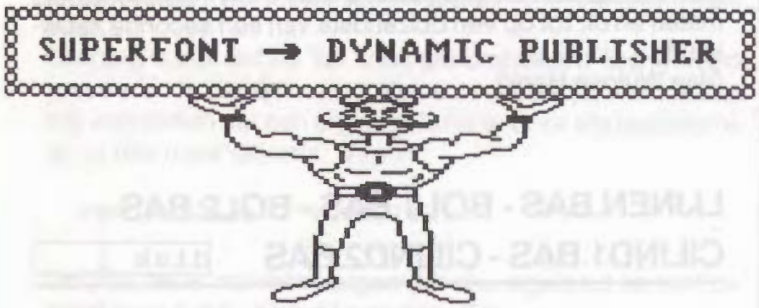
DATATIME BAS D1

Met dit programma, een disk-utility, kan u de datum en het uur van files naar hartewens veranderen. Zo kan u dus perfekt verhinderen dat iemand (uw vrouw ?) weet tot hoe laat u weer achter het toestel zat gekluisterd.
(Van Wulpen Henk)

VANGNET.BAS

VANGNET BAS D1

In dit spelletje is het de bedoeling dat u de hotelgasten, die van het dak springen (vallen), opvangt in het vangnet. Naarmate het spel vordert vallen ze alsmaar sneller. Gelukkig staat de ambulancewagen vrij dichtbij.
(Rutten A.)



Van SUPERFONT naar DYNAMIC PUBLISHER ...

Een krachttoer ? Neen, hoor. Eigenlijk staan de nodige aanwijzingen reeds vermeld in de handleiding van Superfont. Maar voor wie de moed of het geduld nog niet had, zet ik alles even op een rijtje.

De werkwijze is de volgende:

- Via een basic-programma worden de tekeningen op het scherm gezet (programma inladen / Superfontschijf in de computer / RUN).
- Dat scherm wordt (met hetzelfde programma) opgeslagen op disk.
- Met Dynamic Publisher kan je dat scherm inladen om het op zijn beurt weg te schrijven als stempel: PAK STEMPEL en STEMPEL BEWAREN.

ANIMAL1.BAS is een uitgewerkt voorbeeldprogramma. Regel 10 biedt je de mogelijkheid om via CTRL + STOP het programma te onderbreken en zonodig wijzigingen aan te brengen voordat het scherm weggeschreven wordt.
In regel 70 werd SCREEN 6 gekozen, omdat D.P. daarop werkt. Daardoor moeten de karakters dubbelbreed worden weergegeven (regel 80) want SCREEN 6 is 512 pixels breed en niet 256.
In SUPERFONT zijn soms aanhalingstekens voorzien als af te drukken karakter. Hoe je dat in BASIC opvangt, vind je in regel 210. (chr$(34))
Regel 2000 wacht op een toetsaanslag om je de gelegenheid te geven van schijf te wisselen vooraleer alles op te slaan.
Alle tekeningen op die manier in D.P. krijgen is natuurlijk een hele karwei, maar het resultaat is echt lonend. Voor mezelf heb ik trouwens alle tekeningetjes nog eens gerangschikt in 29 stempels (personen, waterdieren, insekten, auto's,...) alles samen zo maar 277 K (!), maar ik beschik dan ook over een arsenaal aan illustratiemogelijkheden.

Deze 29 stempels vind je, samen met het voorbeeldprogramma "ANIMAL1.BAS" op schijf nummer 3 van het SUPERFONT-pakket.
Wie opziet tegen de vele avonden die nodig zijn om alles in te tikken - of is dat nu juist het leuke ?- , moet me maar eens bellen. Tegen een kleine vergoeding stuur ik je dan 107 BASIC-programma's (1 voor elk tekeningenfont). Ik heb ze toch liggen.

ANIMAL1 BAS D2

Jos Simal
(02) 269 27 66

```
1 CLEAR1000,&HA800
10 ONSTOPGOSUB360:STOPON
20 '
30 '  ANIMAL1.BAS
40 '
50 BLOAD "sf",R
60 BLOAD "ANIMAL1.FON"
70 SCREEN 6:CLS
80 PRINTCHR$(14)
90 OPEN "GRP:" AS #1
100 X=10:Y=0'miereneter 1
110 A$="   ABCDEF":GOSUB 350
120 A$="  GHIJKLM":GOSUB 350
130 A$="NOPQRSTUV":GOSUB 350
140 X=180:Y=0'miereneter 2
150 A$="   ABCDWX":GOSUB 350
160 A$="  GHIJKYZ0":GOSUB 350
170 A$="NOPQRSTU":GOSUB 350
180 X=370:Y=0'ree
190 A$="pqrs":GOSUB 350
200 A$="#tuvwx":GOSUB 350
210 A$="  yz!"+CHR$(34):GOSUB 350
220 X=200:Y=100'krokodil
230 A$="456789":GOSUB 350
240 X=20:Y=160'mier 1
250 A$="abc":GOSUB 350
260 X=120:Y=160'mier 2
270 A$="def":GOSUB 350
280 X=220:Y=160'mier 3
290 A$="ghi":GOSUB 350
300 X=320:Y=160'mier 4
310 A$="jkl":GOSUB 350
320 X=420:Y=160'mier 5
330 A$="mno":GOSUB 350
340 GOTO 2000
350 PRESET (X,Y):PRINT A$:Y=Y+16:RETURN
360 SCREEN0:LIST
2000 IF INKEY$=""THEN 2000
2010 COPY(0,0)-(512,212)TO "ANIMAL1.STP"
```

Voorbeeldprogramma : ANIMAL1.BAS

DE TURBO PASCAL COMPILER DEFAULT FILE EXTENSIE

De TURBO PASCAL compiler bevat een vlot bruikbare editor. De editor is natuurlijk bedoeld voor het ingeven en wijzigen van TURBO PASCAL programma's. Hij is echter ook goed bruikbaar voor algemeen edit werk.

Voorbeeld : het grondig herwerken van MSX Basic programma's is niet zo handig met de Basic edit-mogelijkheden, daarom Basic programma saven als ascii file (A optie), editteren met TURBO PASCAL, terug laden in Basic.

Het is wel hinderlijk dat TURBO PASCAL steeds .PAS als als default file extensie gebruikt, dwz. indien u een filenaam opgeeft zonder een punt, dan plakt TURBO er automatisch .PAS achter. Voorbeeld : indien u vraagt om de file PROGRAM te laden, dan gaat TURBO de file PROGRAM.PAS laden.

Met het hier voorgestelde programma is het mogelijk om de default file extensie te wijzigen in 3 willekeurige karakters, bv. BAS.

Opmerking : niet vergeten de default extensie PAS te herstellen als u PASCAL programma's compileert. Dit is belangrijk omdat de wijziging van de default extensie ook de compiler directive $I <filename> beïnvloedt : indien de filename geen punt bevat, wordt ook hier de gewijzigde default extensie gebruikt.

BESCHRIJVING VAN HET PROGRAMMA

HOOFDLUS

De hoofdlus van het programma start na BEGIN {main}. Na het tonen van een hoofding wordt een REPEAT UNTIL loop gestart.

Via een call naar de funktie FETCH_EXTENSION wordt de momenteel ingestelde default extensie opgehaald en vervolgens getoond.

De gebruiker kan nu 3 nieuwe karakters ingeven. Merk het gebruik van de speciale variabele BUFLEN op om de input te beperken tot 3 karakters.

Met een CASE statement wordt de input onderzocht. Indien geen karakters worden ingegeven dan wordt het programma beëindigd, dwz. de laatst getoonde karakters vormen vanaf nu de default extensie. Indien 3 karakters worden ingegeven, dan wordt de procedure UPDATE_EXTENTION opgeroepen om de karakters te bewaren in de TURBO.COM file, waarna ze opnieuw getoond worden. Tenslotte, als de input een andere lengte heeft, dan wordt dit als fout beschouwd.

OPHALEN EN BEWAREN VAN DE EXTENSIE

De 3 karakters die de default extensie bepalen staan op een bepaalde plaats in de file TURBO.COM. Deze lokatie kunnen we aanduiden met een recordnummer (1 record = 128 bytes) en een offset binnen dat record.

Om de 3 karakters op te halen wordt 1 record van TURBO.COM gelezen in de variabele TURBO_IMAGE (zie uitleg FETCH_EXTENSION). Deze variabele is van het type TURBO_IMAGE_REC, welk gedeclareerd is als een RECORD (=struktuur) met een bepaalde volgorde van DUMMY_x velden en CHAR_x velden. Doel van de DUMMY_x velden is om de bytes waarin we niet geïnteresseerd zijn over te slagen. Met de CHAR_x velden kunnen we de karakters die de extensie vormen rechtstreeks lezen of schrijven. Dummy velden zijn arrays van bytes. Char velden zijn van het type CHAR, omdat dit type eenvoudig kan omgevormd worden naar het type STRING (waarin de extensie door het programma bewerkt wordt).

Voor het bewaren van de nieuwe extensie worden de CHAR_x velden overschreven met nieuwe karakters. Vervolgens wordt het voordien gelezen record van TURBO.COM overschreven met de nieuwe inhoud van de variabele TURBO_IMAGE.

Opmerking : laat u niet in verwarring brengen door de dubbele betekenis van record in bovenstaande beschrijving. De eerste betekenis slaat op MSX-DOS record (deel

van een file). De tweede betekenis is record als een onderdeel van de PASCAL taal.

DE FUNKTIE FETCH_EXTENSION

In FETCH_EXTENSION wordt via een ASSIGN en RESET funktie de TURBO.COM file geopend. Met een SEEK funktie wordt het eerste te lezen record bepaald. Vervolgens wordt met de BLOCKREAD funktie 1 record ingelezen in de variabele TURBO_IMAGE.

De 3 CHAR_x velden worden vervolgens achter elkaar geplakt tot een string en teruggeven als funktieresultaat.

Tenslotte wordt de file access afgesloten met een CLOSE.

DE PROCEDURE UPDATE_EXTENSION

De eerste akties zijn identiek aan fetch, dwz. ASSIGN, RESET en SEEK.

De als string parameter meegegeven nieuwe extensie wordt omgevormd naar 3 (uppercase) karakters, die in de CHAR_x velden van TURBO_IMAGE worden geplaatst.

De BLOCKWRITE funktie wordt gebruikt om de ene record in TURBO.COM te overschrijven, gevolgd door CLOSE.

GEBRUIK EN OPMERKINGEN

Compilatie van het programma en gebruik is gedocumenteerd in de commentaar blok vooraan in het programma. Onder WARNING zijn ook enkele beperkingen opgegeven.

EEN TOEMAATJE

In een PASCAL programma zijn meestal veel identifiers gedefiniëerd, dit zijn namen van procedures, funkties, variabelen, types, const'en, ...

Om het programma beter leesbaar te maken is het aangeraden om bepaalde regels te volgen bij de keuze van de namen.

PROCEDURES en FUNCTIONS : de identifier laten beginnen met een werkwoord (voorbeelden uit het programma : FETCH_, UPDATE_)

TYPES : de identifier laten eindigen met een standaard afkorting die de aard van het type aanduidt

```
STRING TYPES   : _STR (zie programa :
EXT_STR)
RECORD TYPES   : _REC (zie programma :
TURBO_IMAGE_REC)
ARRAY TYPES    : _ARR
SET TYPES      : _SET
POINTER TYPES  : _PTR
```

VARIABELEN : alle variabelen die niet simpel zijn (dwz. niet gewoon een integer of ander standaard type) worden vooraf als type gedeclareerd, hierbij komt de naam van de variabele overeen met de naam van type minus de standaard extensie (voorbeeld uit het programma : de variabele TURBO_IMAGE is van het type TURBO_IMAGE_REC)

F. De Raedt

TBEXT PAS D2

MACHINETAAL PRAKTISCH TOEGEPAST!

Hieronder volgt voor de misschien al iets meer gevorderde programmeur een machinetaal-programma dat bijna volledig zelfstandig functioneert, dus zonder gebruik te maken van de **MSXBIOS**. Enkel de routine om van schermmode te veranderen en de routine om de schermkleuren te veranderen zijn opgenomen in het programma om de eenvoudige reden dat de tijdwinst evenredig blijft met een zelfgeschreven routine. De beginnende machinetaal programmeur heeft hier de ideale kans om eens nauwlettend de functie van elke mnemonic te volgen om zo tot een beter inzicht te komen van machinetaal.
Het programma functioneert zowel op een **MSX1** als op een **MSX2** toestel. De voorwaarde om het te laten werken is enkel dat u beschikt over de **GEN80 ASSEMBLER**. Indien je deze assembler hebt, typ je gewoon het programma in de editor in en assembleert het. Beschik je over een ander type assembler, moet je zowel de instructies eigen aan de assembler aanpassen (Bv. defb,defm, enz...), maar tevens het programma op een juiste plaats in het geheugen zetten. Kijk hiervoor uw handleiding na.
Let wel dit programma opereert in een **DOS** omgeving!
Gen80 gebruikers die dit programma hoger in het geheugen willen zetten moeten gebruik maken van de zgn. ORG's instructies.
Meer informatie hierover kan je vinden in uw handleiding.

Raoul Dorissen

PROG1 GEN D2

PROG1 MTP D2

BELGIE BAS D1

HELI1 BIN D1

DRAWING.BAS

DRAWING BAS D1

Een paar maanden terug behandelden Frank en Clark uitvoerig de mogelijkheden van het DRAW-kommando. Het DRAWING-programma sluit hier perfekt bij aan. Het programma is erg lijvig, er zijn echter heel wat mogelijkheden ingebouwd.
DRAWING laat je toe met de cursortoetsen of de joystick een tekening op het scherm te maken. Alle akties en stappen worden nauwkeurig gevolgd en op verzoek genereert Drawing een basic-programma op cassette of diskette. Dit gegenereerde programma krijgt (op disk) automatisch de extensie ".DRW".
Wanneer je dit programma achteraf laadt dan zal het precies dezelfde tekening opnieuw kunnen samenstellen. Het 'produkt'-programma wordt in ASCII bewaard, zodat het eventueel ook zou kunnen worden 'gemerged' met een ander basic programma. Bijgevoegde tabel geeft U een overzicht van de verschillende kommando's, ze worden opgeroepen met behulp van de funktietoetsen van 1 tot en met 9. Veel drawing plezier!
(Van Den Eijnden Wilfried)

FUNCTIE TOETS	FUNCTIE
1	Kleur instellen
2	Nieuw beginpunt plaatsen
3	Schaalinstelling
4	Beweeg zonder te tekenen
5	Tekening opnieuw construeren
6	BASIC-listing naar printer
7	Laden van ASCII-files
8	Bewaren van ASCII-files
9	Bewaren van BASIC-programma's

SLANGEN.BAS

SLANGEN BAS D1

Een nieuwe en leuke slang-variatie: tracht de slang zo lang mogelijk in leven te houden door hem grijze paddestoelen te laten eten. Het zou te eenvoudig zijn als de rode paddestoelen niet giftig waren.
(Timmermans Luc)

BELGIE.BAS (HELI1.BIN wordt geladen)

Dit programma test uw kennis over de ligging van de steden in België. Met uw helikopter dient u zo snel mogelijk de aangegeven bestemmingsplaats te bereiken. De moeilijkheidsgraad verhoogt stelselmatig.
Dit spel is gemaakt voor de MSX 2 computer.
(Boets Erwin)

REISBAG.BAS

REISBAG BAS D1

Een geheugensteuntje voor de reis die voor de deur staat. U maakt een overzicht van de bagage die u wil meenemen. Heel handig daar u het voor het volgende jaar kan aanpassen. De mogelijkheid wordt geboden om een afdruk te vragen op de printer.
(Kersemans Guy)

DIEFSTAL.BAS

DIEFSTAL BAS D1

Dit spel is gebaseerd op het gezelschapsspel 'Scotland Yard' waarin u de rol van dedective vervult. Uw taak bestaat er in om de juwelendief te vatten. U kan hem echter niet zien, alleen maar horen.
U vindt voldoende uitleg in het spel.
(Timmermans Luc)

SCHUIF.BAS

SCHUIF BAS D1

Een prachtig schuifraam voor de MSX 2 computer. Hoe goed denkt u na en hoe vingervlug bent u.
(Druijff Frank)

DARTS.BAS

DARTS BAS D1

Het bekende 'vogelpikken' kan nu dus gebeuren op uw MSX computer. Het spel kan worden gespeeld met toetsenbord en joystick. De moeilijkheidsgraad, het aantal spellen en de beginwaarde kan naar keuze worden ingesteld.
(Rutten A.J.)

SUPER IMPOSE & VIDEO.

Van OASIS, de makers van de T.V.-Krant, ontvingen wij een nieuw programma. Oasis is nogal bezig met de NMS-8280 van Philips zodat men zich niet zal verbazen dat ook SUPER IMPOSE & VIDEO in de eerste plaats bestemd is voor deze computer. Gelukkig maar, want echte toepassingen zijn er buiten de T.V.-krant nooit echt gekomen voor de NMS-8280.

Doel van het programma.

In eerste instantie is het programma bedoeld om uw video-opnames te voorzien van ondertiteling. Dit is natuurlijk ook mogelijk met het door Philips meegeleverde Videographics, maar leest u rustig verder en u zult ontdekken dat van dit programma de mogelijkheden veel uitgebreider en beter zijn. Voorts is het mogelijk om titelpagina's, aftitelingsrol, lichtkrant en T.V.-Krant op uw scherm en videoband te produceren. De laatstgenoemde mogelijkheden zijn ook interessant wanneer u een MSX 2 zonder Super Impose bezit. Hier komen we later op terug.

De mogelijkheden.

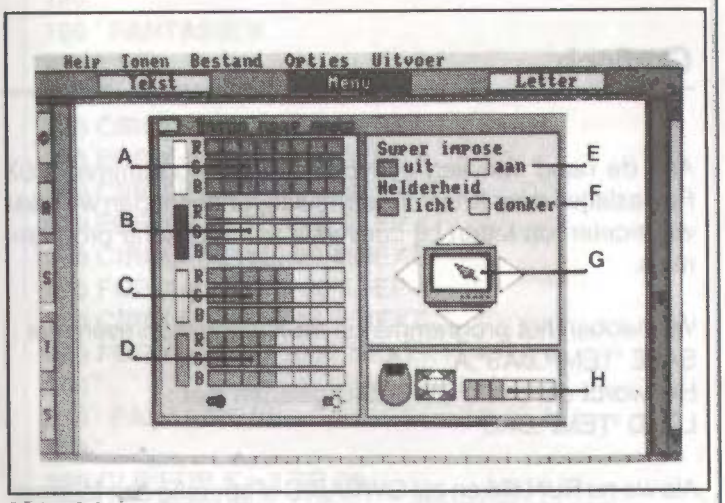
De 3.5 inch floppy start automatisch op. Het programma opent met een fraai OASIS titelscherm, gevolgd door het hoofdscherm. Aan de linkerzijde staat een verticale balk met de nummering van de regelparen. Aan de rechterzijde een balk om snel door de regelparen heen te bladeren.(zoals EASE) Bovenaan vindt u de statusbalk met daarin de keuze uit 5 menu's, w.o. het HELP- menu. Dit hulpmenu geeft de Editfuncties. Deze komen overeen met de combinaties van toetsen die u al gewend bent in de standaard basic-editor van de computer. U kunt deleten, tussenvoegen, karakter en/of regel verwijderen enz. De editor is voor dit werk compleet. Onder de kop TONEN vindt u ondermeer de kantlijn aan of uit, het lettertype waarmee u wilt werken, bij welk regelpaar u wilt beginnen etc. Onder de kop BESTAND kunt u de tekstfiles welke u hebt aangemaakt saven, laden of wissen (op scherm). Het saven gebeurt onder een door u zelf opgegeven naam. Makkelijk wanneer u bv. 2 films tegelijk onder handen hebt.

Onder de kop OPTIES kunt u de breedte van de kantlijn instellen en met welk lettertype u wilt opstarten. Ook onder deze kop vindt u alle instellingen omtrent de videofuncties. Via een fraai hulpscherm kunt u de kleuren van voorgrond, achtergrond, schaduw en kleur van de letter instellen. Ook via dit menu wordt de helderheid van de weergave , snelheid van de wijzer en het verplaatsen van het beeldscherm geregeld Voor de ondertiteling kunt u aangeven of u rechts of links wilt uitlijnen dan wel centreren. Ook de tijd welke het regelpaar blijft staan is middels drie opties instelbaar. Dezelfde instelmogelijkheden zijn er ook voor de wijze van aftiteling etc. Ook kunt u middels dit onderdeel files op schijf wissen. Onder de kop UITVOER start u de ondertiteling, aftiteling en /of licht- krant. De aangemaakte tekst kunt u ook nog afdrukken op iedere printer. Onder de statusbalk staat een tweede rij opties wo. TEKST.

Via deze optie komt u in het tekstscherm hetwelk de rest van het beeldscherm inneemt. Uiteraard zijn hier niet alle opties besproken. Zouden we dat wel doen dan werd het artikel eens zo lang.

Karaktersets.

Standaard krijgt u bij dit programma de beschikking over maar liefst 7 karaktersets tw. CLASSIC en NORMAAL voor de ondertiteling en BROADWAY, GOTISCH, VET, GARNET en DIAMOND voor het overige werk. U kunt natuurlijk ook de laatste vijf voor ondertitelen gebruiken wanneer u dat verkiest. Voorts heeft u onder de balk direct onder de statusbalk nog een optie t.w. LETTER. Onder deze optie kunt u de karakters bewerken door ze normaal of schuin te plaatsen en die beide opties kunt u ook nog eens gevuld activeren. Een handigheid bij het ondertitelen is ook de mogelijkheid om de tekst op een achtergrondbalk te plaatsen hetgeen de leesbaarheid ten goede kan komen. Overigens wordt de tekst proportioneel op het scherm gezet volgens de WYSIWYG methode.



Werking.

Degenen onder u die al eens aan het ondertitelen zijn geweest met Video- Graphics weten dat een en ander uiterst omslachtig verloopt. Om de drie regelparen moet men de volgende weer van schijf inlezen. Ook is er slechts 1 lettertype voor handen. Met Superimpose & Video kunt u achter elkaar van 1 tot 418 regelparen aanmaken en vervolgens in uw video-opname verwerken. Hiervoor staat een editor tot uw beschikking en diverse karaktersets met daarbovenop nog eens de verschillende uitvoeringen. Nadat u de kantlijn hebt ingesteld gaat u de tekstregels aanmaken in regelparen. Een regelpaar bestaat, hoe kan het ook anders, uit 2 regels. Bent u klaar en heeft u het lettertype, kleur etc. ingesteld, dan saved u eerst de file naar schijf. Vervolgens start u via het menu UITVOER het eigenlijke ondertitelen. Dit gebeurt door een druk op de spatiebalk of actieknop van de muis. De regel verdwijnt afhankelijk van uw instelling. Uiteindelijk worden video-opname en ondertiteling via de computer gezamenlijk op een band opgenomen. Een kind kan de was doen.

Structuur.

Het programma werkt volledig met de zgn. 'PULL DOWN' menu's. Deze menu's werken zeer fraai en snel. De besturing van de wijzer geschiedt middels de cursortoetsen of de muis, waarbij opvalt dat ook de besturing middels de cursor soepel en vlot verloopt. Het programma is in hoofdzaak in basic geschreven doch wordt ondersteund door de KUN basic compiler, ook wel X-BASIC genoemd. In SUPER IMPOSE & VIDEO ziet u het resultaat. Het programma werkt snel en wanneer u gaat ondertitelen razendsnel. Zou u dit niet hebben gelezen dan zou u zeker hebben gedacht dat het om een volledig in machinetaal geschreven programma gaat. De programmeurs J. Koppens en J. Jansen hebben in combinatie met KUN basic het onderste uit de MSX-2 weten te halen.

Alternatief gebruik.

Dit programma leent zich naar onze mening niet uitsluitend voor ondertiteling doch heeft meer in zijn mars. In feite is de mogelijkheid tot het aanmaken van een T.V. krant de aanschaf van dit programma al waard. U kunt dan middels aangemaakte schermen allerlei boodschappen of reclame aan uw medemens kenbaar maken. Ook de aftitelingsrol leent zich daarvoor of mogelijk zelfs de eenvoudige lichtkrant. Let wel, met een NMS-8280 en daarnaast vanzelfsprekend twee videobronnen kunt u alle mogelijkheden uitbuiten.

Conclusie.

Met dit programma gooit OASIS zeer hoge ogen. Het programma blinkt uit door gebruiksvriendelijkheid, kwaliteit en het fraaie eindresultaat. Het is mij een volstrekt raadsel dat een dergelijk programma nog niet eerder werd gelanceerd door een commercieel softwarehuis. Een geluk voor u misschien. Een dergelijk program uitgebracht door de commercie zou zeker twee tot driehonderd gulden kosten. Het prijskaartje welk OASIS eraan hangt is beduidend minder. Het programma kost u fl 125,- /Bfr 2500, franco thuis. Hiervoor krijgt u de schijf met Nederlandse handleiding, verpakt in een fraaie hoesverpakking en wordt echte aftersales service gegarandeerd. Persoonlijk vinden wij het deze prijs dubbel en dwars waard. Wilt u meer weten of het bestellen, dan kunt u telefonisch contact opnemen met: J. Jansen tel. 04950-20941. Wel graag na 18.00 u. Voor België : MSX CLUB software service of uw MSX-dealer.

Conversie MSX-BASIC - IBM/GWBASIC

Microsoft heeft in de loop der jaren verschillende versies "BASIC" ontwikkeld. Deze programmeertaal vinden we dan ook terug op verschillende toestellen, al dan niet aangepast aan de beschikbare hardware-mogelijkheden.
In dit artikel willen we een aantal verschillen tussen MSX-BASIC en GWBASIC bestuderen, vooral met het oog op de eventuele conversie van programma's.
Via de "ASCII-bewaar" - methode kunnen bestanden en programma's van MSX naar MS-DOS en omgekeerd. Het schijfformaat is identiek en dat is een belangrijk gegeven ! Programma's die vanuit GWBASIC als ASCII worden weggeschreven, moeten een extensie hebben in de naam (.BAS), bij MSX geldt deze verplichting niet.

De PC-standaard MS-DOS munt momenteel uit in de diversiteit van de beschikbare grafische kaarten, die de compatibiliteit voor de gemiddelde leek aardig in het gedrang brengt.
CGA, HERCULES, EGA, VEGA...een veelheid aan displaymogelijkheden is ter beschikking. Meestal vragen deze extra kaarten extra "drivers" en de meeste programma's dienen dan ook vooraf "geïnstalleerd" te worden.
Voor de eenvoud hebben we het hier over de standaard schermtoestand CGA met volgende mogelijkheden :

SCREEN 0 : tekstmode met 40 of 80 kolommen
SCREEN 1 : MEDIUM RESOLUTIE (320 x 200, 4 kleuren/grijstinten), te vergelijken met MSX SCREEN 2 (255 x 192, 16 kleuren)
SCREEN 2 : HOGE RESOLUTIE (640 X 200 monochroom), te vergelijken met MSX SCREEN 6,7 (512 X 212)

Programma's in tekst-mode

Deze programma's geven de minste problemen. We dienen wel op te letten of er geen speciale schermeffecten gebruikt zijn met de specifieke karaktersets. Raadpleeg de bijgevoegde tabellen voor conversie. Het eerste gedeelte van de MSX-karakterset wordt verkregen door CHR$(1) + (64 + code) te sturen ! Om volledig te zijn hebben we ook de EPSON FX-80 karakterset afgedrukt, merk op dat de karakters boven 127 ITALIC (schuinschrift) zijn.

Zoals bij MSX2 wordt het schermformaat bepaald met het statement WIDTH (40 of 80).
In GWBASIC kan met WIDTH LPRINT ook het aantal kolommen van de printer gekozen worden ! Het WIDTH statement kan hier ook gebruikt worden om een DEVICE te definiëren. (WIDTH #1,40). Als daarna kanaal #1 gebruikt wordt voor uitvoer, wordt de kolombreedte 40 hiervoor gebruikt.

Belangrijk verschil bij de schermbehandeling :
MSX : LOCATE X,Y (x = kolom, y = rij) , waarbij de kolomteller start met 0 en loopt to 39 of 79, afhankelijk van het gekozen WIDTH-statement.
GWBASIC : LOCATE Y,X (x = kolom, y = rij), waarbij de teller loopt van 1 tot 40 of van 1 tot 80 !

Grafisch

Aan de hand van een eenvoudig MSX-programma, MSX Fantasietjes door Kris Vandermeulen, ondervinden we waar we moeten op letten bij conversie van grafische programma's.

We hebben het programma op MSX weggeschreven met :
SAVE "TEMP.BAS",A
Het wordt daarna in GWBASIC geladen met :
LOAD "TEMP.BAS"

Als we nu RUN tikken zal GWBASIC al spoedig een foutmelding geven (SYNTAX ERROR, ILLEGAL FUNCTION CALL...). Deze fout kunnen we dan verbeteren en zo verder het programma doorspitten. Toch wel tamelijk omslachtig ! Na een paar correcties wordt toch duidelijk waar we moeten op letten, en kunnen we vooraf alle problemen opvangen.
Achtereenvolgens moesten we de volgende ingrepen uitvoeren om het programma te laten lopen (vergelijk ook de beide listings):

SCREEN 2 wordt SCREEN 1 (GWBASIC-SCREEN 2 is monochroom)
Het COLOR-commando vraagt 2 parameters, de waarden kunnen variëren van 0 to 3, er zijn slechts 4 kleuren !

FANTASIE M16 D1

MSX-versie

```
10 REM ***********************
20 REM Msx Fantasietjes
30 REM (c) Vandermeulen Kris
40 REM St.-Truidense Stwg 168
50 REM 3350        Linter
60 REM on VG8010
70 REM date : 2-9-1987
80 REM ***********************
100 '
110 ' FANTASIE I
120 '
125 SCREEN 2:COLOR,1,1:CLS
130 PI=3.14159:AL=2*PI:COLOR,1,1
140 FOR K=O TO AL STEP AL/32
150X=INT(70*COS(K))+122:Y=INT(70*SIN(K))+95
160 CIRCLE(X,Y),50,4:CIRCLE(X-1,Y-1),50,4
170 NEXT K:FOR W=0 TO 3000:NEXT
180 '
190 ' FANTASIE II
200 '
210 CLS:FOR Y=152 TO 30 STEP-4
220 CIRCLE(78,Y),25,4:NEXT
230 FOR X=78 TO 200 STEP 4
240 CIRCLE(X,30),25,4:NEXT
250 FOR X=78 TO 200 STEP 4
260 CIRCLE(X,152),25,4:NEXT
270 FOR X=152 TO 30 STEP-4
280 CIRCLE(200,X),25,4:NEXT
290 FOR W=0 TO 3000:NEXT
300 '
310 ' FANTASIE III
320 '
330 CLS:FOR K=1 TO 30
340 X=INT(RND(1)*255):Y=INT(RND(1)*192)
350 X2=INT(RND(1)*255):Y2=INT(RND(1)*191)
360 LINE(X,Y)-(X2,Y2),4,B
370 NEXT:FOR W=0 TO 3000:NEXT
380 '
390 ' FANTASIE IV
400 '
410 CLS:FOR X=25 TO 226
STEP4:CIRCLE(X,30),25,4:NEXT
420 FOR X=30 TO 162
STEP4:CIRCLE(226,X),25,4:NEXT
430 FOR X=226 TO 30
STEP-4:CIRCLE(X,162),25,4:NEXT
440 FOR X=162 TO 90
STEP-4:CIRCLE(30,X),25,4:NEXT
450 FOR X=30 TO 174
STEP4:CIRCLE(X,90),25,4:NEXT
460 FOR W=0 TO 5000:NEXT:RUN
```

GWBASIC-versie

```
10 REM ***********************
20 REM Msx Fantasietjes
30 REM (c) Vandermeulen Kris
40 REM St.-Truidense Stwg 168
50 REM 3350        Linter
60 REM on VG8010
70 REM date : 2-9-1987
80 REM ***********************
100 '
110 ' FANTASIE I
120 '
125 SCREEN 1:COLOR 0,2:CLS
130 PI=3.14159:AL=2*PI
140 FOR K=O TO AL STEP AL/32
150 X=INT(70*COS(K))+122:Y=INT(70*SIN(K))+95
160 CIRCLE (X,Y),50,3:CIRCLE (X-1,Y-1),53,3
170 NEXT K:FOR W=0 TO 3000:NEXT
180 '
190 ' FANTASIE II
200 '
210 CLS:FOR Y=152 TO 30 STEP-4
220 CIRCLE (78,Y),25,3:NEXT
230 FOR X=78 TO 200 STEP 4
240 CIRCLE (X,30),25,3:NEXT
250 FOR X=78 TO 200 STEP 4
260 CIRCLE (X,152),25,3:NEXT
270 FOR X=152 TO 30 STEP-4
280 CIRCLE (200,X),25,3:NEXT
290 FOR W=0 TO 3000:NEXT
300 '
310 ' FANTASIE III
320 '
330 CLS:FOR K=1 TO 30
340 X=INT(RND(1)*255):Y=INT(RND(1)*192)
350 X2=INT(RND(1)*255):Y2=INT(RND(1)*191)
360 LINE (X,Y)-(X2,Y2),2,B
370 NEXT:FOR W=0 TO 3000:NEXT
380 '
390 ' FANTASIE IV
400 '
410 CLS:FOR X=25 TO 226 STEP 4:CIRCLE
(X,30),25,2:NEXT
420 FOR X=30 TO 162 STEP 4: CIRCLE
(226,X),25,2:NEXT
430 FOR X=226 TO 30 STEP -4: CIRCLE
(X,162),25,2:NEXT
440 FOR X=162 TO 90 STEP -4: CIRCLE
(30,X),25,3:NEXT
450 FOR X=30 TO 174 STEP 4: CIRCLE
(X,90),25,3:NEXT
460 FOR W=0 TO 5000:NEXT:RUN
```

```
[ ]= 32  [!]= 33  ["]= 34  [#]= 35  [$]= 36  [%]= 37  [&]= 38  [']= 39
[(]= 40  [)]= 41  [*]= 42  [+]= 43  [,]= 44  [-]= 45  [.]= 46  [/]= 47
[0]= 48  [1]= 49  [2]= 50  [3]= 51  [4]= 52  [5]= 53  [6]= 54  [7]= 55
[8]= 56  [9]= 57  [:]= 58  [;]= 59  [<]= 60  [=]= 61  [>]= 62  [?]= 63
[@]= 64  [A]= 65  [B]= 66  [C]= 67  [D]= 68  [E]= 69  [F]= 70  [G]= 71
[H]= 72  [I]= 73  [J]= 74  [K]= 75  [L]= 76  [M]= 77  [N]= 78  [O]= 79
[P]= 80  [Q]= 81  [R]= 82  [S]= 83  [T]= 84  [U]= 85  [V]= 86  [W]= 87
[X]= 88  [Y]= 89  [Z]= 90  [[]= 91  [\]= 92  []]= 93  [^]= 94  [_]= 95
[`]= 96  [a]= 97  [b]= 98  [c]= 99  [d]= 100 [e]= 101 [f]= 102 [g]= 103
[h]= 104 [i]= 105 [j]= 106 [k]= 107 [l]= 108 [m]= 109 [n]= 110 [o]= 111
[p]= 112 [q]= 113 [r]= 114 [s]= 115 [t]= 116 [u]= 117 [v]= 118 [w]= 119
[x]= 120 [y]= 121 [z]= 122 [{]= 123 [¦]= 124 [}]= 125 [~]= 126 []= 127
[á]= 160 [í]= 161 [ó]= 162 [ú]= 163 [ñ]= 164 [Ñ]= 165 [ª]= 166 [º]= 167
[¿]= 168 [⌐]= 169 [¬]= 170 [½]= 171 [¼]= 172 [¡]= 173 [«]= 174 [»]= 175
[▒]= 176 [▓]= 177 [▒]= 178 [│]= 179 [┤]= 180 [╡]= 181 [╢]= 182 [╖]= 183
[╕]= 184 [╣]= 185 [║]= 186 [╗]= 187 [╝]= 188 [╜]= 189 [╛]= 190 [┐]= 191
[└]= 192 [┴]= 193 [┬]= 194 [├]= 195 [─]= 196 [┼]= 197 [╞]= 198 [╟]= 199
[╚]= 200 [╔]= 201 [╩]= 202 [╦]= 203 [╠]= 204 [═]= 205 [╬]= 206 [╧]= 207
[╨]= 208 [╤]= 209 [╥]= 210 [╙]= 211 [╘]= 212 [╒]= 213 [╓]= 214 [╫]= 215
[╪]= 216 [┘]= 217 [┌]= 218 [█]= 219 [▄]= 220 [▌]= 221 [▐]= 222 [▀]= 223
[α]= 224 [β]= 225 [Γ]= 226 [π]= 227 [Σ]= 228 [σ]= 229 [µ]= 230 [τ]= 231
[Φ]= 232 [Θ]= 233 [Ω]= 234 [δ]= 235 [∞]= 236 [φ]= 237 [ε]= 238 [∩]= 239
[≡]= 240 [±]= 241 [≥]= 242 [≤]= 243 [⌠]= 244 [⌡]= 245 [÷]= 246 [≈]= 247
[°]= 248 [•]= 249 [·]= 250 [√]= 251 [ⁿ]= 252 [²]= 253 [■]= 254 [ ]= 255
```

IBM-karakterset

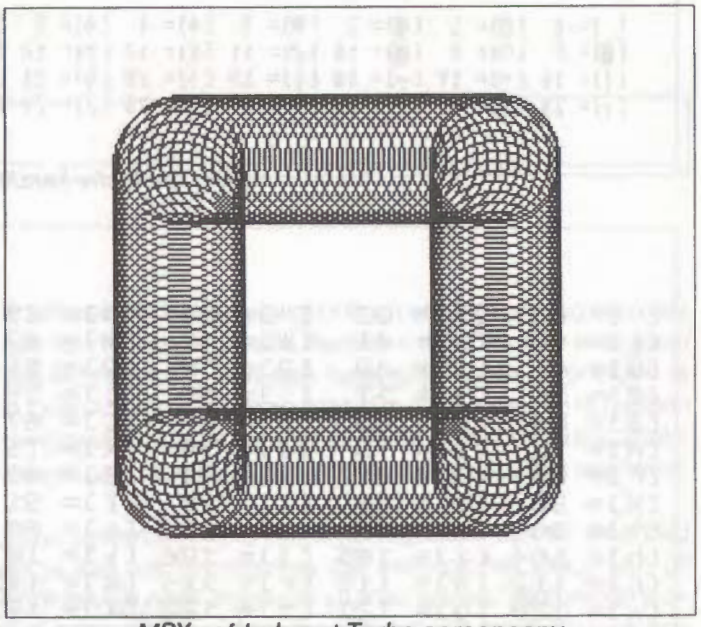
```
[ ]= 32  [!]= 33  ["]= 34  [#]= 35  [$]= 36  [%]= 37  [&]= 38  [']= 39
[(]= 40  [)]= 41  [*]= 42  [+]= 43  [,]= 44  [-]= 45  [.]= 46  [/]= 47
[0]= 48  [1]= 49  [2]= 50  [3]= 51  [4]= 52  [5]= 53  [6]= 54  [7]= 55
[8]= 56  [9]= 57  [:]= 58  [;]= 59  [<]= 60  [=]= 61  [>]= 62  [?]= 63
[@]= 64  [A]= 65  [B]= 66  [C]= 67  [D]= 68  [E]= 69  [F]= 70  [G]= 71
[H]= 72  [I]= 73  [J]= 74  [K]= 75  [L]= 76  [M]= 77  [N]= 78  [O]= 79
[P]= 80  [Q]= 81  [R]= 82  [S]= 83  [T]= 84  [U]= 85  [V]= 86  [W]= 87
[X]= 88  [Y]= 89  [Z]= 90  [[]= 91  [\]= 92  []]= 93  [^]= 94  [_]= 95
[`]= 96  [a]= 97  [b]= 98  [c]= 99  [d]= 100 [e]= 101 [f]= 102 [g]= 103
[h]= 104 [i]= 105 [j]= 106 [k]= 107 [l]= 108 [m]= 109 [n]= 110 [o]= 111
[p]= 112 [q]= 113 [r]= 114 [s]= 115 [t]= 116 [u]= 117 [v]= 118 [w]= 119
[x]= 120 [y]= 121 [z]= 122 [{]= 123 [¦]= 124 [}]= 125 [~]= 126 []= 127
[ ]= 160 [!]= 161 ["]= 162 [#]= 163 [$]= 164 [%]= 165 [&]= 166 [']= 167
[(]= 168 [)]= 169 [*]= 170 [+]= 171 [,]= 172 [-]= 173 [.]= 174 [/]= 175
[0]= 176 [1]= 177 [2]= 178 [3]= 179 [4]= 180 [5]= 181 [6]= 182 [7]= 183
[8]= 184 [9]= 185 [:]= 186 [;]= 187 [<]= 188 [=]= 189 [>]= 190 [?]= 191
[@]= 192 [A]= 193 [B]= 194 [C]= 195 [D]= 196 [E]= 197 [F]= 198 [G]= 199
[H]= 200 [I]= 201 [J]= 202 [K]= 203 [L]= 204 [M]= 205 [N]= 206 [O]= 207
[P]= 208 [Q]= 209 [R]= 210 [S]= 211 [T]= 212 [U]= 213 [V]= 214 [W]= 215
[X]= 216 [Y]= 217 [Z]= 218 [[]= 219 [\]= 220 []]= 221 [^]= 222 [_]= 223
[`]= 224 [a]= 225 [b]= 226 [c]= 227 [d]= 228 [e]= 229 [f]= 230 [g]= 231
[h]= 232 [i]= 233 [j]= 234 [k]= 235 [l]= 236 [m]= 237 [n]= 238 [o]= 239
[p]= 240 [q]= 241 [r]= 242 [s]= 243 [t]= 244 [u]= 245 [v]= 246 [w]= 247
[x]= 248 [y]= 249 [z]= 250 [{]= 251 [¦]= 252 [}]= 253 [~]= 254 []= 255
```

EPSON FX-80 karakterset

GWBASIC blijkt ook strenger te zijn wat betreft "spellingsregels" : Tussen het commando en de parameters is meestal een spatie verplicht :
CIRCLE(X,Y),50,4 wordt dus CIRCLE (X,Y),50,3
Na deze minimale ingrepen loopt het programma zonder problemen in GWBASIC.

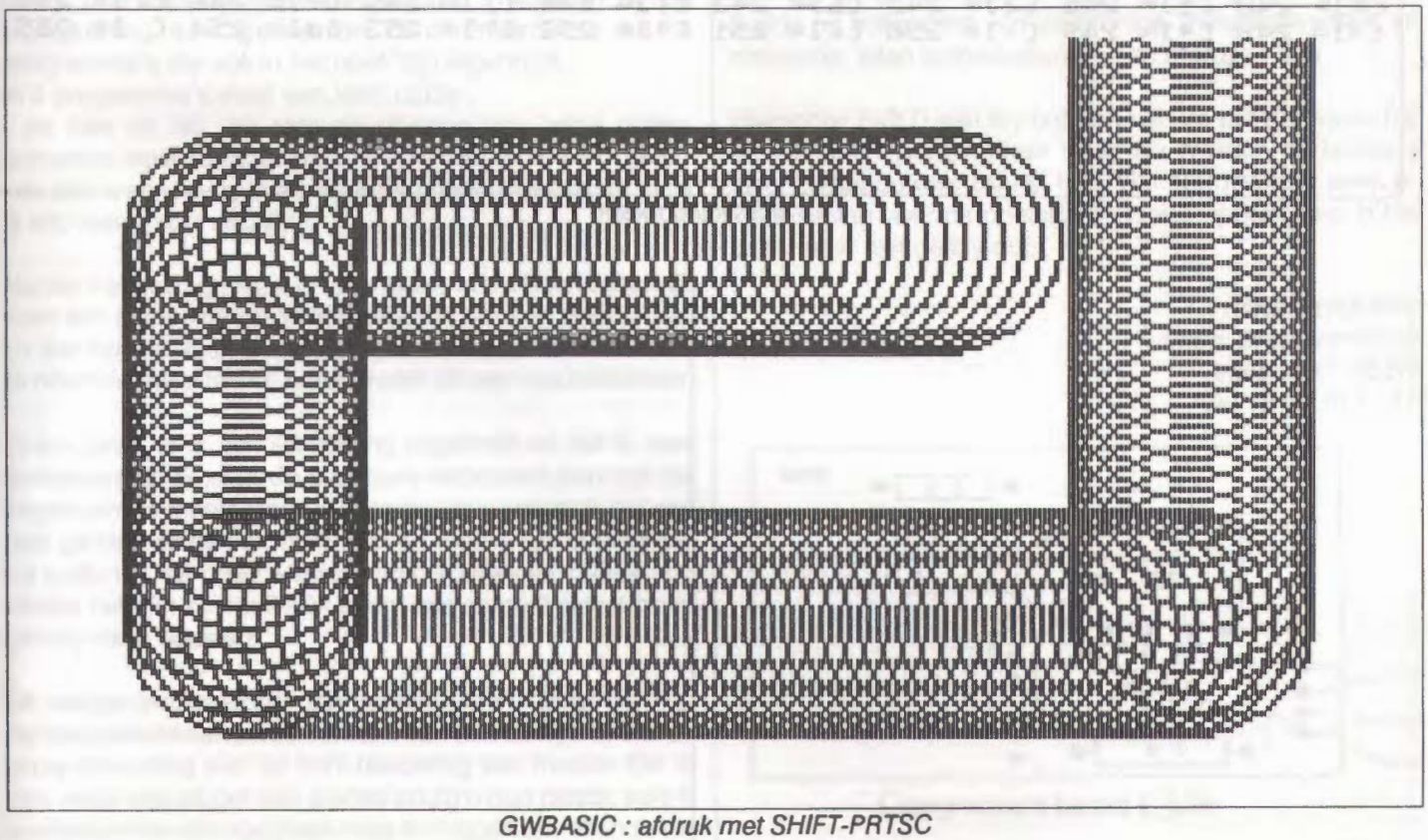
EDIT-tip

MSXBASIC en GWBASIC beschikken beide over een (identieke) full-screen editor. Als in een lijn een wijziging wordt aangebracht dient [RETURN] gegeven te worden om deze ook effectief in te voeren in het programma.
Dit kan het editeren wel eens vertragen. Met een beetje aandacht kan je de volgende truc toepassen:
Breng op het scherm alle nodige wijzigingen aan, zonder telkens [RETURN] te drukken. Je kan zo gemakkelijk van lijn tot lijn springen zonder steeds opnieuw vooraan in een lijn te moeten beginnen. Let wel op dat je INSERT aanzet als je tekst wil invoegen, en ook als je een lijn wil langer maken. Zo voorkom je dat je lijnen aan mekaar gaat plakken, met vervelende gevolgen.
Als alle wijzigingen voor dit scherm klaar zijn, ga dan op het bovenste lijnnummer staan en druk telkens [RETURN], tot je beneden bent. Alle lijnen zijn nu in hun gewijzigde vorm ingevoerd !



MSX : afdruk met Turbo screencopy

In een volgende aflevering bestuderen we de verschillen in TIME en TIME$-notatie.



GWBASIC : afdruk met SHIFT-PRTSC

```
[ ]= 0  [☺]= 1  [●]= 2  [♥]= 3  [♦]= 4  [♠]= 5  [♠]= 6  [·]= 7
[■]= 8  [○]= 9  [■]= 10 [♂]= 11 [♀]= 12 [♪]= 13 [♫]= 14 [*]= 15
[┼]= 16 [┴]= 17 [┬]= 18 [┤]= 19 [├]= 20 [┼]= 21 [│]= 22 [─]= 23
[┌]= 24 [┐]= 25 [└]= 26 [┘]= 27 [╳]= 28 [/]= 29 [\]= 30 [+]= 31
```

MSX : grafische karakters, CHR$(1) + (64 + CODE)

```
[ ]= 32  [!]= 33  ["]= 34  [#]= 35  [$]= 36  [%]= 37  [&]= 38  [´]= 39
[(]= 40  [)]= 41  [*]= 42  [+]= 43  [,]= 44  [-]= 45  [.]= 46  [/]= 47
[0]= 48  [1]= 49  [2]= 50  [3]= 51  [4]= 52  [5]= 53  [6]= 54  [7]= 55
[8]= 56  [9]= 57  [:]= 58  [;]= 59  [<]= 60  [=]= 61  [>]= 62  [?]= 63
[@]= 64  [A]= 65  [B]= 66  [C]= 67  [D]= 68  [E]= 69  [F]= 70  [G]= 71
[H]= 72  [I]= 73  [J]= 74  [K]= 75  [L]= 76  [M]= 77  [N]= 78  [O]= 79
[P]= 80  [Q]= 81  [R]= 82  [S]= 83  [T]= 84  [U]= 85  [V]= 86  [W]= 87
[X]= 88  [Y]= 89  [Z]= 90  [[]= 91  [\]= 92  []]= 93  [^]= 94  [_]= 95
[`]= 96  [a]= 97  [b]= 98  [c]= 99  [d]= 100 [e]= 101 [f]= 102 [g]= 103
[h]= 104 [i]= 105 [j]= 106 [k]= 107 [l]= 108 [m]= 109 [n]= 110 [o]= 111
[p]= 112 [q]= 113 [r]= 114 [s]= 115 [t]= 116 [u]= 117 [v]= 118 [w]= 119
[x]= 120 [y]= 121 [z]= 122 [{]= 123 [¦]= 124 [}]= 125 [~]= 126 [▲]= 127
[á]= 160 [í]= 161 [ó]= 162 [ú]= 163 [ñ]= 164 [Ñ]= 165 [ª]= 166 [º]= 167
[¿]= 168 [⌐]= 169 [¬]= 170 [½]= 171 [¼]= 172 [¡]= 173 [«]= 174 [»]= 175
[Ã]= 176 [ã]= 177 [ĩ]= 178 [ĩ]= 179 [õ]= 180 [õ]= 181 [Õ]= 182 [ũ]= 183
[Ÿ]= 184 [ÿ]= 185 [¾]= 186 [^]= 187 [◇]= 188 [%]= 189 [¶]= 190 [§]= 191
[▁]= 192 [▚]= 193 [▄]= 194 [▔]= 195 [▬]= 196 [■]= 197 [▌]= 198 [▟]= 199
[█]= 200 [ ]= 201 [▌]= 202 [▨]= 203 [▧]= 204 [▼]= 205 [▲]= 206 [▶]= 207
[◀]= 208 [▒]= 209 [▶]= 210 [▀]= 211 [▗]= 212 [▘]= 213 [▄]= 214 [☼]= 215
[△]= 216 [‡]= 217 [ω]= 218 [█]= 219 [▄]= 220 [▌]= 221 [▐]= 222 [▀]= 223
[α]= 224 [β]= 225 [Γ]= 226 [π]= 227 [Σ]= 228 [σ]= 229 [µ]= 230 [τ]= 231
[Φ]= 232 [θ]= 233 [Ω]= 234 [δ]= 235 [∞]= 236 [ø]= 237 [∈]= 238 [∩]= 239
[≡]= 240 [±]= 241 [≥]= 242 [≤]= 243 [⌠]= 244 [⌡]= 245 [÷]= 246 [≈]= 247
[°]= 248 [•]= 249 [·]= 250 [√]= 251 [ⁿ]= 252 [²]= 253 [▮]= 254 [ ]= 255
```

MSX karakterset

TEKENPLANK

Bij deze volgt de bespreking van de tekenplank gemaakt volgens het boek 'Elektronicaprojecten voor MSX-computers'.

Het allereerste wat men in dit boek bespreekt is de joystick-ingang. Deze zal men verder in het boek steeds gebruiken als interface. Behalve voor meer ingewikkelde schakelingen, dan stapt men over op de printeruitgang, evenwel met samenspel van de joystickinterface daar waar dit nodig is.

Voor de tekenplank heeft men 2 timer IC's nodig (555). Deze IC's dienen dan op een gaatjesprint of voor de verder gevorderden op het gebied van electronica, een zelf geëtste print, te worden samen gebouwd tot een simpel maar accuraat tellertje.

Van dit printje heeft men 2 exemplaren voor de tekenplank nodig. Dus alles wordt 2 maal gebouwd. Het testen van de schakeling kan gebeuren door meerdere basic programma's die ook in het boek zijn afgedrukt.
In 2 programma's staat een klein foutje :
- de 1ste op blz. 19 voor de uitlezing van beide potentiometers regel 70 moet zijn : 70 A=PDL(1) : B+PDL(A3)
- de 2de is een foutje in de regelnummering, de laatste regel is 400 maar moet 460 zijn.

Nu dan het belangrijkste de tekenhaak. In het boek beschrijft men een scharnier dat wordt gebruikt om een deurtje van bv een keukenkast, of het schrijfblad van een bureel open te houden. Volgens het boek noemt dit een meubelschaar.

In elk geval er is een afbeelding afgedrukt en die is zeer verhelderend. Ik raad de eventuele nabouwer aan om die zogenoemde meubelschaar te gebruiken, omdat ik het niet heb gedaan en de verhouding in de maten een belangrijke rol spelen in het goed werken van de tekenplank. Ook nog omdat het niet zo makkelijk is om iets dergelijks met goed gevolg na te bouwen.

Dit stangenstelsel wordt dan gedemonteerd zodat men op de draaipunten een potentiometer kan aanbrengen, liefst de grote uitvoering niet de mini uitvoering wat mooier lijkt te zijn, maar wat inboet aan sterkte en zo'n dun plastic asje is snel gebroken om nog maar niets te zeggen betreffende het inwendige. De grote modellen verdragen wel iets meer.

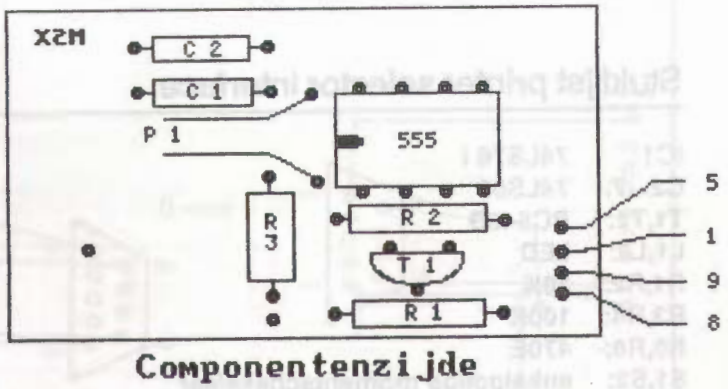
Ik zelf heb het kleine type genomen omdat ik er nog een paar had rondslingeren, mede daardoor weet ik maar al te goed wat ik hier vertel. Wat ook belangrijk is om tekeningen te kunnen kopiëren is om zo weinig mogelijk speling te hebben ten opzichte van de stangetjes en de potentiometers, zoniet zal het zeer moeilijk zijn om een goed resultaat te bekomen.

Eens samengebouwd komt het ogenblik van de waarheid en ja met het programma dat men op het einde van dit hoofdstukje heeft afgedrukt kunnen zowaar tekeningen worden gemaakt. Men kan natuurlijk zelf een programma schrijven om het tekenen aan uw wensen volledig aan te passen, alle nodige gegevens betreffende het berekenen zijn voldoende uitgelegd in het boek. Natuurlijk mag men niet hetzelfde resultaat verwachten zoals van een professionele tekenplank maar voor het kleine en eenvoudig werk voldoet dit goed, om nog niet te spreken van het prijsverschil.

In elk geval kan U nu de winteravonden in aantocht zijn uw kunde op het gebied van elektronica, metaalbewerking en computer, laten samenvallen op een prettige wijze.

Hieronder treft U een lay-out aan van de printopbouw, het is een klein printje maar voor de minder onderlegde electronikabouwers zal het toch zijn nut bewijzen denk ik. De tekening voor de printsporen is wel opgenomen in het boek maar niet de lay-out.

SMULDERS Dirk
Frans Broersstraat 23
2800 MECHELEN
Tel. (015) 20.84.77



PRINTER SELECTOR INTERFACE

Soms kan het een probleem zijn wanneer je als MSX'er wil gebruik maken van meerdere afdrukelementen. Aangezien onze MSX-computer slechts beschikt over één paralleluitgang (centronics), dient men telkens connectoren te verwisselen bij gebruik van bijv. 1 printer en 1 plotter. Een tamelijk vervelende situatie wanneer beide bij eenzelfde programma gebruikt worden.

Dit probleem kan men op 2 manieren oplossen: mechanisch (dwz. met behulp van schakelaars) of electronisch.
Wij kiezen voor de tweede methode aangezien deze meer mogelijkheden biedt. We kunnen dan de selector softwarematig, dit is via de computer, aansturen. Dit zullen we in hetvolgende artikeltje bespreken.

Opbouw van de schakeling

In fig.1 is het schema weergegeven. De schakeling bestaat uit een selector (IC1) en schakelaars voor het sturen van de respectievelijke kanalen (IC2...7).
Het sturen van IC1 kan op twee manieren: manueel mbv. momentschakelaars of softwarematig via CL1/CL2. Dit zal het omklappen van de selector (flip-flop) teweegbrengen. Bij indrukken van S1 (of aansturen van CL1) wordt de flipflop gepreset zodat de uitgang Q/ (pen 10,14) hoog wordt. Hierdoor zal L1 oplichten en IC2,3,4 de informatie van de ingang naar de uitgang doorgeven. Uitgang Q (pen 11,15) van IC1 daarentegen heeft een laag niveau zodat L2 gedoofd is en IC5,6,7 een laagniveau aan de uitgang hebben. Hierdoor ontvangt de op deze uitgang aangesloten printer geen informatie.

Stuklijst printer selector interface:

IC1:	**74LS76 I**
C2...7:	**74LS08**
T1,T2:	**BC548B**
L1,L2:	**LED**
R1,R2:	**10K**
R3,R4:	**100K**
R5,R6:	**470E**
S1,S2:	**enkelpolige momentschakelaar**
Kabel:	**2x 1m printerkabel (min 11 aders)**
	1m 4 aderig snoer
Connector:	**2x centronics 36 pen (man.) + afscherm-kapjes**
	1x centronics 36 pen (vrouw.)
	1x D-sub 9 pen (man.)
Kastje:	**TANDY nr. 270-210**

Geschatte bouwprijs: 1250 BF

Indien er geen gebruik zal gemaakt worden van de OUTPUT-INTERFACE , zie volgend artikeltje , kan de printer selector toch gebruikt worden. Er zal dan enkel manueel geschakeld kunnen worden. De voeding voor onze schakeling zullen we dan halen uit een joystick uitgang.

Opgelet:

- Max. stroom via joystick bedraagt 50 mA.
- Enkel gebruik maken van LS of beter HCT type IC's.
- + verbinden met pen 5 en - met pen 9 van een D-sub 9 pen (vrouw.) connector (CL1 en CL2 niet gebruiken.)
- Er kan ook een afzonderlijke +5 V voeding gebruikt worden.

OUTPUT-INTERFACE CARTRIDGE

Deze eenvoudige 8-bits output-interface wordt gebruikt voor het aansturen van de CL1/CL2 lijnen van de selector. Afhankelijk van de gekozen bouwsteen voor IC8 (zie fig.4) verkrijgen we een input-output-interface of een output-interface met al of niet geinverteerde uitgangen die voor tal van sturingen toepasbaar is.

Rond IC9 wordt een adresselectie opgebouwd. Het met de DIP-switches ingestelde adres wordt vergeleken met de adreswaarde door de computer op adreslijn A0...A7 gezet. Indien beide gelijk zijn wordt IC8 (pen 19) geactiveerd zodat de data op D0...D7 aan de uitgang verschijnt.

Opgelet: de hier toegepaste schakeling geeft slechts een puls op de betreffende datalijn! Indien de informatie een langere tijd aan de uitgang dient behouden te blijven, zal men een andere schakeling dienen te bouwen, bv. met Z80-PIO.

Het aansturen van de cartridge.

De adresinstelling van de cartridges gebeurt met de DIP-switches: bv. het adres volgens fig. 6. Het hier ingestelde adres komt overeen met de hexadecimale notatie &H06.
Verbinden we nu CL1 van de printerselector met uitgang D0 en CL2 met D1. Wanneer nu in het programma de instructie OUT &H06,&H02 gezet wordt, dan zal bij het uitvoeren van deze programma regel een puls verschijnen op uitgang D1 en schakelt de printerselector: L2 licht op. Bij het geven van een PRINT-instructie zal de printer op kanaal 2 inwerking treden.
Bij uitvoer van de instructie OUT &H06,&H01 schakelt de printerselector om. L1 gaat branden en de hierop volgende PRINT-instructie's worden uitgevoerd door de printer van kanaal 1.
De overige datalijnen van onze cartridge kunnen we voor andere toepassingen aanwenden. Bijvoorbeeld OUT &H06,&HC9 zal gelijktijdig een puls geven op D0,D3,D6 en D7.

Als behuizing voor onze output-cartridge kunnen we een doosje van een compact-muziekcassette gebruiken. Dit doosje heeft namelijk dezelfde afmetingen als een cartridge-cassette zodat het precies in een cartridge-slot van de MSX-computer past.
We halen de nokken uit het deksel en maken de nodig openingen in het doosje zoals aangegeven in fig. 7. Opdat het deksel gesloten kan worden, is het nodig dat de IC's rechtstreeks in de print gesoldeerd worden.

Stuklijst output-interface cartridge:

IC8: 74LS541 (74LS540: geïnverteerde uitgang 74LS245: input/output)
IC9: 74LS688
R7..R14: 10K
C1,C2: 100n
S3: 8 pen DIP-switch
Connector: 1x d-sub 9 pen (vrouw.)
Kastje: compact-muziekcassette

Geschatte bouwprijs: 500 BF

Veel succes bij de bouw van deze interface. Hopelijk draagt dit kleine stukje electronica bij tot het vlotter en aangenamer gebruik van uw randapparatuur.

Printen kunnen besteld worden bij :

Wim De Roover Negenbunders 6 B-3150 Heist O/D Berg

* Printer selector : 400 BF/fl 23
* Output cartridge : 450 BF/fl 26
(alleen vooruitbetaling per cheque)

Wim en Dirk De Roover.

FIG. 3

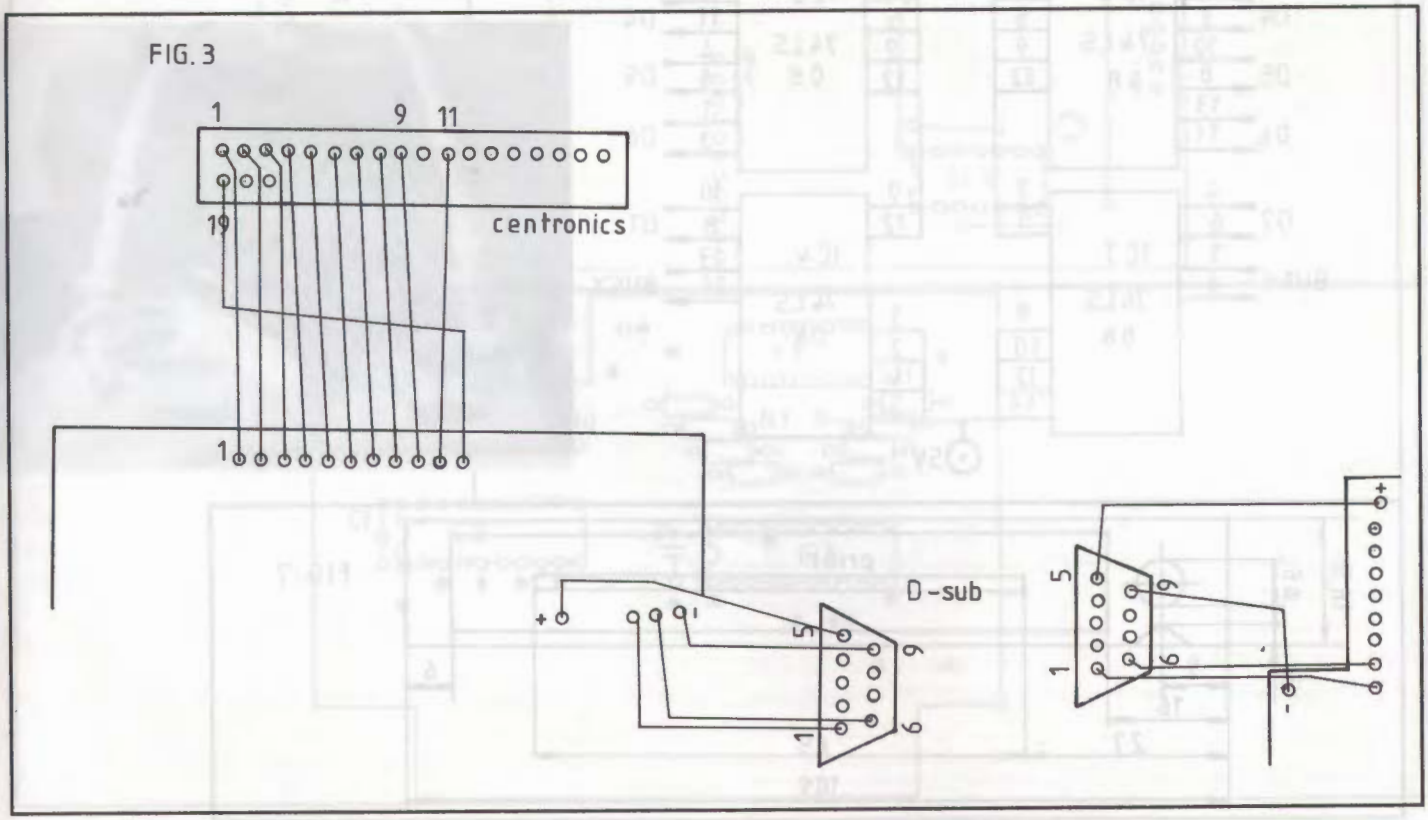
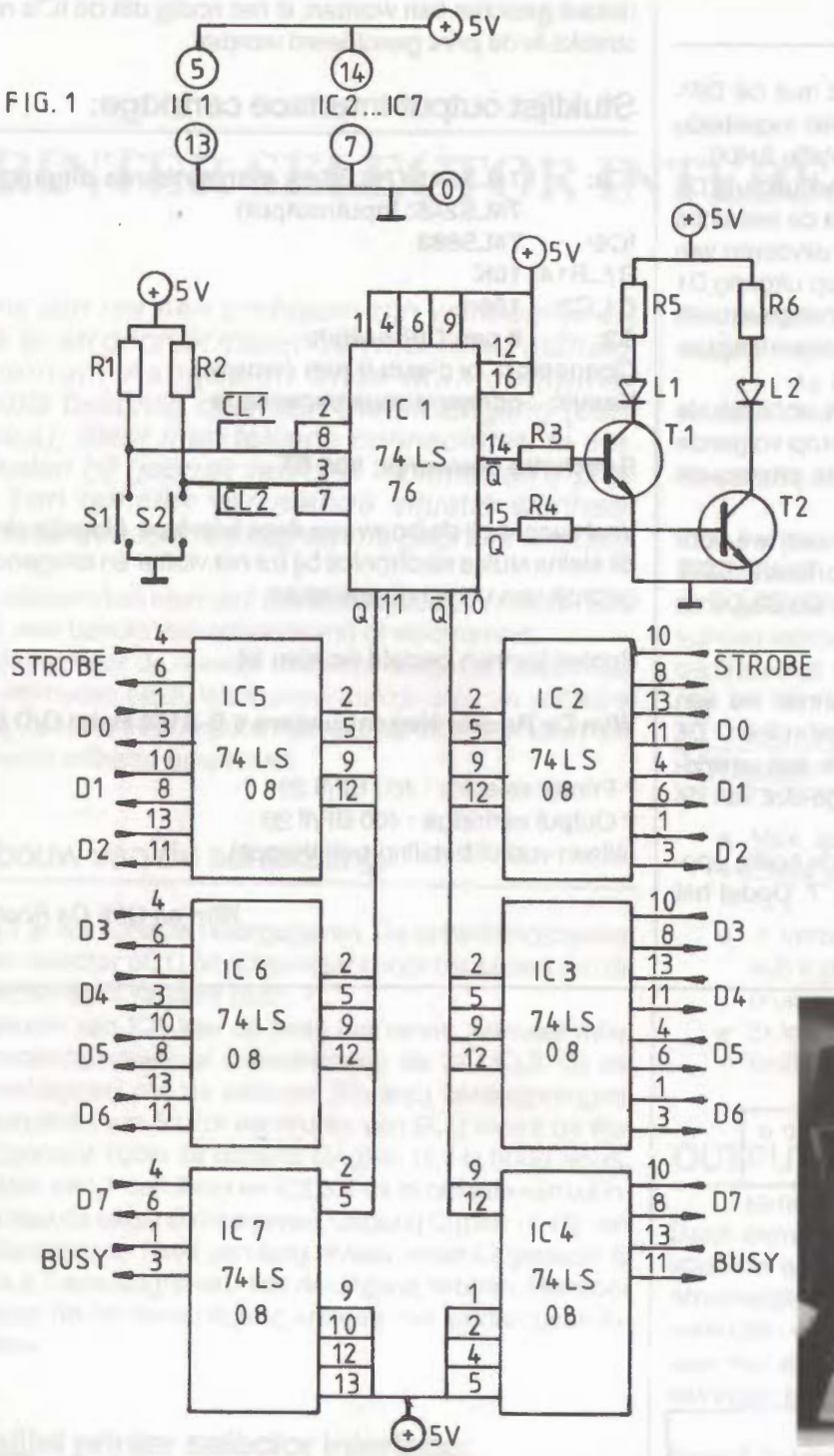
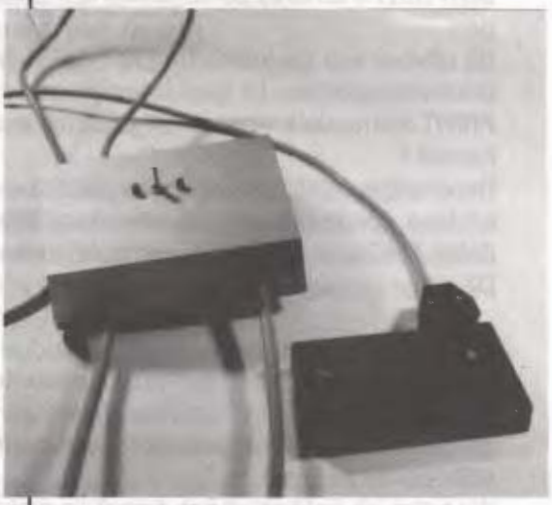
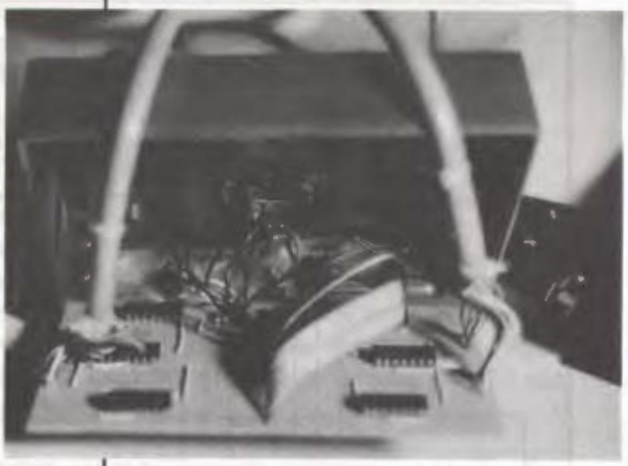


FIG. 1


5V
IC 1
IC 2 ... IC 7
R1
R2
R3
R4
R5
R6
L1
L2
T1
T2
S1
S2
CL 1
CL 2
IC 1
74 LS
76
STROBE
D0
D1
D2
D3
D4
D5
D6
D7
BUSY
IC 5
IC 2
IC 6
IC 3
IC 7
IC 4
74 LS
08





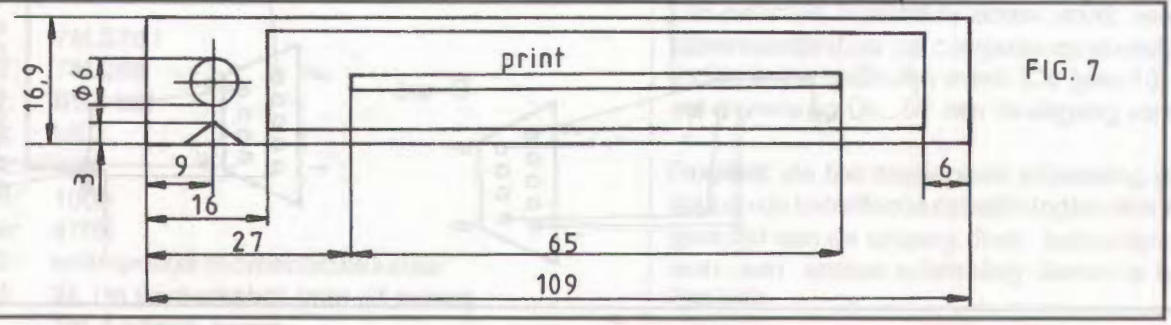
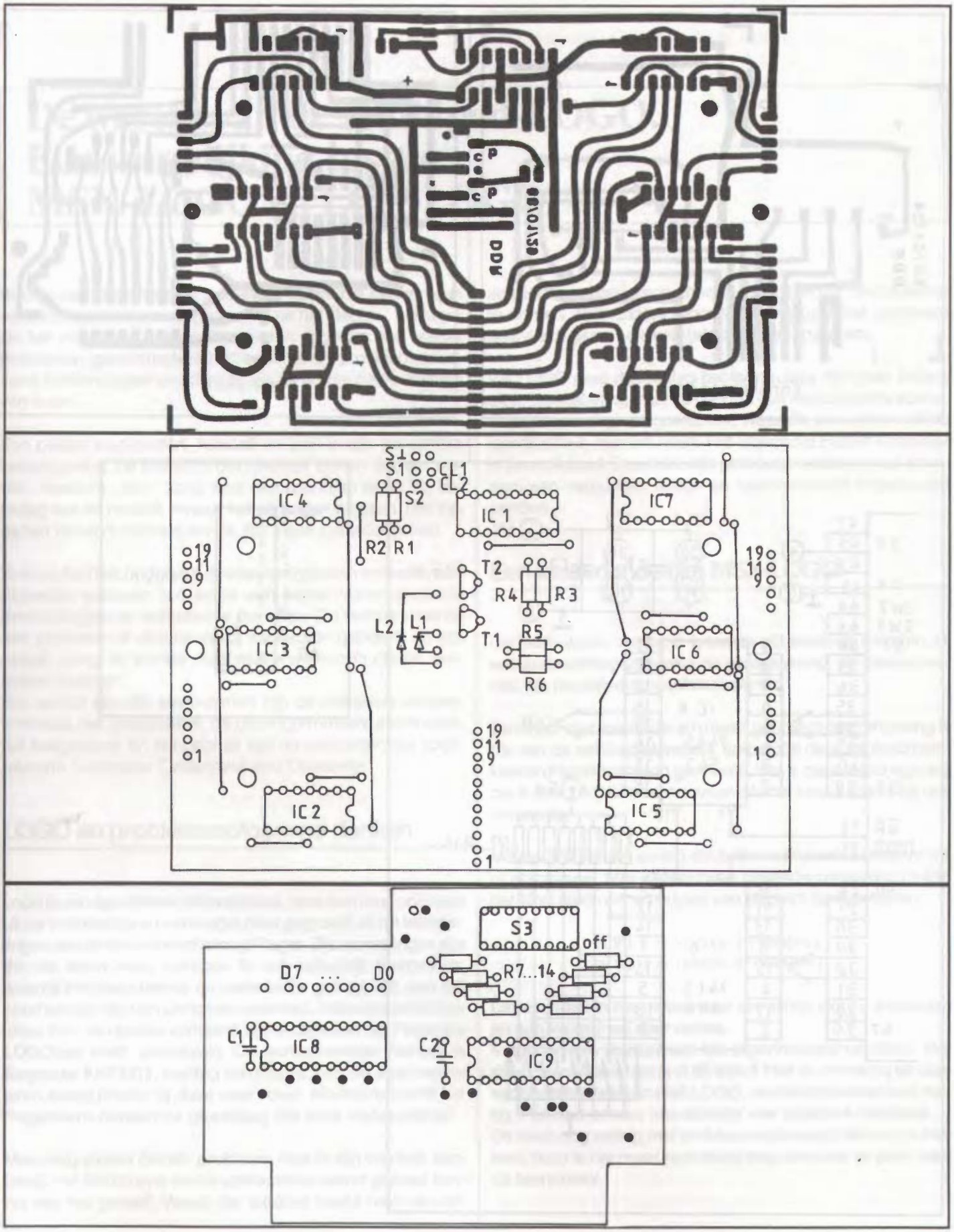
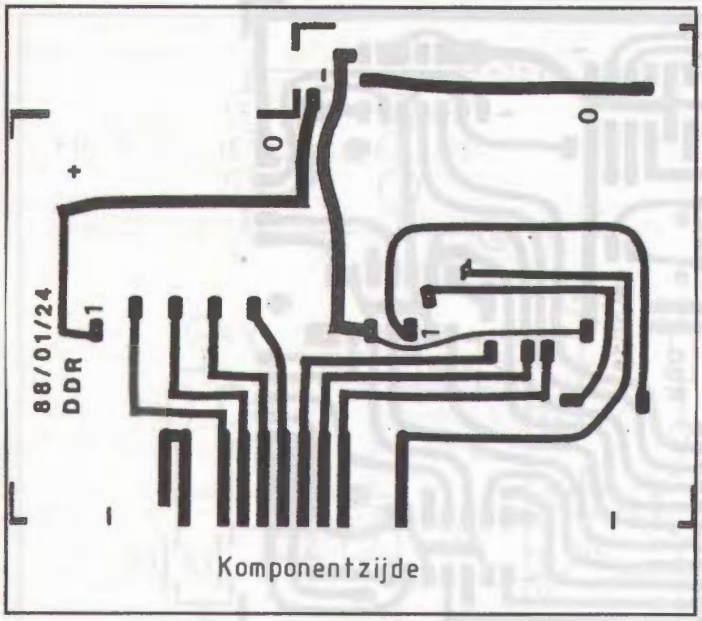
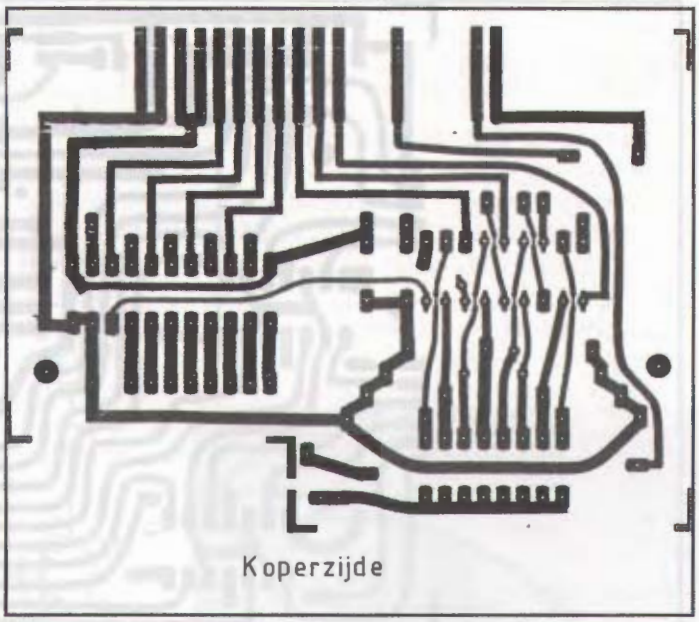

print
16,9
ø6
3
9
16
27
65
109
6

FIG. 7


S1
S1
CL1
CL2
S2
IC 4
IC 1
IC7
R2 R1
T2
R4
R3
L2
L1
T1
R5
R6
IC 3
IC 6
IC 2
IC 5
S3
off
D7 ... D0
R7...14
C1
IC 8
C2
IC 9


88/01/24
DDR
Komponentzijde


Koperzijde

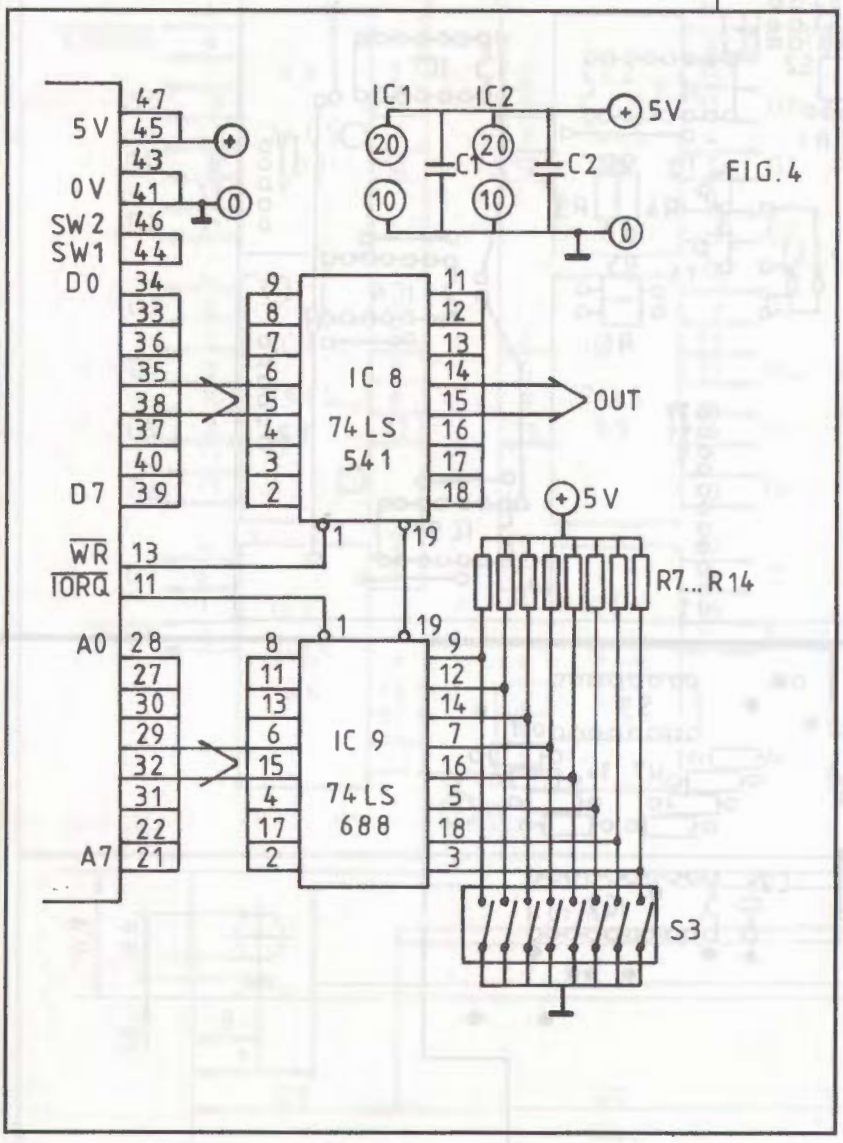

5V
0V
SW2
SW1
D0
D7
WR
IORQ
A0
A7
IC1
IC2
C1
C2
IC 8
74LS
541
IC 9
74LS
688
OUT
R7... R14
S3

FIG. 4

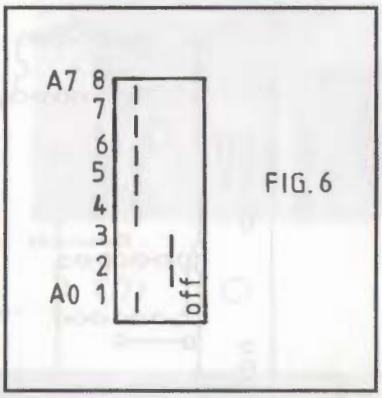

A7
A0
off

FIG. 6

De wereld van de schildpad-LOGO. Een vergelijking tussen MSX-LOGO en LCN-LOGO.

Binnen de wereld van het onderwijs neemt het denken een centrale plaats in. Leerlingen zowel uit het kleuter-, het lager als het voortgezet onderwijs, worden dagelijks met allerlei problemen geconfronteerd. Produktief en probleemoplossend denken liggen erg dicht bij elkaar en zijn beide vormen van leren.

Een peuter manipuleert, handelt en leert in zijn natuurlijke leefomgeving. De interactie die optreedt tussen kind en 'wereld - rondom - zich ' zorgt voor een spontaan leren. Via beleving van de realiteit ervaart het zijn eigen lichaam. Het benadert de werkelijkheid vanuit zijn eigen persoonlijkheid.

Evenzo kan het onderwijs diverse leergehelen en basisvaardigheden aanleren ten einde een aantal vooropgestelde doelstellingen en attitudes te bereiken. De manier waarop een probleem of situatie wordt naar voren gebracht en opgelost, hangt af van de didactische werkvorm die de leerkracht hanteert.
Een aantal van die werkvormen zijn de klassikale doceermethode, het groepswerk, de geprogrammeerde instructie, het leergesprek en het gebruik van de computer; het zogenaamde 'Computer Ondersteunend Onderwijs'.

LOGO en probleemoplossend denken.

Logo is een specifieke onderwijstaal, voor een deel ontstaan uit de informatica en een ander deel gegroeid uit de leerstellingen van de kinderpsycholoog Piaget. Zijn opvattingen zijn dat het leren moet verlopen in een natuurlijk leerproces, waarbij intuïtieve kennis en zelfreflectie gekoppeld, aan het voorhanden zijn van concreet materiaal, leiden tot inzicht en leren. Het zou echter verkeerd zijn te beweren dat Piaget de LOGOtaal heeft ontwikkeld. De echte bezieler hiervan is **Seymour PAPERT**, leerling van Piaget. Aan het eind van de jaren zestig bracht hij deze naar voren. Nochtans vormt het Piagetiaans denken de grondslag van deze onderwijstaal.

Men mag stellen dat elk probleem zich in zijn totaliteit aanbiedt. Het kind zowel als de volwassene neemt globaal kennis van het geheel. Vanuit die totaliteit tracht men via verschillende oplossingsmethoden tot de kern en de oplossing te komen. Met andere woorden: men gaat het probleem analytisch gaan bekijken (**algoritmisch denken**).

Via LOGO gaat de leerling het leerproces zelf gaan leiden, dit in tegenstelling met het gebruik van verschillende educatieve toepassingsprogramma's, waar de computer zelf de leersituatie in handen heeft. Het object dat Papert voorstaat is de schildpad. Daarmee kan getekend worden en zal spontaan een natuurlijke relatie en communicatie opgebouwd worden.

Een kindvriendelijke MSX-LOGO.

De MSX-LOGO omvat in essentie vijf **leeromgevingen**, te weten de schildpadwereld, de woordwereld, de getallenwereld, de muziek-en de animatiewereld.

De direct interessantste en meest gebruikte leeromgeving is die van de **schildpadwereld**, soms ook de schildpadmeetkunde of schildpadland genoemd. Het is deze leeromgeving die ik hier kort wil bespreken en die het best aansluit bij ons onderwijs.

MSX-LOGO is kort en erg duidelijk in zijn basiscommando's of primitieven. Met slechts twee gekende commando's kan het kind reeds de schildpad een vierkant laten tekenen.

vt 20 (= vooruit 20 pasjes)
re 90 (= rechts 90 draaien)

Dit resulteert in het trekken van een lijntje van 20 eenheden en een kwartdraai naar rechts.
Volgt dan een stadium van het experimenteel uittesten. Wat doet de schildpad als ik vt 85 intik ? Hier is, vooral bij de aanvang van het werken met LOGO, de **directmodus** heel nuttig. Het kind ervaart onmiddellijk een positieve feedback.
Dit heeft nog weinig met probleemoplossend denken te maken. Toch is het maar een kleine stap om over te gaan naar de **leerkamer**.

Ook hier is het gemak van een nederlandstalige onderwijs- en programmeertaal met klare opdrachten bijzonder welkom.
Alle instructies naar de MSX schildpad toe zijn eenvoudig. Als voorbeeld bij het '?' tikken we in : ' **leer vierkant** '. MSX-LOGO beantwoordt met het openen van de **leermodus** :

```
leer vierkant ...
eind
```

De cursor plaatst zich tussen 'leer' en 'eind' zonder dat we dit hoeven in te tikken. We kunnen nu beginnen invoegen.

Subprocedures in superprocedures zijn mogelijk; m.a.w. in de leermodus gaan we aangeleerde handelingen (tekeningen, begrippen) gaan toepassen als onderdelen in een groter geheel. Bijvoorbeeld : het aan de computer geleerde "vierkant" kan nu gebruikt worden als onderdeel van een "huis". Verschillende vierkanten vormen de ramen van een huis. "Huis" kan nu dienen om een "huizenrij" of een "straat" op te bouwen. Steeds zullen bij het oproepen van een **superprocedure** in MSX-LOGO (bijvoorbeeld "straat"), de subprocedures (hier "vierkant", "huis", "huizenrij") als een gedefinieerd object worden beschreven. Dit is een inwendige evaluatie van alle bestaande onderliggende procedures.

Recursie is het steeds hernemen van een opdracht. Die opdracht kan een procedure zijn. Door condities te gaan inbouwen, en dat kan al met de 'als'- voorwaarde onmiddellijk na het heroproepen van de procedure, kan je het programma onder controle krijgen. Ook wel handig is dat van een volledig tekstscherm naar een teken- en tekstscherm kan overgegaan worden.

vb. zettekst 20 = begin tekst op lijn 20. Van de 24 lijnen op het scherm worden 20 voorbehouden voor het tekenen en 4 voor de commando's.
zettekst 0 = begin tekst bovenaan op het scherm.

Nog zo'n handigheidje is de mogelijkheid die MSX biedt om toetsen te **herdefiniëren**. We kunnen achter elke toets ieder gewenste opdracht steken. Zo kan de leerkracht achter de toets 'r' de opdracht "rechts 90" steken. Dus door het indrukken van de toets 'r' draait de schildpad 90 naar rechts. Dit is uiteraard zeer interessant voor kinderen die nog moeilijkheden hebben met de complexiteit van het toetsenbord. Dergelijke ingreep op het programma wordt ook wel ééntoetsprogramma genoemd. Een ééntoets-programma wordt tevens dikwijls gebruikt als vooroefening en bij zeer jonge kinderen.

Een heel sterk punt bij MSX vind ik de eenvoud om **variabelen** aan te brengen bij onze oudste lagere schoolkinderen. Een zo belangrijk onderdeel, zowel vanuit pedagogisch als programmatisch standpunt, garandeert het succes waarmee ze later variabelen zullen gebruiken in klassieke of moderne wiskunde.
Ten slotte wil ik de LOGO-**editor** vermelden voor het definieren (uiteraard) en editen (verbeteren) van procedures. Je kan zelfs eigenhandig één of meerdere vormen gaan bepalen (creëren) in de plaats van de gekende schildpad. Je komt al erg aardig in de buurt van de animatiefilm, als je ze ook nog gelijktijdig wil laten bewegen.

LCN LOGO voor het voortgezet onderwijs.

Het werken met **LCN LOGO** is niet zo eenvoudig als de hierboven besproken **MSX LOGO**. Met MSX LOGO en een glimp af en toe naar het handboek, ben je zo gestart. LCN LOGO daarentegen is een combinatie van **sterk gestructureerd programmeren** en A.I.-programmeren (A.I. = artificiële intelligentie). Met dit laatste bedoel ik dat LCN elke syntax- of structuurfout aanwijst, nog voor de gebruiker door heeft wat er hapert. Er bestaat een levendige interactie tussen editor (de SED, waarover straks meer) en de gebruiker.
Met LCN (versie 1.0) word je letterlijk gedwongen structureel en logisch te denken. Het was toch PAPERT zelf die pleitte voor leesbaarheid. Daaraan voldoet deze programmeertaal uitstekend. Deze LOGO versie sluit bijzonder dicht aan bij LISP, zoals Harry Pinxsteren uit het Nijmeegse LOGO-team beweert : "LOGO is LISP, maar zonder haakjes". Wie PASCAL kent, weet zich ook al vlug thuis in deze onderwijstaal. Voor wie als eerste kennis maakt met LCN (een goede keuze overigens om mee te beginnen), zal toch even moeten doorbijten. De handleiding is lijvig en niet zo direct naar de beginner toe geschreven (ongeveer 400 blz; een meer uitgebreide bespreking vormt een afzonderlijk vervolg). De nadruk ligt op de beschrijving van de kenmerken van LCN LOGO, lees je in het voorwoord. Toch heeft LCN alles aan boord. LCN leert, meer nog dan MSX, stapsgewijs en programmatisch denken.

Na het openingsscherm komt de gebruiker standaard in de editor. Verder is er ook nog een tekstscherm en een grafisch scherm. Werk je op het eerste niveau, dus in het **editscherm**, dan kan je het programma invoeren. De statusregel onderaan vertelt je precies wat er gebeurt : tekst invoer toestand. Zodra we de eerste letter intypen, verandert de regel van kleur (oranje) en vormt een vlak. Per regel kan je maar één elementair datatype (of gegevenstype) intikken. Met RETURN ga je naar de volgende regel, het volgende niveau, één stap rechts. Een voorbeeld maakt veel duidelijk.

```
HERHAAL 4
    vooruit 30
    rechts 90
```

Een dubbele RETURN na vooruit 30 zorgt ervoor dat je je gegevens kan inbrengen onder de 30. In LCN LOGO worden alle primitieven die geen bewerkingen inhouden (op elementaire datatypen) met hoofdletters geschreven. Vaste afkortingen gelden alleen voor de meest gebruikte primitieven. Je tikt 1 of 2 letters in, gevolgd door een spatie en de Structor EDitor (SED) vult automatisch aan.

vt (spatie) = vooruit
re (spatie) = rechts
M (spatie) = MAAK
H (spatie) = HERHAAL

Leesbaarheid staat voor alles: een LOGO structuur die visueel en in vlakken of blokken zijn inhoud kleurt.

Structuren maken we ook zichtbaar door met de cursor door de vlakken te wandelen. De mededeling onder in de statusregel geeft aan dat we in de wandeltoestand zijn. Met andere woorden, door te wandelen kunnen we duidelijk zien dat vlakken een hiërarchie vormen. Wijzen we HERHAAL aan dan is de hele structuur gekleurd (groen): dus ook de vlakken vooruit en rechts en de vlakken die weer door die vlakken worden gedomineerd: de vlakken 30 en 90. Het vlak dat begint met HERHAAL omvat alle andere onderliggende vlakken.

In de **leeromgeving** kunnen we een funktie met variabelen declareren, hetzij globale variabelen, hetzij lokale variabelen. We gaan als het ware een recept opstellen met gegeven ingrediënten. Let ook op de prefix notatie; terug een voorbeeld van algoritmisch of gestruktureerd denken: 'Wat gaan we doen ? Optellen. Wat tellen we op ? Het eerste en het tweede element.'

```
MAAK   optelling
       FUNK MET      getal1
                     getal2
                     plus      maal     getal1
                                        getal1
                               maal     getal2
                                        getall2
```

```
Maak de functie 'optelling
met 2 variabelen, nl
getal1 en getal2
op de expressie, nl
(getal1 maal getal1)
plus
(getal2 maat getal2)
```

SED, of de (structuur) editor van LCN LOGO is speciaal gemaakt voor het werken met structuren. Tekst en vlakken kunnen veranderd of aangevuld worden (knippen, copiëren, plakken, openen, ...). Achter de funktietoetsen zit een heel hulpprogramma met diverse kommandotoetsen verborgen.

Het **tekstscherm** is het tweede scherm dat ons ten dienste staat (evaluatiescherm). Alles wat we intypen kunnen we onmiddellijk laten runnen en/of beoordelen door het eenvoudig aan te wijzen. We hoeven alleen maar met de cursor naar een blok (onderdeel) te gaan (vb een vlak) en op F10 te drukken. Handig om fouten in een programma op te zoeken. Het hele blok zal in dat geval rood kleuren. LCN LOGO is een dubbele interpreter programmeer- en vooral leeromgeving. SHIFT F10 evalueert en runt het volledige programma van boven tot beneden. Je kan alles (nogmaals) bekijken op het **grafisch** scherm.

De "objecten-om-mee-te-denken" vormen de 8 schildpadden. In deze versie is het nog niet mogelijk om meerdere objecten tegelijkertijd aan te spreken of om zelfs nieuwe (klassen van) objecten te maken; misschien in een latere versie. Voordeel dan weer is de mogelijkheid om **drie-dimensionaal** te tekenen, wat weer een goede voorbereiding vormt op het ruimtelijk denken en het latere technisch tekenen. Andere objecten die standaard kunnen aangesproken worden zijn: de toetsen, de cursor, de tursor, het gs of grafisch scherm en het ts of tekstscherm. Aan ieder van die objecten kunnen opdrachten gegeven worden.

Ik besluit met enkele punten uit beide LOGO's naast elkaar te zetten. In het achterhoofd onthouden we dat bij elke LOGO een doelgroep hoort. Hier mogen we stellen : het basisonderwijs tegenover het voortgezet onderwijs.

MSX-LOGO

- eenvoudig in aanleren;
- eigen animaties mogelijk;
- gebruik in lagere school en bijzonder lager onderwijs;
- kindvriendelijk karakter;ééntoets-programma's mogelijk;

LCN-LOGO

- nadruk op programmeren;
- veel geswitch tussen funktietoetsen, kleine en hoofd letters;
- erg leesbaar, ook door de visueel gekleurde structuren;
- sterke editor;
- drie-dimensionaal tekenen.

Datacommunicatie deel 1

Datacommunicatie. Iedereen heeft er de mond van vol. Wat is het, wat kunnen we ermee en kan het op de MSX? Allemaal vragen die na het lezen van deze bespreking niet open zullen blijven. Bij deze nemen we dan ook twee producten van Micro Technology onder de loep: MT-Telcom II en MT-MiniHost.

MT-Telcom II zit in een ietwat groot uitgevallen cartridge, vergelijkbaar met de Musicmodule van Philips, alleen is deze communicatiemodule iets dikker.

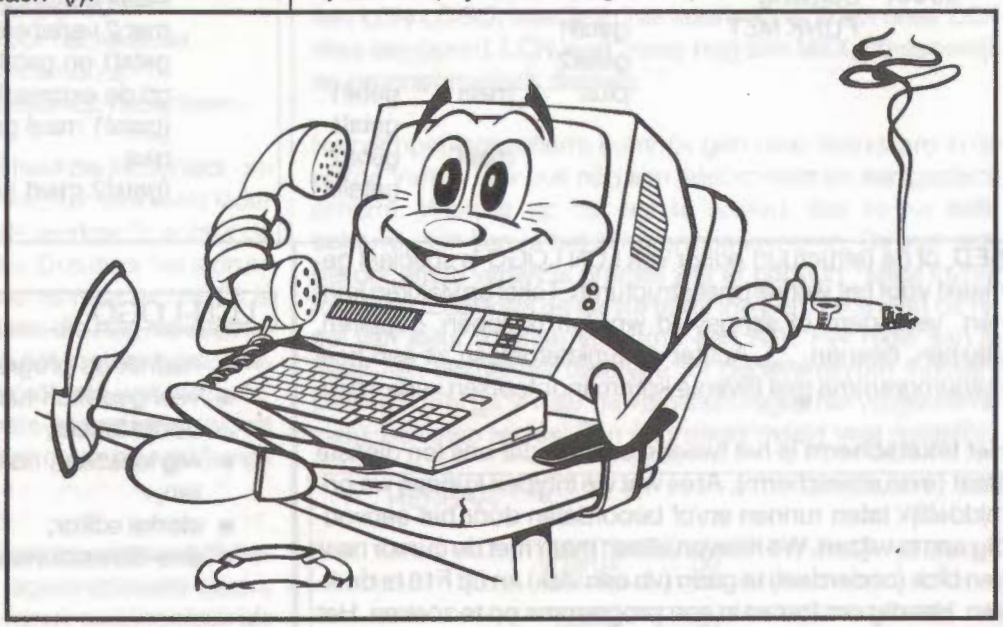
Het programma start vanzelf op. Er hoeft geen software te worden ingeladen want alle benodigde programmatuur zit al in het pakket zelf opgeslagen. De cartridge is voorzien van een snoer met een standaard telefoonstekker, die in de telefoonkontaktdoos kan worden gestoken. Na het aanzetten van de computer krijgt u van MT-Telcom een welkomsbeeld te zien. We bevinden ons nu in de "viditel-mode". Dit wil zeggen er gewerkt wordt met een snelheid van 1200/75 baud. Baud(rate) is de snelheid waarmee tekens verzonden en ontvangen worden. De module wacht op een commando, dat we vooraf laten gaan door de "slash" (/).

Als we /= intoetsen wordt er naar een telefoonnummer gevraagd. Even kijken.. laten we eens naar Tele-Wim bellen. Na het opgeven van het nummer (016200845) drukken we op de RETURN-toets. Het modem begint vrolijk te tikken en na een tijdje verschijnt het Tele-Wim welkomsbeeld: de verbinding is tot stand gebracht! We tikken de gastcodes in (2222 2222) en kijken even rond in de databank. Even maar, want de telefoonrekening loopt vlug op! Via pagina *90# verlaten we Tele-Wim weer. Dit kunnen we overigens ook doen met /I, maar dit is een wel erg onvriendelijke manier om het bestand te verlaten. Tele-Wim kan er zelfs in de war van raken en niet weten dat de beller het bestand verlaten heeft. Met de nodige gevolgen vandien!

Nu is dit bijlange niet alles wat Telcom u te bieden heeft. We kunnen pagina's uit de databank halen en ze opslagen in een van de 34 buffers. Dit aantal hangt overigens af van de beschikbare geheugenruimte. Natuurlijk is het ook mogelijk buffers weer uit te lezen en de inhoud op het scherm te plaatsen. Ook het opladen of wegschrijven naar disk of cassette is geen probleem. We kunnen zelf plaatjes maken en deze dan naar een zogenaamde prikbordpagina in de databank versturen. Men kan pagina's adrukken, de functietoetsen instellen en een andere karakterset kiezen.

Tenslotte kunnen we overschakelen naar de zogenaamde "terminal-mode". In deze mode kunnen we andere baudrates gaan gebruiken. Ideaal om bijvoorbeeld via de ingebouwde RS-232 interface randapparaten aan te sturen. Ook de Bulletin Boards (databanken met alleen tekst) kunnen we hiermee bereiken (meestal 300/300 baud).

Een volledige uitleg van alle commando's ligt buiten de bedoeling van deze bespreking. De handleiding die bij het MT-Telcom modem bijgeleverd wordt is duidelijk geschreven, zowel in het Nederlands en in het Frans. Ze telt ruim 90 bladzijden. De prijs voor het gehele pakket: rond de 9.990 Bfr.



Hierbij komen telkens dan nog de telefoonkosten en het abonnement van de databank die men opbelt. Bij Tele-Wim en MSX teleclub kan men gratis lid worden. De kosten van de telefoonverbinding kan je nakijken in de telefoongids. Bellen na 18.30 en vanuit dezelfde zone is natuurlijk het goedkoopst.

Voor wie zelf een databank wil beginnen is er MT-MiniHost. De module is volledig gelijk aan MT-Telcom, met uitzondering van een ROM waarin de software zit. De prijs van dit pakket (met een mooi verzorgde handleiding in een mapje): 33.990 Bfr. Niet voor iedereen weggelegd dus! Overigens heb je een MSX2 computer nodig, anders werkt MiniHost niet! Maar laten we eens even kijken wat voor mogelijkheden dit pakket biedt.

Bij het opstarten komen we in dezelfde mode terecht als bij MT-Telcom. Er is nu echter een nieuw commando: /q. Hiermee kunnen we het MiniHost programma opstarten.
Afhankelijk van het aantal disk-drives en de capaciteit ervan kunnen we tussen de 280 en 680 bladzijden opslaan. De databank kan maximaal 127 leden aan, met elk een klantnummer (6 tekens), een gebruikerscode (4 tekens) en een prive-codenummer (4 tekens).

Leden kunnen (indien gewenst) een eigen pagina verzorgen in de base. Indien ze een groter aantal bladzijden wensen kan men ze als "informatie leverancier" in het bestand opnemen. Ze kunnen dan pagina's aanmaken, wissen, copiëren, etc.

Elke pagina die men aanmaakt heeft een "schaduwbeeld". Hierop staat naar welke pagina het programma moet springen indien de gebruiker op 0,1,2,3,4,5,6,7,8,9,# of ? drukt. Het hekje gebruikt men meestal om naar het vervolg van een bladzijde te springen, het vraagteken om een help-pagina te zien. Op dit schaduwbeeld staat ook om welk paginatype het gaat: een antwoordpagina, een geforceerde pagina (voor telesoftware), een informatiepagina, een prikbordpagina, enz.

Vanzelfsprekend is er ook een berichtendienst: leden kunnen berichten naar elkaar sturen, men kan software of boeken bestellen of met de systeem operator (sysop) een praatje maken!

Eigenlijk zijn er veel te veel mogelijkheden om op te noemen. De 100 bladzijden tellende handleiding bezorgt je soms wel eens grijze haren, omdat er vele fouten in staan. En eigenlijk moet je al heel wat ervaring hebben met databanken eer je kan inzien wat precies de bedoeling van bepaalde pagina's is.

De diskette die bij het pakket zit, bevat een standaard start-bestand. Van hier uit kan de sysop zijn eigen bestand opbouwen.

MiniHost is een programma om te "ontdekken". Hoe meer je er mee bezig bent, des te meer dingen je ontdekt. Zo is het mogelijk van "dynamische beelden" (beelden die bewegen) te maken, iets wat normaal alleen duurdere pakketten aankunnen.

Helaas is het MiniHost pakket niet geheel foutloos. Allereerst zitten in het systeem enkele bugs (fouten) die fataal kunnen zijn voor uw databank. Pagina's die ongewenst worden gewist, paginaverwijzingen die verdwijnen, het systeem dat blijft hangen, de tel kwijt geraken bij het aantal in te lezen bladzijden, enz... Desalniettemin is het wonderlijk wat er allemaal met de MSX computer mogelijk is. We kijken alvast uit naar een verbeterde versie!

Er is ook een soort (Mini)Host verschenen op de PC: MT-TEL. Dit pakket is echter driemaal zo krachtig, maar dat uit zich dan ook in de prijs (50.000 Bfr.)!

Ik hoop hiermee de deur van de databankwereld voor een aantal lezers wat verder open gezet te hebben. Een nadeel is echter dat je er gemakkelijk aan verslaafd kan worden.

Volgende keer bespreken we enkele van de honderden databanken en BBS'sen die in de Benelux te bereiken zijn. Natuurlijk zijn reacties en/of opmerkingen welkom. Stuur ze naar het volgende adres:

Wim Dewijngaert
Van Monsstraat 14
3000 Leuven
Belgie

U kan ook een berichtje achterlaten bij Tele-Wim (016/20.08.45, klantnr.000001) of MSX tele-club (014/54.98.21, klantnr.872009).

Wim Dewijngaert

Datacommunicatie deel 2

Datacommunicatie. Iedereen heeft er de mond van vol. Wat is het, wat kunnen we ermee en kan het op de MSX? Allemaal vragen die na het lezen van deze artikelenreeks niet open zullen blijven.

In deze tweede aflevering nemen we u, zoals beloofd, mee naar de wereld van de databanken.

We beginnen met een Belgische base: Videotex. Deze databank wordt uitgebaat door de RTT, en treedt op als tussenschakel van honderden databanken uit binnen- en buitenland. Videotex heeft een Nederlandse, een Franse en een Duitse service. De telefoonnummers voor deze diensten zijn respectievelijk **1210, 1310** en **1410** (zonder zonenummer!)

Men moet eerst codes aanvragen bij de RTT, gastennummers zijn er niet. Daarna kan men in de base zelf zoeken op dienstnamen, dienstaanbieders en onderwerpen. Van elke databank is er een fiche met daarop een beknopte uitleg over de base zelf, de kosten en of het een openbare of besloten dienst is. In een mum van tijd kan men dan in verbinding treden met de gekozen databank.

En toch vinden we bijna uitsluitend bedrijven terug in de ledenlijst, gewone burgers komen bijna niet in voor. Waarom? Dan moet u maar eens naar de kosten informeren bij uw plaatselijke RTT dienst. Het enige positieve dat aan Videotex zit is dat je altijd een PERFECTE lijn hebt. Geen enkel tekentje komt verkeerd over. Zonde dat de diensten niet via een gewoon telefoonnummer op te roepen zijn. Dat zou veel goedkoper zijn, en een stuk handiger.

PTT telecommunicatie, de Nederlandse versie van de RTT, heeft ook een soort van Videotex, TELEPAD genaamd (**06-8310**). Gratis te bereiken, maar er zijn op dit ogenblik slechts twee diensten aangesloten.

Een grotere dienst van de PTT is Viditel (**06-910**). Ook hier moet je eerst codes aanvragen. Na een eerste kijkje in de base werd al duidelijk dat dit helemaal iets anders was dan onze Videotex. Want wat de RTT in honderden databanken heeft zitten, heeft de PTT in één weten te stoppen.

Een hele brok informatie dus. Je kan het zo gek niet bedenken of het zit er in. Inderdaad, ook de bekende verhaaltjes-voor-het-slapengaan. Maar als Viditel je slaapmutsje wordt, kan er de volgende maand wel een factuurtje in je bus glijden met een som op waar je "U" moet tegen zeggen. Het is denk ik duidelijk genoeg: Viditel is eveneens afdokken geblazen. Elke pagina heeft rechtsboven een "prijsje" meegekregen. Voor sommige pagina's is dit 0 cent maar voor het grootste deel staat daar dus een ander bedrag(je).

Dan nemen we onze toevlucht maar tot ComNet. En wie daar nog nooit van hoorde moest zich schamen. Want er is nog nooit zoveel reklame gemaakt voor een databank als voor ComNet. En terecht! Gastencodes zijn aanwezig, en we kunnen GRATIS (U leest het goed!) lid worden. Een paar dagen later krijgen we persoonlijk onze codes thuisgestuurd. Wat een service! Maar wat is er nu te zien in ComNet? Wel, eigenlijk UW informatie. Want ComNet op zichzelf is LEEG. Dit wil zeggen dat 95% van deze databank bestaat uit informatie-leveranciers en reklamebeelden.

Een groot aanbod (capaciteit: 100.000 pagina's), onder andere veel informatie over computers, maar daar blijft het dan ook bij. ComNet draait op niets anders dan het geld van de informatie leveranciers. Over de service hebben we overigens niet te klagen... de postbus staat altijd open voor opmerkingen en de IL krijgt zelfs een keurige handleiding thuisgestuurd (zo hoort het ook!) Sinds einde juni is er een rechtstreekse lijn aangelegd tussen Belgie en Nederland, zodat de Belgen ook eens (voor de prijs van een gesprek naar Brussel) naar Nederland's grootste base kunnen kijken. Telefoonnummer: **02/252.40.45** (voorlopig 6 lijnen). Men telde eind juli 1988 ruim 12.500 leden!

ComNet heeft ook nog twee andere "broertjes": de "ComNet chat-lijn" en de "ComNet actua-lijn". Beiden zijn niet te bereiken vanuit Belgie, omdat ze op een 06-nummer zitten (50 cent per minuut).

Tot zover de grote, professionele bases. We stappen over naar de hobbybanken. We beginnen met MSX tele-club (**014/54.98.21**, 24/24 uur). Deze dienst is overigens helemaal gratis, je moet wel lid van de MSX club zijn om in het volledige bestand te kunnen rondkijken. Zeshonderdtachtig bladzijden staan tot uw beschikking: veel tele-software en tele-shopping, een paar prikborden, IL'ers en een postbus. Daarmee is tele-club uitgeteld. En dat is maar normaal ook, want het is gewoon de bedoeling om wat aanvullende informatie bij het clubblad te geven. Het Nederlands computerblad "TRON" deed hetzelfde en kwam met "TRON VIEW-

TEXT EXTENDED" (**079/31.01.66**, 24/24 uur), in principe volledig gelijk aan tele-club. Toegangscode's zijn niet nodig, je logt gewoon in met je naam.

Data Tel Belgium (10:00 tot 17:00 en 23:00 tot 07:00 uur) biedt u niet veel meer dan de doorsnee-hobbybase: prikborden, IL'ers, tele-software, een chatpagina en nog wat afzonderlijke rubriekjes. We troffen er veel "afgekeken" prentjes in. Code's zijn gratis aan te vragen.

Tele-Wim (**016/20.08.45**) sluit het rijtje van de Belgische hobbydatabanken. Nu moet ik even voorzichtig worden en op mijn woorden letten: Tele-Wim is immers mijn eigen databank. Wat wij u aan te bieden hebben? Gratis toegangscodes, reklamepagina's, IL'ers, sport (100 biz.!), een databankenrubriek (100 biz.!), de kontaktrubriek, vrije tijd, een computercorner (MSX, Sega, Nintendo, PC, Spectrum, Amstrad, Commodore, Atari ST), drie verschillende types prikborden, tele-software, tele-shopping, chat-line, speeltips... Kortom: te veel om op te noemen. Kom maar eens een kijkje nemen: we zijn elke dag open vanaf 07:00 tot 23:00 uur (capaciteit: voorlopig 700 bladzijden, uitbreiding wordt verwacht).

En dit waren ALLE Belgische hobbybanken. Een aardig aanbod, maar als ik u over de Nederlandse situatie vertel, zal u merken dat de Belgen er maar een beetje triest bij zitten. Datacommunicatie is daar een rage geworden, net als de CB-ers een tijdje geleden. De Hollanders hebben namelijk meer dan honderd hobbydatabanken (waarvan 90% met gratis codes) op hun actief staan!!! Ja, daar mag ik best drie uitroeptekens voor gebruiken.

We wippen even over de grens en brengen een bezoekje aan MiamiTel (mijn favoriet, tel.nr. **023/35.59.49**, 24/24 uur)! Beslist geen base die opgezet is door een beginner: de systeem operator heeft immers al vijf jaar ervaring. Zijn base heeft iets aparts, iets wat je nergens anders terug vindt: prachtige grafics. Qua inhoud is er niet veel verschil met een standaard base, maar de grafische omkadering en de vele truukjes maken het tot een prachtige databank (overigens met gratis codes). Sinds kort is MiamiTel echter voor een tijdje gestopt. We laten u via deze rubriek weten wanneer hij weer on-line gaat.

Volgende keer bespreken we een paar andere Nederlandse bases.

Bezint eer ge begint...

Voor alle lezers die na het lezen van dit artikel zin hebben gekregen zelf een databank op te starten, even een waarschuwing.

Het is best leuk om een eigen base te beginnen, maar dan moet je wel zoveel mogelijk (liefst elke dag) nieuwe informatie in je bank zetten. En dat kost heel wat energie en inspanning, neem dat maar van mij aan!

Buiten dat het een tijdrovende hobby is, is het ook een erg dure hobby. Je hebt nodig: een MSX2 computer (liefst met 2 dubbelzijdige disk-drives), een (kleuren) monitor, het MiniHost pakket, een extra telefoonlijn en desgewenst een printer.

Daarbij komt dan nog de maandelijkse telefoonrekening, die hoog kan oplopen. Want je moet natuurlijk op de een of andere manier reklame maken in andere databanken. Het is voor een sysop (=systeem operator) immers bijzonder waardevol als hij veel inloggers krijgt, want dat is de "beloning" voor zijn werk.

Hoe zit het nu met Tele-Wim? De databank loopt sinds 1 juli 1988 en momenteel hebben we meer dan zeventig leden. In de maanden juli-augustus hebben we 115 telefoontjes mogen ontvangen... Maar dat is natuurlijk niet helemaal vanzelf gekomen. Buiten de vermelding in Club Magazine, hebben we ook reklame gemaakt in ComNet, MiamiTel, ZaanData en TRON Viewtext. Daar zitten veel Nederlandse databanken tussen, want als je alleen op de Belgische modembezitters rekent, kom je niet ver.

Geld uit je databank halen is bijna niet mogelijk, vermits de meeste informatie-leveranciers een faktuur wensen, en dat is iets dat je als particulier niet kan uitschrijven.

Na het lezen van deze opmerkingen zullen er dus al heel wat mensen van idee veranderd zijn. Desondanks wil ik iedereen die zin heeft ermee te beginnen, ook aanraden van het te doen. Het is een prettige hobby, die niet vlug zal vervelen!

TELE-Wim Dewijngaert

Zet dit programma op je diskettes onder de benaming 'AUTOEXEC.BAS' en het is zover. Je moet alleen nog met de cursor je gellefkoosd programma aanduiden en het loopt. Een afstandsbediening is er nog niet bij.

Een beetje uitleg nu.

In regel 80 kies je voor breedte 40 : zo kunnen er alvast 3 kolommen naast elkaar. Maar je zet ook het scherm uit want het kan er wat rommelig aan toe gaan. Als regel 100 merkt dat niet alle files op het scherm kunnen dan wordt alles weer gewist, schakel je breedte 80 in en wordt alles overgedaan.

Om de rest van het programma te kunnen begrijpen, moet je alleen weten dat de scherminhoud van SCREEN 0 in de video ram wordt opgeslagen vanaf adres 0.
Zo geeft bij voorbeeld PRINT CHR$(VPEEK(0)) je als antwoord het karakter dat linksboven op je scherm staat. In breedte 40 is VPEEK(40) het begin van de tweede regel, in breedte 80 wordt dat VPEEK(80) enz.

In regel 150 wordt eerst het karakter onthouden waarmee de derde schermregel begint, want vanaf programmaregel 180 gaat daar een nagemaakte cursor knipperen (om beurten plaats je een blokje en het oorspronkelijke karakter : de filenaam mag niet worden verminkt of hij is straks niet te lezen).

Vanaf regel 240 kan je met de pijltjes werken, maar je aktieterrein wordt via 'plaats' en 'regel' beperkt tot uitsluitend die plaatsen waar inderdaad files staan. Naar onder toe gebeurt dat door eerst te kijken of daar wel iets voorhanden is (regel 260 : if ... ASC(" ")).
Op de regels 300, 360, 420 en 480 is je pijlkeuze gemaakt, wordt de filenaam hersteld en begint ergens anders wat te knipperen. De regels 580 en volgende (na 'return') lezen dan de 12 tekens vanf de plaats die jij aanduidde en zo is de gekozen filenaam bekend. Dus : 650 run "...".

Voor de fijnproevers misschien nog dit : wil je maar bepaalde files op je scherm, dan kan de selektie gebeuren in de regels 90 en 100. Dat wordt dan bij voorbeeld files "*.bas" voor alles wat de extentie "bas" draagt enz.

Eigenlijk heb ik het programma zelf gemaakt om het te gebruiken in een ander programma waarin geregeld bestanden moesten ingeladen worden. Zoiets kan dus ook !

Veel plezier ermee ! J. Simal

FILES BAS D2

```
Kies bestand met pijlen en druk return.

HELI1    .BIN YESTERDA.Y    GALF     .BAS
DARTS    .BAS SCHUIF  .BAS  PACMAN   .BAS
REISBAG  .BAS SLANGEN .BAS  DIEFSTAL.BAS
TWEESTEM.BAS MSXDOS   .SYS  COMMAND  .COM
#A-015ML.BAS TESTDEM2.BAS   CLS-ROTR.BAS
BELGIE   .BAS #B-00B  .BAS  KAR-TEST.BAS
#B-00A   .BAS #B-00C  .BAS  BALSPEL  .BAS
FILES    .BAS TRANS         GALF     .D2
GALF     .D1  Q             TEMP
HELIDATA.BAS DRAWING  .BAS  LIST1    .BAS
LIST4    .BAS LIST5   .BAS  LIST6    .BAS
LIST7    .BAS LIST8   .BAS  LIST9    .BAS
LIST2    .BAS LIST3   .BAS  SFEDIT   .BAS
```

MUSICMOD BAS D2

We hebben de laatste tijd veel brieven gekregen (onder andere van Henk Van Wulpen) van mensen die zelf aan het experimenteren wilden slaan met de Musicmodule. Hierbij publiceren we een testprogramma van Philips voor deze cartridge. Start de computer op met de ESC toets ingedrukt en run dan onderstaand programma. Bij een goede studie van de listing zou het mogelijk moeten zijn je eigen routines in BASIC te schrijven. Natuurlijk zijn we erg benieuwd naar de resultaten!

MSX-PAKKET UNIVERSELE INPUT/OUTPUT KAART MSX1/2

Om tegemoet te komen aan de vraag van vele zend-en luister amateurs in Belgie en Nederland, en wegens het ontbreken van degelijke hard- en software op dit gebied, wordt er hier een universeel pakket besproken dat meerdere dingen in zijn mars heeft dan alleen ontvangen van TELEX en ASCII uitzendingen. De MSX en zijn standaard, leent zich bijzonder goed in dit gebied, wegens zijn grote storings-onderdrukking op radio en ontvangst-apparatuur, hetgeen meestal een van de grootste problemen is bij dit soort aktiviteiten.

Dit pakket bevat:

- 1) een cartridge met schakeling
- 2) een tape of disk met de volgende programma's:
  - een BASIC RTTY/ASCII program.
  - een MC-code RTTY.BIN program.
  - een hulpprogramma
- 3) handleiding
- 4) listings

Algemene specificaties cartridge:

- input signaal 5 V-TTL tot +12 V
- output 5 Volt TTL
- aantal bits: 5 tot 8 programmeerbaar
- aantal stopbits: 1, 1.5 of 2 stopbit
- parity of no-parity bit
- speed: 45.45 bit/sec tot 9600 bit/sec (vrij programmeerbaar)

Toepassingen

- serieel naar parallell omzetting & omgekeerd
- aansluiting seriele printer,plotter,modem,enz.
- RTTY of TELEX interface (BAUDOT)
- ASCII interface (mailbox,basicode)
- pseudo-RS232 interface (mits signaal aanpassing)
- koppeling tussen 2 MSX-computers
- koppeling tussen MSX- en vreemde computers (PC)

Installatie

1) Zet de computer uit en steek de cartridge in een willekeurig slot.
2) Maak de verbinding tussen cartridge en modem met een stereo-snoer, aan de modem met een DIN-stekker en aan de cartridge met een mini-stereo jack (TANDY) (Aansluitingen : zie technische gegevens)
3) Zet eerst de modem aan, daarna monitor en computer
4) Laad programma RTTY van tape of disk (bij tape daarna ook MCRTTY laden met BLOAD"CAS:")
5) RUN het programma voor een eerste kennismaking

Werking en kontrole

Op de cartridge bevinden zich twee LED's, rood en groen. Bij elke verandering van het binnenkomende signaal, afkomstig van de modem, zal de groene LED dit signaal volgen. Omgekeerd zal elke signaal verandering afkomstig uit de computer de rode LED laten knipperen. Zo kunt U gemakkelijk zien of er DATA-transport plaatsvindt of niet. Dit (serie) signaal wordt in de cartridge automatisch omgezet naar een parallell-signaal, en naargelang de instelling, daarna aangenomen door de computer en verder verwerkt.

Het programma RTTY/ASCII

Dit programma is speciaal bedoeld voor de zendamateurs en heeft als zodanig alle mogelijkheden van een commercieel programma. Het kan zenden en ontvangen in BAUDOT of ASCII, en dit in een vrij te kiezen snelheid van 45.45 tot 4800 baud (bits/sec). Bovendien kan automatisch alle data opgeslagen worden in een buffer en later desgewenst afgedrukt worden op een printer. Het programma bevat ook extra's zoals voorgeprogrammeerde teksten of boodschappen en een mogelijkheid tot het automatisch inloggen van een gewerkt station. Het gebruik ervan is zeer eenvoudig en spreekt voor zichzelf na het geRUNd te hebben.
Na het RUNnen wordt automatisch het benodigde machinetaalgedeelte ingeladen, en getest of dit aanwezig is.
Bij het kiezen van BAUDOT RX of TX wordt automatisch gekozen voor een snelheid van 45.45 baud, tenzij U vooraf gekozen hebt voor SET SPEED, waarin een aantal veel gebruikte snelheden zijn aangegeven. In elke gekozen RX of TX mode, is onderaan een status-lijn zichtbaar waar alle gemaakte instellingen zichtbaar blijven, totdat U met ESC terugkeerd naar de CONTROL-MODE. Tijdens een TX mode, kan U met de SELECT toets een voorgeprogrammeerde zin selecteren; de computer vraagt dan welke van de 9 teksten

U wilt hebben (0 tot 8); en deze worden dan automatisch uitgezonden. Bij de keuze LOG CALL in CONTROL MODE wordt U gevraagd naar de CALL van het tegenstation. Deze wordt dan automatisch opgenomen in de voorgeprogrammeerde teksten voor later gebruik. Kiest U voor Print Buffer, dan kunnen alle teksten die op het scherm te zien waren nog eens rustig nagelezen worden, en- of eventueel afgedrukt worden op een printer.Bij MSX-2 is het mogelijk om het hele video-geheugen te gebruiken als tekstgeheugen (128K !), en deze tekst te saven op disk. Zo is het b.v. mogelijk, bij een snelheid van 45 Baud, om meer dan 2 uur aan tekst op te slaan, die u nadien kan bekijken. Bij de MSX-1 versie zijn de programma's iets minder uitgebreid. Dit komt natuurlijk door het kleinere video-geheugen, en de beperking van 40 karakters per regel.

Bespreking van het schema

De werking van de schakeling is gebaseerd op het gebruik van een UART als serieel-parallel omzetter. De UART krijgt via een buffer-transistor zijn serie-signaal toegevoerd, en zet dit automatisch om in een parallel-signaal. Dit signaal komt daarna op de data-bus van de computer terecht en wordt verder verwerkt door het programma, waarna het als letters of cijfers op het scherm komt. Omgekeerd gebeurt hetzelfde; wanneer een letter of cijfer op de data-bus staat, zal dit op de gepaste tijd aangenomen en verzonden worden door de UART, en komt dan via de buffer-transistor als serie-signaal aan de uitgang. Dit alles gebeurt volgens een strikt te volgen protocol : de ASCII of BAUDOT mode.

Bij ASCII worden meestal 7 of 8 data-bits gebruikt, met 1 stopbit en bij BAUDOT meestal 5 bits met 1 of 2 stopbits.
De parity-bit wordt slechts zelden gebruikt. De UART krijgt een kloksignaal aangeboden van een programmeerbare BAUD-generator met een snelheid van 16 maal de BAUD-snelheid. Zo kan de snelheid van het serie-signaal liggen tussen 45 en 4800 bits/sec , met elke tussenliggende waarde. De snelheid wordt ingesteld door op de data-bus een getal te zetten, wat het deeltal instelt van de opgewekte frequentie van de kristal-oscillator met delers HC4060 en HCF40103. Het deeltal wordt doorgegeven via het buffer-ic HCT74374, en mag liggen tussen 1 en 210 voor resp. 4800 en 45.45 BAUD.
Onderaan het schema bepaalt het ic HCT74688 , via zijn ingesteld adres over welke poortnummers men de schakeling wil laten werken (0 tot 255).
Standaard wordt poort 0 van de Z80 gebruikt, vermits reeds vele poortnummers van de MSX-2 in gebruik zijn.
Rechtsonder bepalen de poorten HC7432 en het selectie-ic HCT74139, of er gelezen, geschreven, snelheid instellen, of de UART ingesteld moet worden.

Een overzicht:

poort 0 RD = lees data binnen
poort 0 WR = zend data buiten
poort 1 RD = lees status UART
poort 1 WR = zet UART in bep. mode
poort 2 WR = stel snelheid in

UART modes :

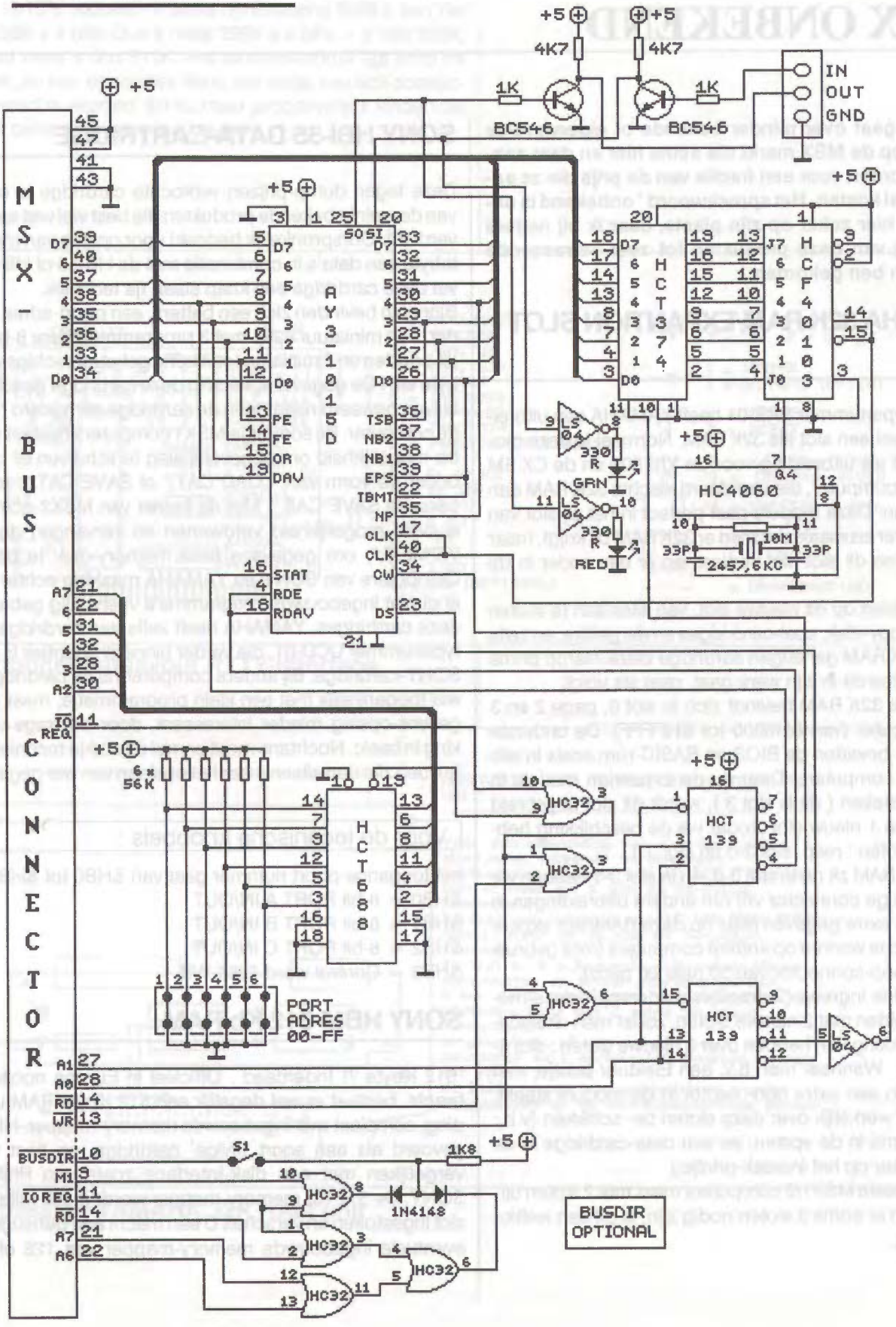
data &H19 = 5 bits 1 1/2 stp-bit Baudot
data &H17 = 8 bits 1 stop-bit no parity ASCII
data &H1F = 8 bits 2 stop-bit no parity ASCII
data &H15 = 7 bits 1 stop-bit no parity ASCII
data &H1D = 7 bits 2 stop-bit no parity ASCII

De schakeling in optie onderaan, is bedoeld voor MSX-computers die het BUSDIR-signaal nodig hebben, en is vooral van belang voor MSX2 machines zoals de SONY HB-F700P en PHILIPS 8250, 8255 en 8280.
De VG-8235 werkt nog zonder dit signaal, evenals de meeste MSX1 computers, zodat dit stuk schakeling mag weggelaten worden. Met enige kleine wijzigingen is het mogelijk om de cartridge te gebruiken als een soort RS-232 serie-interface of als modem met een toon-modulator.
Hiertoe moeten de ingang en uitgang geschikt gemaakt worden voor signalen van -12 en +12 volt. De collector-weerstanden van de in en uitgangs-transistoren moeten daarvoor aan +12V gelegd worden. Op deze manier is het mogelijk 2 verschillende computers te koppelen (vb.: MSX1/2 en PC of andere computer met seriele ingang).

Geïnteresseerden kunnen voor het bekomen van dit pakket met de nodige programma's kontakt opnemen met:

DE DAUW HERMAN
WALLENHOF 93
B-2770 NIEUWKERKEN-WAAS
Tel. 03/7770676

MSX RTTY INTERFACE



MSX ONBEKEND

Dit verhaal gaat over minder bekende of eigenaardige produkten op de MSX markt die soms hier en daar aangeboden worden voor een fractie van de prijs die ze anders normaal kosten. Het spreekwoord ' onbekend is onbemind ' is hier zeker op zijn plaats, daar ik bij nadere bestudering van deze produkten tot zeer verrassende bevindingen ben gekomen...

1) YAMAHA 32K RAM/EXPANTION SLOT UNIT

Onder het typenummer SRE-01 heeft YAMAHA een uitbreiding van zowel een slot als 32k RAM. Normaal is deze module bedoeld als uitbreiding voor de YIS 503 en de CX 5M MSX-muziekcomputer, die standaard slechts 32K RAM aan boord hebben. Deze module past perfect in het zij-slot van deze computer en maakt dat men er 32K RAM bij krijgt, maar vooral dat men dit slot niet verliest, en er een ander in de plaats kijgt.
Het is toegelaten op dit nieuwe slot, van alles aan te sluiten zoals een floppy-disk, spel-cardridges en dergelijke, en zelfs een extra 64K RAM geheugen cardridge bleek hierop prima te werken ! Hoe dit in zijn werk gaat, gaat als volgt.
De inwendige 32K RAM bevindt zich in slot 0, page 2 en 3 van de computer (van &H8000 tot &HFFFF). De onderste pages 0 en 1 bevatten de BIOS en BASIC-rom zoals in alle andere MSX computers. Door nu de expantion module in het zijslot te steken (dit is slot 3), wordt dit slot uitgebreid met tenminste 1 nieuw slot ,zodat we de beschikking hebben over 2 sloten : resp. slot 3-0 en slot 3-1.
De extra 32K RAM zit nu in slot 3-0, en in slot 3-1 hebben we nu een volledige connector vrij om andere uitbreidingen in te steken. Als extra gegeven blijkt nu deze RAM/slot expander even goed te werken op andere computers (mits gebruik van een verloop-connector van 50 naar 60 pens).
Door een kleine ingreep (2 draadjes solderen) is deze module uit te breiden met 2 nieuwe sloten, zodat men uiteindelijk de beschikking kan hebben over 3 nieuwe sloten : slot 3-1, 3-2 en 3-3 . Wanneer men b.v. een Elektuur-printje, met een Eprom en een extra con- nector in de module steekt, kan men ook werkelijk over deze sloten be- schikken (v.b.: een programma in de eprom, en een data-cardridge in de extra connector op het insteek-printje.)
Omdat de meeste MSX1/2 computers maar met 2 sloten uitgerust zijn, en er soms 3 sloten nodig zijn, is dit een welkome oplossing.

SONY HBI-55 DATA-CARTRIDGE

Deze tegen dump-prijzen verkochte cardridge is ook een van de weinig bekende produkten die best wel wat aandacht verdient. Oorspronkelijk bedoeld voor opslag van max. 4 kilobyte aan data's in combinatie met de HB-55 of HB-75, bevat deze cardridge een knap staal- tje techniek.
Binnenin bevinden zich een batterij, een poort-adres de- coder, een miniatuur 8255 met 3 programmeerbare 8-bits in of uit poorten en 2 miniatuur statische geheugen-chips van 2K-byte elk ! De gegevens, eens in deze cardridge geschreven, blijven bewaard nadat men de cardridge verwijderd heeft uit de computer. Bij sommige MSX1 computers bestaat een extra mogelijkheid om gegevens weg te schrijven of te lezen onder de vorm van: LOAD"CAT:" of SAVE"CAT:", zoals bij cassette SAVE"CAS:". Met de komst van MSX2-computers is deze mogelijkheid verdwenen en vervangen door SAVE"MEM:", om gegevens naar memory-disk te brengen. Computers van SONY en YAMAHA maakten echter in hun al of niet ingebouwde programma's veelvuldig gebruik van deze cardridges. YAMAHA heeft zelfs een cardridge onder typenummer UCD-01, die verder binnenin identiek is aan de SONY-cardridge. Bij andere computers is de cardridge best wel toegankelijk met een klein programmatje, maar als gegevens-opslag minder interessant, door de trage verwerking in basic. Nochtans moet er met dit stukje techniek meer te doen zijn dan alleen maar het opslaan van wat gegevens...

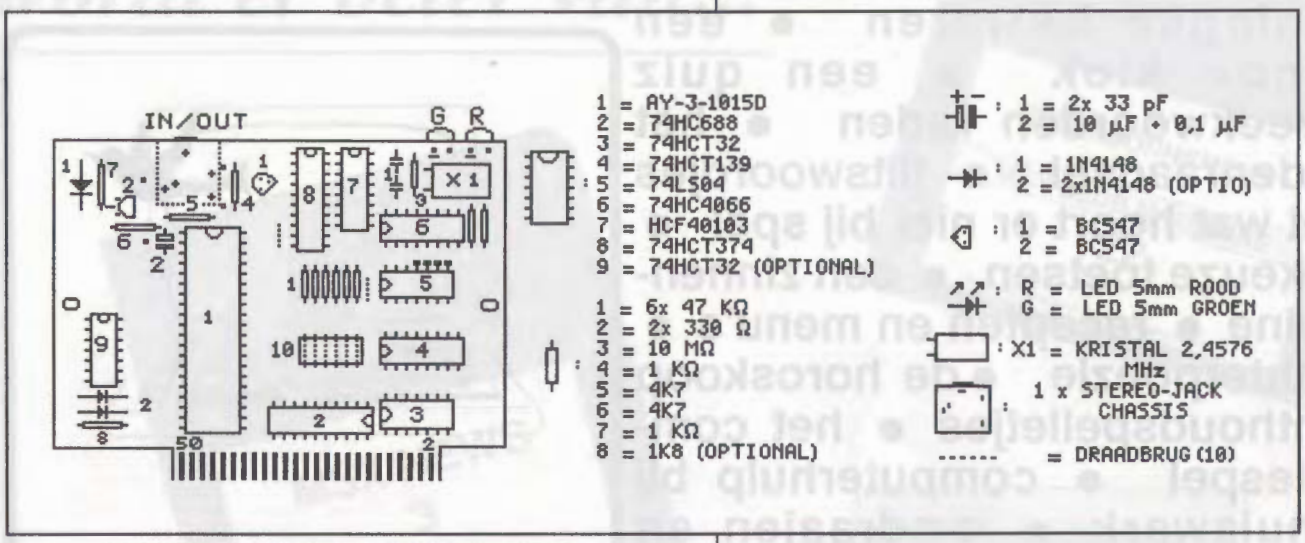
Voor de technische knobbels :

het toegangs-poort nummer gaat van &HB0 tot &HB3 :
&HB0 = 8-bit PORT A IN/OUT
&HB1 = 8-bit PORT B IN/OUT
&HB2 = 8-bit PORT C IN/OUT
&HB3 = Control word 8255 WR

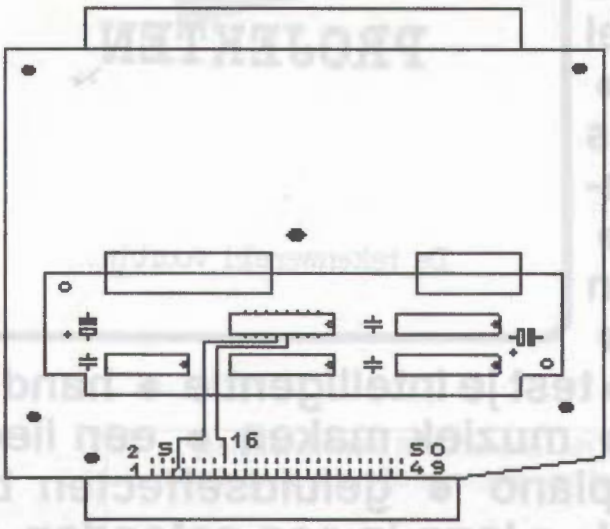
SONY HBM-512 Kb RAM

512 Kbyte ?! Inderdaad . Officieel in Europa nooit uitgebracht, bestaat er wel degelijk een 512 Kbyte RAM uitbreiding, compleet met ingebouwde memory-mapper. Hij is uitgevoerd als een soort 'lijvige' cardridge, en best wel te vergelijken met een disk-interface zoals van Philips of SONY. De 512-Kb memory-mapper wordt in elk willekeurig slot ingestoken en verschaft U een macht aan geheugen. De eventuele ingebouwde memory-mapper van 128 of 256K

bent U dan wel gelijk kwijt, omdat de MSX standaard (?) maar 1 memory-mapper tegelijk ondersteunt. De cardridge bevat 19 IC's waarvan 4 stuks dynamische RAM's van het type 256K x 4 bits. Dus 2 maal 256K x 4 bits = 8 bits 256K, dit maal twee is dus 512K. Het stroomverbruik ligt rond de 300 mA, en kan er nog net door, om langs een slot-connector gevoed te worden. En nu maar programma's vinden die met dit geheugen rekening houden....



Componenten-layout RTTY-interface



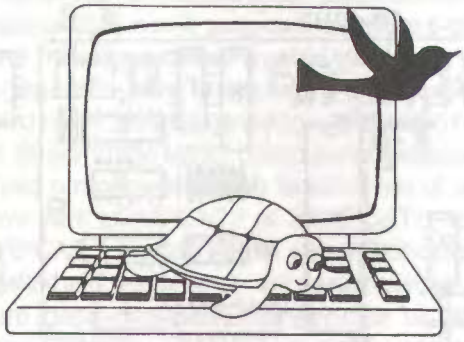
AANPASSING VAN YAMAHA 32K RAM/EXP. SLOT UNIT

Voor een uitbreiding van 2 bijkomende slots,maak volgende verbindingen:
Soldeer een draadje van IC 105 (74ALS138) /pin 12 naar pin 5 van de 50-polige connector.
Soldeer een draadje van IC 105 (74ALS138) /pin 13 naar pin 16 van de 50-polige connector.
Zo zijn er 2 bijkomende slot-select lijnen aangesloten op 2 niet gebruikte pennen.
U kan de 2 extra sloten gebruiken door een opsteek-printje met 2 extra 50-polige connectoren
te gebruiken, of een printje met 1 extra connector en 1 eprom-voet. Daartoe moeten ook op dit printje
de beide nieuwe slot-selectlijnen worden aangesloten op de corresponderende pennen.

Aanpassing YAMAHA 32K RAM unit

Een MSX LOGO WERKBOEK
met 50 boeiende projekten
om mee te stoeien
door John Peeters & Maria Randoe

schildpad besturen met joystick • schildpad op jacht • hartjes jagen • jagen met z'n tweeen • vang de dief • snel-tekenen • landkaarten maken • plattegronden tekenen • tekeningen bewaren • een lopende klok • een quiz • spreekwoorden raden • het woordenraadsel • flitswoordjes • het wat hoort er niet bij spel • meerkeuze toetsen • een zinnen-machine • recepten en menu's • computerpoëzie • de horoskoop • onthoudspelletjes • het com-binatiespel • computerhulp bij het huiswerk • omdraaien en sorteren • op bezoek bij de dok-ter • het dierenraadsel • dierenkennis • de computer als expert • galgje • memoryspel • mastermind • zeeslag • kerstballenspel • helikopterreis • Tour de France • de elfsteden-tocht • diamant in het labyrint • een paardenwedrenrace • een sommetjesmachine • hoofdrekenen met de computer • test je intelligentie • handig rekenwerk • grafieken maken • muziek maken • een lied-jesparodie • de computer als piano • geluidseffecten bij spelletjes • muziek uit het heelal • logo in een notendop • logo hulpcommando's • van engelse naar nederlandse logo



50 MSX-LOGO PROJEKTEN

De tekenwereld voorbij ....

50 MSX LOGO PROJECTEN (boek) ..... 590 fr / fl 33
50 MSX LOGO PROJECTEN (boek + schijf)..... 990 fr / fl 55
verkrijgbaar bij uw MSX dealer of bij
MSX CLUB Mottaart 20 3170 Herselt

Een abonnement op

MSX CLUB MAGAZINE :

uw MSX computer

wordt er beter van...

jaarabonnement (6 nummers, van januari tot december)	850 fr / fl 45
diskabonnement (6 nummers, van januari tot december)	2250 fr / fl 122

abonneren kan door overschrijving op onze bankrekening :
t.n.v. DAInamic V.Z.W.
voor België :
GENERALE BANK TONGERLO 230-0096323-22
voor Nederland :
AMRO BAARLE-NASSAU 46.07.36.051 (of giro 1091055)

Wim Dewijngaert

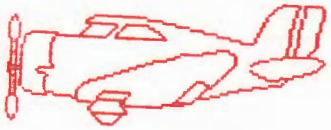
Verkoopprijs: Bfrs 295 / Hfl 14,75

HET GROTE MSX PEEKS, POKES EN TRUUKS BOEK

Deel 2

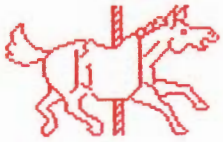
een uitgave van MSX-club BELGIË - NEDERLAND

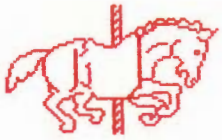
SUPERFONT



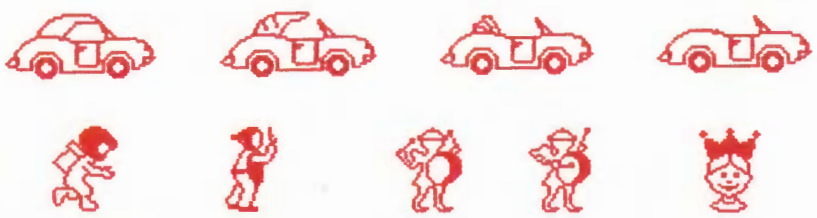
Vragen ze me toch wel om even de tekeningen van **SUPERFONT** toe te lichten ...lijkt me onmogelijk.. dus doen we hier een greep uit de **1234** beschikbare tekeningen. Persoonlijk houd ik niet van vliegtuigen, maar zo'n vliegtuig kan best bruikbaar zijn als illustratie. Deze

miereneter spreekt me meer aan , vooral omdat het afwisselen van de twee tekeningen een razendmooie animatie mogelijk maakt. Hetzelfde effect is mogelijk met deze twee lieve paardjes van





de molen - deze beestjes maken trouwens een aardige prijs op de antiekmarkt ! Kangoeroe-met-baby is ook een dankbaar onderwerp, vooral omdat baby moeiteloos in en uit de buidel kan. Wat dacht je van konijntje in 3 posities ? De kameel doet al een tijdje dienst in onze educatieve programma's, tot groot jolijt van de jonge gebruikers. Met deze cabriolet reeks zijn zeker ook leuke effecten mogelijk. Of was je op zoek naar een reeks figuren voor uw volgende graphic-adventure ?



Ik mag niet eindigen zonder de trein-fanaten te verrassen.
Het einde van deze ruimte is in zicht...

Er blijven nog **1212** tekeningen te bespreken.....

MSX WORKSHOP 88
biedt U op 3 schijven meer dan 1 MEGABYTE software :

BASIC
PASCAL
LOGO
MACHINETAAL

PLAATJES
STEMPELS & FONTS
VOOR DYNAMIC PUBLISHER

CURSUS PROGRAMMEERTECHNIEKEN
CURSUS SCREEN 1 SCROLLING

HARDWARE PROJECTEN

Onze vrienden van COCON hebben w
DYNAMIC PUBLISHER ontworpen.
U vindt deze fonts op een van de schijven van het diskabonn
Ze hebben de extensie .FNT en zijn onmiddellijk bruikbaar
Dank aan het actieve team uit Nederland

een uitgave van DAInamic VZW / MSX - CLUB
Mottaart 20 B-3170 Herselt (014) 54 59 74