MSX

truuks en tips

deel 4

Hans Klopper en Marcel Le Belle

MSX truuks en tips deel 4 Hans Klopper en Marcel Le Belle

MSX
Truuks en tips
Deel 4

truuks en tips

deel 4

Hans Klopper en Marcel Le Belle

uitgeverij STARK-TEXEL
postbus 302 - 1794 ZG Oosterend tel. 02223 - 661

CIP-GEGEVENS KONINKLIJKE BIBLIOTHEEK, DEN HAAG

MSX

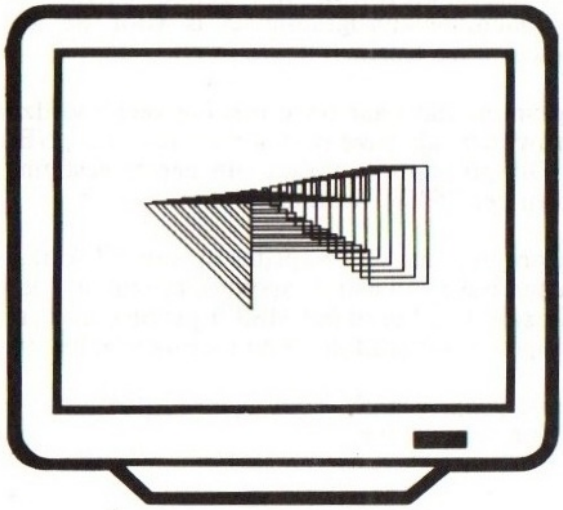
MSX truuks en tips. – Oosterend : Stark-Texel
Dl. 4 / Hans Klopper en Marcel Le Belle.
ISBN 90-6398-897 4
SISO 365.3 UDC 681.3.06
Trefw.: Programmeren (computer) MSX (computer).

1e druk 1986
ISBN 90 6398 897 4

Inhoud **pagina**

Machinetaal

Inleiding



Daar zijn we al weer met deel 4 uit de truuks en tips reeks. Deel 3 en deel 4 hebben elkaar dit keer vrij snel opgevolgd. We hopen u ook in dit deel veel plezier te doen met de vele slimmigheidjes. Zoals u reeds in het derde deel heeft kunnen zien, behandelen wij veel problemen door middel van machinetaalroutines. Dit is makkelijk, omdat de snelheid gemiddeld 50 maal hoger ligt dan die van Basic. We hebben nu bovendien een extra hoofdstukje opgenomen met enkele truuks en tips voor de machinetaalprogrammeur.
We verheugen ons in het feit dat er duidelijk steeds meer belangstelling ontstaat voor het programmeren in machinetaal. Logisch, als je ziet hoeveel méér je met je computer kunt doen. Het vergt echter een behoorlijke studie, maar dit betaalt zich dubbel en dwars terug door middel van de nieuwe mogelijkheden die u worden geboden.

We vermelden er voor alle zekerheid bij, dat u beslist geen kennis van machinetaal hoeft te hebben om de routines toe te kunnen passen. Slechts het hoofdstukje met truuks en tips voor de machinetaalprogrammeur is voor de wat gevorderden onder onze lezers.

Een programma dat naar onze mening veel hoofdzorgen zal laten verdwijnen als sneeuw voor de zon, is: „NEW, zeker weten." Dit programma vraagt om een bevestiging na het ingeven van het "NEW"-kommando.

Ook een programma als "5 sprites op een rij" kan zeer goed zijn diensten bewijzen aan de spelprogrammeur, die zich nog steeds afvraagt hoe hij/zij het MSX-1 probleem van maximaal 4 sprites op een horizontale rij zou kunnen oplossen.

Naast deze programma's kunt u natuurlijk nog veel meer leuke dingen verwachten.

Dankwoord

Onze dank gaat ook dit keer weer uit naar Electronics Nederland voor het beschikbaar stellen van twee Spectravideo X' Press MSX computers met het Wordstar tekstverwerkingssysteem.

Bovendien wil ik (Hans Klopper) mijn vriendin (Benedicte) bedanken voor al de steun die zij mij bij het schrijven van dit boekje heeft gegeven.

Tevens gaat onze dank uit naar Kees, Joke, Alfred, Peter en alle mensen die ons de nodige hulp hebben geboden.

MSX gegevens

Na het opstarten van de computer verschijnen er bovenin het beeld een aantal gegevens omtrent het aantal vrije geheugenplaatsen en de BASIC versie.

Met de routine die in onderstaand programma is verwerkt kunt u deze gegevens ten alle tijden opvragen.
Dit gebeurt door een ROM-routine aan te roepen. Deze routine is gelokaliseerd op adres &H7D29.

Als u deze routine vanuit BASIC direkt aanroept, dan genereert de computer een foutmelding. Daarom zit in onderstaande programma een "ON ERROR GOTO" verwerkt, de foutmelding blijft dan achterwege.

```
2 '**********************************
3 '*                                *
4 '*        MSX     GEGEVENS        *
5 '*                                *
7 '**********************************
8 :
10 DEFUSR=&H7D29
20 ON ERROR GOTO 30
25 X=USR(0)
30 RESUME 40
40 END
```

Listbeveiliging

Het begint zo langzamerhand een gewoonte te worden, om een zogenaamde listbeveiliging te plaatsen in de truuks en tips boekjes.
Ditmaal is het echter een beveiliging die geheel los staat van het BASIC-programma. De beveiliging is helemaal geschreven in machinetaal.
Nadat u de machinetaal beveiliging heeft geladen kunt u het BASIC-programma inladen. Het BASIC-programma is nu onmogelijk te listen. Omdat de beveiliging is geschreven in machinetaal is het moeilijker (voor de potentiële kraker) om uit te vinden hoe de beveiliging is opgebouwd.

Het machinetaalprogramma maakt gebruik van twee zogenaamde hooks. Dit zijn plaatsen bovenin het geheugen waar de ROM soms naar toe springt. Door op deze plaats een sprong te maken naar een eigen routine, is het mogelijk om de werking van de ROM enigzins te veranderen.

De hook voor de toetsenbord-decoder bevindt zich op geheugenlocatie FF0CH. Op deze positie staat normaal C9H. C9H is de machinetaalinstruktie voor RET. RET komt overeen met RETURN in BASIC. Na de sprong naar deze hook gaat de ROM dus gewoon verder waar deze was gebleven.
Hier wordt echter een CALL (CD) naar de routine op locatie C600H gemaakt. Hier wordt gekeken of CTRL-STOP is ingedrukt. Als dat gebeurt, dan wordt er een tekst op het scherm geprint.
Door de LIST-statementhandler wordt ook naar een hook gesprongen (FF89H). Deze hook springt naar een routine die het eerste regelnummer van het programma op het hoogst mogelijke regelnummer "poket", waardoor het programma niet meer is te listen.

N.B. Zet altijd aan het begin van het te beveiligen programma:

```
5  CLEAR 200, &HC4FF
10 IFPEEK (32771) <> 255THEN L1= PEEK (32771)
   ELSE 20
20 POKE 32771, L1:POKE 32772, L2
```

Deze routine zorgt ervoor dat uw programma's altijd goed hun werk blijven verrichten.

```
Hisoft GEN Assembler. Page     1.

Pass 1 errors: 00

                        10 ; Beveiliging: CTRL-STOP
                        20 ;               en LIST
                        30 ;
                        40 ; (c) 1986 - H. Klopper
                        50 ;
4A24                    60 PTMESG:    EQU  #4A24        ;PRINT MESSAGE
                        70 ;
C500                    80            ORG  #C500
                        90 ;
C500  210CFF           100            LD   HL,#FF0C     ;INITIALISEREN
C503  36C3             110            LD   (HL),#C3     ;VAN DE HOOK
C505  210DFF           120            LD   HL,#FF0D     ;VOOR DE KEY-
C508  3600             130            LD   (HL),#00     ;BOARD DECODER.
C50A  210EFF           140            LD   HL,#FF0E     ;JP C500H
C50D  36C5             150            LD   (HL),#C5     ;
                       160 ;
                       170 ; LIST ONDERDRUK HOOK
                       180 ;
C50F  2189FF           190            LD   HL,#FF89     ;INITIALISEREN
C512  36C3             200            LD   (HL),#C3     ;VAN DE HOOK
C514  218AFF           210            LD   HL,#FF8A     ;VOOR DE LIST
C517  3600             220            LD   (HL),#00     ;ONDERDRUKKING.
C519  218BFF           230            LD   HL,#FF8B     ;JP C600H
C51C  36C6             240            LD   (HL),#C6     ;
```

```
                245 ;
                310 ;
C51E  CDB700    320             CALL #00B7          ;CTRL-STOP?
C521  DC25C5    330             CALL C,TEKST        ;JA? ==> PRINT TEKST
C524  C9        340             RET
C525            350 TEKST:
C525  21AEF3    360             LD   HL,#F3AE       ;SCHERMBREEDTE
C528  3628      370             LD   (HL),40        ;40 KOLOMMEN
C52A  CD6C00    380             CALL #006C          ;SCREEN 0
C52D  2159C5    390             LD   HL,MSG         ;TEKST HALEN
C530  E5        400             PUSH HL
C531  210101    410             LD   HL,#0101       ;XY
C534  CDC600    420             CALL #00C6          ;LOCATE
C537  E1        430             POP  HL
C538  CD244A    440             CALL PTMESG         ;PRINT ROUTINE
C53B  CD9F00    450             CALL #009F          ;INPUT$(1)
C53E  2166C5    460             LD   HL,MSG1
C541  CD244A    470             CALL PTMESG
C544  2190C5    480             LD   HL,MSG2
C547  CD244A    490             CALL PTMESG
C54A  CDC000    500             CALL #00C0          ;BEEP
C54D  CDC000    510             CALL #00C0
C550  CDC000    520             CALL #00C0
C553  CDC000    530             CALL #00C0
C556  C39B40    540             JP   #409B
                550 ;
C559  22        560 MSG:        DEFB 34
C55A  48454C41  570             DEFM "HELAAS... "
```

```
C564  2200        580             DEFB 34,0
C566  22          590 MSG1:       DEFB 34
C567  44697420    600             DEFM "Dit programma is niet te
C58E  2200        610             DEFB 34,0       'breaken'...  "
C590  22          620 MSG2:       DEFB 34
   C591  28632920   630             DEFM "(c) 1986 - Stichting
C5B8  2200        640             DEFB 34,0   tegen het copieren"
                  650 ;
                  660 ;  LIST ONDERDRUK ROUTINE
                  670 ;
C600              680             ORG  #C600
C600  E5          690             PUSH HL
C601  210380      700             LD   HL,32771
C604  36FF        710             LD   (HL),255
C606  210480      720             LD   HL,32772
C609  36FF        730             LD   (HL),255
C60B  E1          740             POP  HL
C60C  C9          750             RET
C60D              800             END

Pass 2 errors: 00

Table used:    72  from  1000
```

Naast de hiervoor geplaatste machinetaal/assembler-listing voor de machinetaalprogrammeur, vindt u hieronder de listing voor de MSX gebruiker zonder assembler.
De machinetaal is hier, zoals gebruikelijk, opgenomen in zogenaamde DATA-regels. Hierdoor is het mogelijk om het programma gewoon vanuit BASIC in te tikken.

Beseft u wel, dat als het programma eenmaal is gerund, u geen programma's meer kunt listen!

Veel sukses bij de toepassing in uw eigen programma-archief.

```
10 CLEAR 200,&HC4FF
20 REM **********************
30 REM *                      *
40 REM * BASIC "poke" pro-    *
50 REM * gramma voor de       *
60 REM * list beveiliging     *
70 REM *                      *
80 REM **********************
90  :
100 :
110 COLOR 1,14:SCREEN0:WIDTH40:KEYON
120 :
130 DATA 21,0C,FF,36,C3,21,0D,FF,36,00
140 DATA 21,0E,FF,36,C5,21,89,FF,36,C3
150 DATA 21,8A,FF,36,00,21,8B,FF,36,C6
160 DATA CD,B7,00,DC,25,C5,C0,21,AE,F3
170 DATA 36,28,CD,6C,00,21,59,C5,E5,21
180 DATA 01,01,CD,C6,00,E1,CD,24,4A,CD
190 DATA 9F,00,C3,9B,40,**
200 :
210 REM  HIER ZOUDEN DE MESSAGES MOETEN
220 REM  KOMEN. DEZE WORDEN NU ECHTER
```

```
230 REM  HANDMATIG IN HET GEHEUGEN GE-
240 REM  POKED.
250 :
260 REM  ORG C600H  ==>> LIST ONDERDR.
270 :
280 DATA E5,21,03,80,36,FF,21,04,80,36
290 DATA FF,E1,C9,**
300 :
310 REM ****************************
320 REM *                          *
330 REM * STRINGS IN HET GEHEUGEN *
340 REM * POKEN                    *
350 REM *                          *
360 REM ****************************
370 :
380 REM Alleen Helaas!!!! wordt ge-
390 REM print.
400 :
410 T=&HC559
420 A$="Helaas... "
430 POKE T,34
440 FOR X=1TOLEN(A$):M$=MID$(A$,X,1):POK
E X+T,ASC(M$):NEXT:POKEX+T,0
450 :
460 REM ************************
470 REM *                      *
480 REM * Het programma in het *
490 REM * geheugen poken       *
500 REM *                      *
510 REM ************************
520 :
530 T=&HC500: 'het programma
540 READA$:IFA$="**"THEN 550 ELSE POKE T
```

```
,VAL("&H"+A$):T=T+1:GOTO 540
550 T=&HC600: 'de listonderdrukker
560 READA$:IFA$="**"THEN 570 ELSE POKE T
,VAL("&H"+A$):T=T+1:GOTO 560
570 DEFUSR=&HC500:X=USR(0)
```

Invoerproblemen

Bij het zelf maken van BASIC-programma's komt het vaak voor dat u een keuzemenu opstelt. De keuze uit dit menu wordt meestal door middel van het "INPUT"-statement behandeld. Dit brengt echter een hele reeks vergelijkingen met zich mee.

In truuks & tips deel 3 hebben we reeds een routine gepubliceerd, die het mogelijk maakt om twee vergelijkingen te reduceren tot 1. De volgende routine werkt echter nog efficiënter. Ze maakt gebruik van het statement "INSTR", dat dient om een string uit een string te zeven.

Er wordt gekeken of het antwoord J, j. N of n is. Is dat het geval dan is het goed en wordt het programma vervolgd. Is de invoer echter niet overeenkomstig de bovenvermelde eisen dan zal net zolang worden gewacht tot dat wel het geval is.

Deze routine kunt u heel goed toepassen in uw eigen programma's waar menukeuze in voor komt.

```
10 REM ********************
20 REM *                  *
30 REM * INVOER CONTROLE *
40 REM * ROUTINE...       *
50 REM *                  *
60 REM ********************
70 :
80 COLOR 15,1:SCREEN0:WIDTH40
90 PRINT"Verder gaan (j/n)
100 A$=INPUT$(1):B=INSTR("jJnN",A$)
110 PRINT A$
120 IF B=0 THEN 100
```

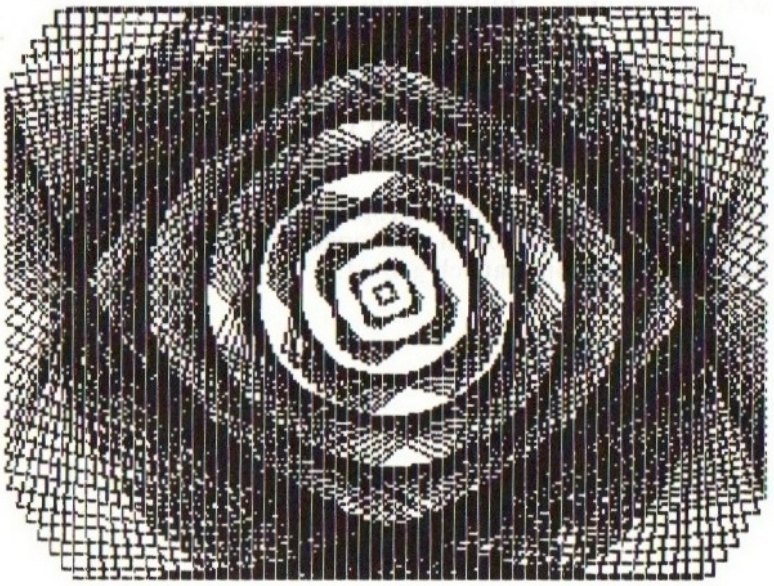
```
130 ON B GOSUB 150,150,170,170
140 END
150 PRINT"Dat was dus ja!"
160 RETURN
170 PRINT"Dat was dus nee!"
180 RETURN
```

Schermbreedte misbruiken

Normaliter zult u het "WIDTH"-statement alleen gebruiken als u de breedte van het scherm wilt aanpassen. Het volgende korte programmaatje maakt echter op heel slimme wijze gebruik van het "WIDTH"-statement. Het resultaat is toch wel verbluffend te noemen. Tik snel het programma in en aanschouw het zelf.

```
10 ' ******************
20 ' * SCHERMBREEDTE *
30 ' *  MISBRUIKEN   *
40 ' ******************
50 '
60 STOPON:ONSTOP GOSUB190
70 KEYOFF
80 FOR I=1TO20
90 PRINT"Stark-Texel"
100 FOR M=1 TO50:NEXT
110 WIDTHI
120 NEXT
130 FOR I=20TO1STEP-1
140 WIDTHI
150 PRINT"Stark-Texel"
160 FOR M=1 TO50:NEXT
170 NEXT
180 RUN
190 SCREEN0:WIDTH40:END
```

MSX grafisch



Grafische eigenschappen van huiscomputers trekken duidelijk de meeste aandacht van de gemiddelde computeraar. Dit verklaart hoogstwaarschijnlijk de grote vraag naar de nieuwe generatie MSX-2 computers. Deze computers beschikken over een grafische chip die de capaciteit van bijvoorbeeld een IBM-PC met kleurenkaart ver overtreft.

Een logisch gevolg van deze grote belanstelling is een behoorlijke reeks programma's die betrekking hebben op dit onderwerp. Zo kunt u in dit hoofstukje ondermeer terugvinden:

- Screen 2 coördinaten uitlezen
- Draaiend uitroepteken
- Kleuren instellen in Screen 1
- 5 sprites op een rij

en nog veel meer. Al deze programma's hebben tot doel u meer inzicht te geven in de mogelijkheden van uw MSX homecomputer op grafisch gebied.

De SCREEN 2 coördinaten uitlezen

Met de statements POS(X) en CSRLIN kunnen respectievelijk de X- en Y-coördinaten van de cursor uitgelezen worden in screen 0/1. Statements om de X- en Y-coördinaten van de grafische cursor uit te lezen ontbreken. Het volgende programma maakt dit wel mogelijk.

U zou de volgende routine kunnen toepassen in een tekenprogramma. In dit soort programma's is het namelijk vaak wenselijk om op de hoogte te zijn van de cursorlocatie.

De coördinaten bevinden zich in de systeemvariabelen. Adres &HFCB7 bevat de X-coördinaat, adres &HFCB9 de Y-coördinaat. Door deze adressen uit te lezen kunnen we beide coördinaten opvragen.
Onderstaand programma laat zien hoe dit gebeuren moet.

We kunnen de inhoud van de adressen natuurlijk ook veranderen. . .

```
10 '*********************************
20 '*                               *
30 '* DEZE ROUTINE MAAKT HET MOGE- *
40 '* LIJK OM DE X EN Y COORDINAAT *
50 '* VAN DE GRAFISCHE CURSOR UIT  *
60 '* TE LEZEN.                    *
70 '*                               *
80 '*********************************
90 :
100 SCREEN2
110 OPEN "GRP:" AS 1
```

```
120 PSET(30,30)
130 DRAW "U10R30D70"
140 :
150 GOSUB 230:'NU WORDEN DE X EN Y
                COORDINATEN OPGEVRAAGD
160 :
170 PSET (100,150):PRINT#1,"X COORINAAT:
";A
180 PSET (100,160):PRINT#1,"Y COORINAAT:
";B
190 GOTO 190
200 :
210 ' SUBROUTINE DIE DE X EN Y COORDINA-
      TEN UITLEEST
220 :
230 A=PEEK(&HFCB7)
240 B=PEEK(&HFCB9)
250 RETURN
```

De lopende cursor

Met het LOCATE-statement is het mogelijk om de cursor op een bepaalde plaats te positioneren. Bovendien kan men met dit kommando kiezen of de cursor in beeld moet blijven (LOCATE, ,1) of dat ze onzichtbaar moeten worden (LOCATE, ,0).

Het onzichtbaar maken van de cursor werkt altijd feilloos. Echter, het zichtbaar maken van de cursor is alleen mogelijk als de computer een invoer verwacht via het toetsenbord zoals bijvoorbeeld in 'direct mode'. In alle andere gevallen is de cursor onzichtbaar (bijvoorbeeld na het kommando CLOAD).

Dit probleem is op te lossen door de inhoud van een systeemvariabele te veranderen. De bedoelde variabele bevindt zich op adres &HFCA9. Als dit adres een 1 bevat, dan is de cursor altijd in beeld, bevat ze de waarde 0, dan is de cursor alleen in beeld als er een invoer via het toetsenbord wordt verwacht (de normale situatie).

Met een cursor die constant in beeld is zijn leuke grapjes uit te halen. Onderstaand programma toont er twee.

```
10 '********************************
20 '*                              *
30 '*     DE LOPENDE CURSOR        *
40 '*                              *
50 '********************************
60 :
70 A=1
80 :
90 ' ALS A=1 ==> CURSOR ALTIJD IN BEELD
```

```
100 ' ALS A=0 ==> CURSOR ALLEEN IN BEELD
                  BIJ WACHTEN OP INVOER
110 :
120 POKE&HFCA9,A
130 SCREEN0:KEYOFF:WIDTH40:CLS
140 :
150 FOR T=1 TO 10
160 PRINT"DEZE TEKST VERDWIJNT NIET OMDA
T
170 PRINT"DE CURSOR ER OVER HEEN LOOPT.
180 NEXT T
190 :
200 FOR Y=1 TO 3
210 FOR X=1 TO 33
220 LOCATE X,Y:FOR M=1 TO40:NEXT:NEXT
230 NEXT
240 :
250 '********************************
260 '*                              *
270 '*       LEUKE EFFECTEN         *
280 '*                              *
290 '********************************
300 :
310 CLS
320 FOR X=0 TO 799
330 PRINT CHR$(255);:NEXT
340 :
350 LOCATE 12,10:PRINT"ABCDEFGHIJKLM"
360 FOR T=1 TO 15
370 FOR X=12TO 25:LOCATE X,10
380 FOR S=1 TO 20:NEXT S
390 NEXT X:NEXT T
```

Multi kleuren sprites

Met het "PUTSPRITE" -kommando is het mogelijk om een sprite op het beeldscherm te laten verschijnen. Als u dit kommando geeft, dan dient u tevens een aantal gegevens te geven, zoals het spritenummer, de spritevorm, de coördinaten en de kleur.
Het kleur gegeven mag variëren van 0 t/m 15, ieder getal stelt een kleur voor. Het is dus niet mogelijk om een sprite te ontwerpen die uit meerdere kleuren bestaat.

De volgende truuk lost dit probleem op.

Door twee verschillende sprites, die samen een figuur vormen, gelijktijdig op dezelfde coördinaten te plaatsen lijkt het alsof ze samen een sprite vormen.
Als u iedere sprite een andere kleur geeft, dan ontstaat er dus een sprite die uit meerdere kleuren bestaat.
Om dit te verduidelijken volgt hieronder een voorbeeld programma.

```
10 REM*****************************
20 REM*                           *
30 REM* DEMONSTRATIE 2 KLEUREN    *
40 REM*        SPRITE             *
50 REM*                           *
60 REM*****************************
70 :
80  DATA 11111111
90  DATA 00000000
100 DATA 11111111
110 DATA 00000000
120 DATA 11111111
130 DATA 00000000
```

```
140 DATA 11111111
150 DATA 00000000
160 :
170 DATA 00000000
180 DATA 11111111
190 DATA 00000000
200 DATA 11111111
210 DATA 00000000
220 DATA 11111111
230 DATA 00000000
240 DATA 11111111
250 :
260 COLOR10,1,1
270 SCREEN2,1
280 GOSUB 460
290 SPRITE$(0)=B$
300 GOSUB 460
310 SPRITE$(1)=B$
320 :
330 KL=15:KT=6
340 FOR X=10 TO 200
350 KL=KL XOR 9
360 KT=KT XOR 9
370 :
380 ' * TWEE SPRITES OVER ELKAAR HEEN
390 '
400 PUTSPRITE 0,(X,50),KL,0
410 PUTSPRITE 1,(X,50),KT,1
420 FOR M=1 TO 200:NEXT
430 NEXT
440 GOTO 440
450 :
460 B$="":FOR X=1 TO 8
```

```
470 READ A$:B$=B$+CHR$(VAL("&b"+A$)):NEX
T
480 RETURN
```

Kleuren instellen in SCREEN 1

Screenmode 1 is eigenlijk een heel vreemde schermmode. U kunt tekst in 32 kolommen op het scherm plaatsen en ook gebruik maken van de ruime sprite mogelijkheden die uw MSX te bieden heeft. Grafisch stelt dit scherm op het eerste gezicht niet echt veel voor. Door een goede kennis van de opbouw van het Video RAM kunt u hele speciale dingen doen in deze schermmode.

Eén van deze mogelijkheden is dat u de kleuren van 8 opeenvolgende karakters kunt instellen door middel van het V-POKE-statement. Normaliter dient u dit voor elk karakter apart uit te rekenen. De volgende routine zorgt ervoor dat de computer voor u al het moeilijke werk uitvoert.
Tegelijk met het in te stellen karakter, veranderen er nog 7 andere karakters van kleur. Dit komt omdat de "karakter vorm tabel" 2047 bytes lang is. Uw MSX computer bezit 255 verschillende karakters. Een karakter bestaat uit een 8*8 matrix. 8*255=2040, dus is elk karakter gedefinieerd in de "karakter vorm tabel." De kleuren tabel zou dus eigenlijk even lang moeten zijn om er voor te zorgen dat elk karakter een eigen voor- en achtergrond kleur zou kunnen krijgen. De kleuren tabel is echter slechts 31 bytes lang. Dat betekent dat 1 byte in de kleuren tabel tegelijk acht karakters behandelt. Dat kunt u zo zien, want 8*31=248. Hierdoor verwijzen dus 8 karakters tegelijk naar een positie in de kleuren tabel.

Om u een beter inzicht te geven in de opbouw van het Video RAM geven we hieronder de indeling bij screen mode 1:

DECIMAAL	HEXADECIMAAL	BESCHRIJVING
0000-2047	0000H-07FFH	Karakter vorm tabel
6144-6911	1800H-1AFFH	Karakterposities op 't scherm
6912-7039	1B00H-1B7FH	Sprite attribute tabel
8192-8223	2000H-201FH	Kleuren tabel
14336-16383	3800H-3FFFH	Sprite vorm tabel

```
10 REM ***************************
20 REM *                         *
30 REM * Kleuren per karakter    *
40 REM * in: SCREEN 1            *
50 REM *                         *
60 REM ***************************
70 :
100 COLOR 15,1:SCREEN0:WIDTH40
110 BS=8192  :'begin kleurentabel
120 PRINT"Welk karakter wilt u een kleur
en ?"
130 A$=INPUT$(1)
140 PRINT
150 PRINT"Te kleuren karakter: ";A$
160 PRINT
170 A=INT(ASC(A$)/8)
180 PK=BS+A
190 INPUT"Welke voorgrondkleur (1-15)";V
R
200 INPUT"Welke achtergrondkleur (1-15)"
;AR
210 SCREEN1
220 B$=HEX$(VR)+HEX$(AR)
230 VPOKE PK,VAL("&H"+B$)
240 LOCATE 2,2:PRINT"Het karakter: ";A$
```

Het draaiende uitroepteken

Met de volgende routine is het mogelijk om elk karakter rond te laten draaien. Als u deze routine in uw eigen programma's verwerkt, dan zal het geheel een stuk levendiger worden. Naast de stilstaande saaie letters, zijn er nu ook draaiende spectaculaire letters te bewonderen.
Hoe werkt het programma?
Om de werking te verklaren nemen we het uitroepteken zoals dat in dit voorbeeld is verwerkt.
Het principe is vrij eenvoudig: we definiëren vier uitroeptekens. De eerste, een gewone rechtopstaande, de tweede, één die een kwart slag naar rechts is gedraaid, de derde, één die een halve slag is gedraaid en de vierde, één die een kwart slag naar links is gedraaid. De eerste is al aanwezig, de tweede vpoken we in het $-teken, de derde in het %-teken en de vierde in het &-teken.
Printen we deze vier tekens achter elkaar op dezelfde plaats, dan lijkt het alsof het uitroepteken draait.
Het programma zorgt ervoor dat de vier verschillende uitroeptekens gegenereerd worden, u hoeft ze dus niet zelf te maken.
Hoe verandert u het draaiende uitroepteken in een ander teken?
Ook deze vraag is snel te beantwoorden. U hoeft alleen alle uitroeptekens die in de listing voorkomen te veranderen in het door u gewenste teken. Een uitroepteken komt voor in de volgende regels: 120 (2 keer), 210, 320, 370.
De volgende tekens kunt u niet rond laten draaien daar zij door deze routine gebruikt worden: $-teken, %-teken en het &-teken.

```
10 '*******************************
20 '*                             *
30 '*  Dit programma laat het     *
```

```
40 '*  uitroepteken ronddraaien   *
50 '*                             *
60 '*******************************
70 :
80 SCREEN1:KEYOFF
90 CLEAR 200,&HE300 :DEFINT A-Z
100 LOCATE1,2:PRINT"Een ogenblikje a.u.b
."
110 FOR C=0 TO 3:TT=0:Q=0
120 FOR X=ASC("!")*8 TO ASC("!")*8+8
130 A=VPEEK(X):Q=Q+1
140 POKE&HE3FF+Q,A
150 NEXT X
160 FOR MM=7 TO 0 STEP-1
170 FOR X=0 TO 7
180 A=PEEK(&HE400+X+V)
190 B=(A OR (2^MM) )
200 IF B=A THEN GOSUB400
210 NEXT:VPOKE(ASC("$")+C)*8+TT,FF:VPOKE
 ASC("!")*8+TT,FF:FF=0:TT=TT+1:NEXTMM
220 NEXTC:CLS
230 LOCATE 1,5:PRINT"Let op het uitroep
teken"
240 :
250 ' *******************************
260 ' * draaien van uitroep tekens *
270 ' *******************************
280 :
290 FOR P=0 TO 2
300 FOR S=1 TO  5
310 :
320 LOCATE25+P,5:PRINT"!":GOSUB410
330 LOCATE25+P,5:PRINT"$":GOSUB410
```

```
340 LOCATE25+P,5:PRINT"%":GOSUB410
350 LOCATE25+P,5:PRINT"&":GOSUB410
360 NEXT
370 LOCATE25+P,5:PRINT"!":GOSUB410
380 NEXT
390 END
400 FF=(FF OR (2^X) ):RETURN
410 FOR SS=1 TO 40:NEXT :RETURN
```

16 bij 16 Sprites inlezen

MSX computers beschikken over een video chip en een BASIC die sprites kunnen verwerken. Het is mogelijk om zowel 8 bij 8 als 16 bij 16 sprites te definiëren.
Het ontwerpen van sprites kan het beste in binaire vorm (nullen en enen) geschieden. De enen tonen dan de vorm, de nullen de lege ruimten.
Voorbeeld:

```
00000000
00010000
00010000
11111110
00010000
00010000
00000000
00000000
```

In bovenstaand voorbeeld vormen de enen het '+' teken. De 8 bij 8 sprite bestaat uit 8 regels van elk 8 bits (pixels).
Het ontwerpen van 8 bij 8 sprites in binaire vorm onder BASIC levert geen problemen op. Wil men echter 16 bij 16 sprites ontwerpen (in binaire vorm), dan ontstaan er wel degelijk problemen. Een 16 bij 16 sprite dient namelijk opgebouwd te worden uit vier 8 bij 8 sprites. De eerste twee 8 bij 8 sprites vormen de linker helft, de tweede twee de rechter helft. Het is dus niet mogelijk om een 16 bij 16 sprite op te bouwen uit 16 gegevens die elk 16 bits (pixels) groot zijn.
Met behulp van onderstaande routine wordt dit probleem opgelost. Het is nu wel mogelijk om 16 bij 16 sprites op dezelfde manier te vormen als 8 bij 8 sprites.
De routine dient als volgt gebruikt te worden:

1 In regel 10 staan twee "DIM"-opdrachten. Indien u deze routine in een ander programma verwerkt, dan moet u deze opdrachten altijd overnemen.

2 Ontwerp een 16 bij 16 sprite in binaire vorm. LET OP: het eerste bit (pixel) dient altijd een 0 te zijn.

3 De sprite wordt in de variabele S$ gelezen door de subroutine aan te roepen die op regel 240 begint.
Dit gebeurt met "GOSUB 240"

4 Definieer nu de sprite als volgt: SPRITE$ (X) = S$.
Hierbij stelt X het spritevormnummer voor.

5 De sprite kan nu op het beeldscherm geprojekteerd worden.

In onderstaand programma is tevens een voorbeeld verwerkt. Er worden drie verschillende sprites gedefinieerd. Door deze sprites beurtelings op het beeldscherm te laten verschijnen, lijkt het alsof er zich een kronkelende worm voortbeweegt. Het is dus mogelijk om een sprite in "leven" te roepen door van een levend wezen verschillende vormen te creëren en deze beurtelings op het scherm te projekteren.

```
1 '*****************************
2 '*                           *
3 '*  DEZE ROUTINE MAAKT HET   *
4 '*    MOGELIJK OM 16*16      *
5 '*   SPRITES DIRECT IN TE    *
6 '*          LEZEN.           *
7 '*                           *
8 '*****************************
9 :
10 DIM B(16),C(16)
20 :
```

```
30 COLOR 15,1,1 :SCREEN2,2
40 LINE (0,150)-(255,196),4,BF
50 GOSUB 240
60 SPRITE$(0)=S$
70 GOSUB 240
80 SPRITE$(1)=S$
90 GOSUB 240
100 SPRITE$(2)=S$
110 FOR X=1 TO 255 STEP 2
120 NR=NR+1:IF NR>2 THEN NR=-1:A=2
130 PUTSPRITE0,(X ,142),15,NR+A
140 A=0
150 FOR M=1 TO 200:NEXT
160 NEXT
170 GOTO 170
180 :
190 '*******************************
200 '    SUBROUTINE OM 16*16
210 '    SPRITES IN TE LEZEN
220 '*******************************
230 :
240 S$="":FOR T=1 TO 16
250 READ A$:A$="&b"+A$:A=VAL(A$)
260 B(T)=INT(A/256):C(T)=(A AND&B0000000
011111111)
270 :
280 NEXT
290 FOR T=1 TO 16
300 S$=S$+CHR$(B(T)):NEXT
310 FOR T=1 TO 16
320 S$=S$+CHR$(C(T)):NEXT
330 RETURN
340 :
```

```
350 '**************************
360 '   GEGEVENS VOOR SPRITES
370 ' DIE TEZAMEN EEN BEWEGENDE
380 '         WORM VORMEN.
390 '**************************
400 :
410 DATA 0000000000000000
420 DATA 0000000000000000
430 DATA 0000000000000000
440 DATA 0000000000000000
450 DATA 0100000000000000
460 DATA 0111111111111111
470 DATA 0011011101110111
480 DATA 0,0,0,0,0,0,0,0,0
490 :
500 DATA 0000000000000000
510 DATA 0000000000000000
520 DATA 0000000000000000
530 DATA 0011110000001000
540 DATA 0111111000011000
550 DATA 0100001111110000
560 DATA 0000000111100000
570 DATA 0,0,0,0,0,0,0,0,0
580 :
590 DATA 0000000000000000
600 DATA 0000000000000000
610 DATA 0011001100000000
620 DATA 0111101110100000
630 DATA 0111111111100000
640 DATA 0100111011100000
650 DATA 0000111011100000
660 DATA 0,0,0,0,0,0,0,0,0
```

5 Sprites op een rij

MSX computers staan er om bekend dat ze zeer goed en op eenvoudige manier in staat zijn om grafische klusjes op te knappen.

Eén van de vele goede grafische eigenschappen van MSX-1 computers is de mogelijkheid om gelijktijdig over 32 sprites op het scherm te beschikken.

Een tekortkoming is echter dat er maximaal slechts 4 sprites op een horizontale rij mogen staan. De vijfde sprite zal niet zichtbaar zijn.
Bij MSX-2 computers is dit probleem opgelost en heeft men maximaal de beschikking over 8 sprites op een rij.
De MSX-1 gebruikers schieten hier echter niets mee op. Voor deze mensen denken wij echter een oplossing te hebben gevonden in de volgende routine.

De routine kijkt naar de inhoud van het VDP status register. Dit register bevat belangrijke informatie over de VDP.

Het STATUS REGISTER:

BIT 7: INTERRUPT BIT
BIT 6: 1 – Als 5 sprites op een rij
0 – Als ←4 sprites op een rij
BIT 5: 1 – Als twee sprites botsen
0 – Geen spritebotsing
BITS 0 t/m 4: 5e sprite nr.

Voor ons is dus vooral de informatie van BIT 6 belangrijk. Ook zouden we gebruiken kunnen maken van BITS 0 tot en

met 4 om te kijken welke sprite de vijfde is.

Als BIT 6 op 1 wordt gezet wordt er naar een routine gesprongen die sprite 5 en 4 om beurten aan en uit zet.
Hierdoor wordt sprite 5, welliswaar knipperend, toch zichtbaar op het scherm.

Dit knipperen van sprites kunt u ook wel tegenkomen in de vele spelen die er bestaan voor uw MSX computer. Nu weet u dus waarom dat gebeurt.

```
10 SF=PEEK(&HF3E7) :'SPRITE VLAG
20 :
30 INTERVAL ON:ONINTERVAL=25 GOSUB260
40 COLOR 15,1,1:SCREEN1:KEYOFF
50 VDP(6)=VDP(4):'KARAKTERSET IN SPRITE
                 SET COPIëREN.
60 X=1
70 FOR Z=20 TO 120 STEP 20:X=X+1
80 FOR I=10 TO 90 STEP 1
90 PUT SPRITE X,(Z,I),15,1
100 NEXT
110 NEXT
120 :
130 REM ******************
140 REM *                *
150 REM * 5e sprite lo-  *
160 REM * caliseren...   *
170 REM *                *
180 REM ******************
190 :
200 REM ***********************
210 REM *                     *
220 REM * 5e SPRITE ROUTINE *
```

```
230 REM *                    *
240 REM ************************
250 :
260 B=PEEK(&HF3E7)
270 :
280 REM * CHECKEN 5E SPRITE VLAG *
290 :
300 B=B AND &B01000000
310 IF B>0 THEN GOTO 450 ELSE RETURN
320 GOTO 320
330 :
340 REM ******************
350 REM * Bit 6 van      *
360 REM * VDP-register   *
370 REM * nr. 8 ==> 5e-  *
380 REM * sprite vlag.   *
390 REM *                *
400 REM ******************
410 :
420 POKE &HF3E7,SF :'VDP(8) RESET
430 LOCATE 5,5:PRINT"5 sprites op een ri
j."
440 :
450 REM ****************************
460 REM *                          *
470 REM * Sprites aan/uit zetten *
480 REM *                          *
490 REM ****************************
500 :
590 VPOKE &H1B14,207  :'sprite 5 uit
600 VPOKE &H1B14,90   :'sprite 5 aan
610 VPOKE &H1B18,207  :'sprite 6 uit
620 VPOKE &H1B18,90   :'sprite 6 aan
630 GOTO 590
```

Kleuren opvragen

Bij het programmeren van spelprogramma's of andere programma's waarbij u kleuren gebruikt is het vaak nodig om de achter-, voor- en borderkleur op te vragen.
Deze kleuren kunt u instellen door middel van het "COLOR" commando. De kleurcodes worden hierna opgeslagen in de zogenaamde systeemvariabelen. Dat betekent dat we op elk gewenst moment de kleur kunnen opvragen door een eenvoudige "PEEK"-opdracht.

Onderstaande routine laat u zien hoe alles in z'n werk gaat.

```
100 SCREEN0:WIDTH40
110 :
120 REM ******************
130 REM *                  *
140 REM * Achtergrond,     *
150 REM * Voorgrond en     *
160 REM * Border kleur     *
170 REM * opvragen...      *
180 REM *                  *
190 REM ******************
200 :
210 COLOR 14,1,4 :'
                - voorgrond   = 14
                - achtergrond= 1
                - border      = 4
220 SCREEN1:WIDTH32
230 :
240 :
250 FOR AANTAL=1TO10   :'AANTAL MAAL OPVR
AGEN
```

```
260 V=INT(RND(1)*12+2)
270 A=INT(RND(1)*12+2)
280 B=INT(RND(1)*12+2)
290 :
300 COLOR V,A,B
305 :
310 REM ******************
320 REM *                  *
330 REM * Opvraag rou-     *
340 REM * tine.            *
350 REM *                  *
360 REM ******************
370 :
380 LOCATE 5,2:PRINT"COLOR";V";";A;","B
390 LOCATE 5,5:PRINT"Voorgrondkleur    =
";PEEK(&HF3E9)
400 LOCATE 5,6:PRINT"Achtergrondkleur =
";PEEK(&HF3EA)
410 LOCATE 5,7:PRINT"Borderkleur       =
";PEEK(&HF3EB)
415 :
416 FOR T=1TO1500:NEXT :'PAUZE
417 :
420 NEXT
500 COLOR 14,1:SCREEN0:WIDTH40
```

Tekst op de grafische schermen

Bij de MSX computers is het niet zo eenvoudig om op de juiste manier tekst op een grafisch scherm (SCREEN 2/3) te printen. Allereerst moet door middel van 'OPEN"GRP:"AS1' de computer worden ingesteld voor grafisch schrijven.
Hierna dient de cursor door middel van het "PSET" of "DRAW BM" commando op de juiste schermpositie te worden gezet. Dit komt overeen met het statement "LOCATE." Beide systemen hebben echter een nadeel. "PSET" zet een puntje op het scherm en "DRAW BM" is niet variabel te gebruiken.

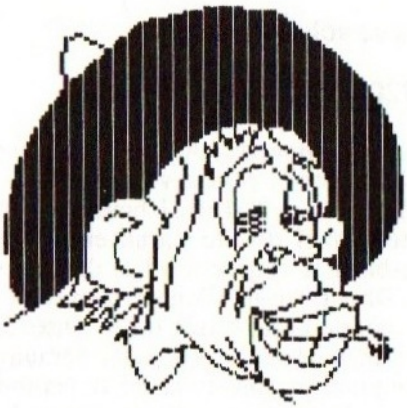
Onderstaande routine geeft de oplossing voor bovenstaand probleem. Er wordt een "LOCATE"-opdracht uitgevoerd door middel van het "PSET"-statement, waardoor de X- en de Y-coördinaten variabel worden. Daarbij is het puntjesprobleem voorbij omdat het "POINT"-statement er voor zorgt dat het puntje altijd gelijk is aan de achtergrondkleur.

Ter verduidelijking is de routine opgenomen in een kort demo-programma.

```
10 REM ******************
20 REM *                 *
30 REM * Het printen     *
40 REM * van tekst in    *
50 REM * grafische mode  *
60 REM *                 *
70 REM ******************
80 :
90 COLOR 14,1,1:SCREEN3
```

```
100 CLOSE:OPEN"grp:"AS1
110 :
120 REM ******************
130 REM *                  *
140 REM * standaard me-    *
150 REM * thode            *
160 REM *                  *
170 REM ******************
180 :
190 PSET (50,70),8:PRINT#1,"Stark
200 :
202 FOR T=1TO2000:NEXT:' PAUZE
203 CLS
204 :
210 REM ******************
220 REM *                  *
230 REM * De beste me-     *
240 REM * thode d.m.v.     *
241 REM * point(x,y)       *
242 REM *                  *
243 REM ******************
245 :
246 PSET (50,70),POINT(50,70):PRINT#1,"S
tark
247 :
250 GOTO 250
```

Peeks & pokes



Ook in dit deel zijn weer een groot aantal peeks en pokes terug te vinden. Onder andere de programma's die tot dit hoofdstuk behoren.

Probeert u eens de volgende poke:

```
POKE&HF920,&HC6
```

Geef het screen0 kommando. Als alles goed verlopen is, dan is uw normale karakterset geheel vernietigd.
De adressen &HF920 en &HF921 bevatten namelijk het beginadres van een tabel die de karakterset vormt. Normaal zijn deze adressen met respectievelijk de waarden &HBF en &H1B geladen. Deze twee getallen vormen te zamen &HIBBF, het beginadres van de karakterset tabel. Deze tabel wordt bij elke screen-opdracht door de computer geraadpleegd.
Door nu het beginadres van de tabel te veranderen, kunnen we ten allen tijde (ook in screen2 en screen3) een eigen karakterset maken...

Op adressen &HF40B en HF40C bevinden zich gegevens die te maken hebben met de circle-opdracht. Normaliter bevat deze systeemvariabele de waarde &H100. Verander deze waarde, bijvoorbeeld door POKE&HF40C,&HFF:POKE&-HF40B,&H50 en teken een cirkel als volgt.

```
10 SCREEN2:CIRCLE(125,96),80,15
20 GOTO 20
```

Ziet u een verschil zonder de POKE?

Het BASIC-geheugen verleggen

Met de opdracht CLEAR (aantal bytes voor variabelen), (hoogste geheugenlocatie door BASIC gebruikt) is het mogelijk om een deel van het geheugen te reserveren voor machinetaal c.q. data opslag.

In plaats van de hoogste vrije geheugenlocatie naar beneden te verschuiven (zoals de clear-opdracht doet), kunnen we natuurlijk ook de laagste geheugenlocatie verhogen waardoor het laagste gedeelte van het RAM niet voor BASIC programma's gebruikt kan worden. Dit laatste is echter niet zonder meer mogelijk. We dienen daartoe een tweetal systeemvariabelen te veranderen. De eerste systeemvariabele bevindt zich op adres &HF676 en &HF677, de tweede bevindt zich op adres &HFC48 en &HFC49. Beide variabelen dienen de laagste geheugenlocatie te bevatten (in hexadecimale vorm).

Een voorbeeld

Stel we willen de laagste RAM-locatie die door BASIC gebruikt mag worden op adres &HA000 vast leggen. U dient dan de voorgenoemde geheugenplaatsen met de volgende waarden te laden.

&HF677 en &HFC49 met &HA0.
&HF676 en &HFC48 met &H00.

Na het laden van deze geheugenplaatsen dient het 'NEW'-kommando gegeven te worden.
U kunt nu het aantal vrije geheugenplaatsen opvragen met: PRINT FRE (0).
Een tweede voorbeeld kunt u vinden in het volgende program-

ma.

Probeert u het volgende ook eens:

POKE &HF677, 0: POKE 7HF676, 0: POKE &HFC49,0:
POKE &HFC48,0

En vraag het aantal vrije geheugenplaatsen op.

```
1 '********************************
2 '*                              *
3 '* Dit programma demonstreert   *
4 '* het verleggen van het begin *
5 '* van het BASIC geheugen naar *
6 '* D000H                        *
7 '*                              *
8 '********************************
9 :
10 POKE&HF677,&HD0:POKE&HF676,&H0
20 POKE&HFC49,&HD0:POKE&HFC48,&H0
30 :
40 NEW
```

Top van het geheugen bepalen

Als u gebruik wilt maken van "Poke"- opdrachten dan dient u goed op de hoogte te zijn van de geheugen indeling.

U kunt gebruik maken van de bij de MSX toegepaste systeemvariabelen. Deze variabelen kunnen u allerlei bruikbare informatie over de toestand van uw computer geven.

Een voorbeeld van de toepassing van systeemvariabelen wordt gegeven in onderstaande routine. Hier wordt u informatie verschaft over het hoogste geheugen adres dat voor BASIC c.q. Machinetaal programma's beschikbaar is.

De BASIC programmeur hoeft zich, als hij geen gebruik maakt van "poke"-opdrachten geen zorgen te maken over de "top" van het geheugen. Bij het gebruik van "poke"-opdrachten, moet men bij het wegschrijven van databytes, deze bytes onder "top" wegschrijven. Bij het "poken" in de regionen boven de "top" bevindt u zich op glad ijs. Dit gebied is namelijk ingericht als werkgebied voor de computer. Zoals reeds eerder vermeld is het "poken" in dit gebied alleen aan te raden als u precies weet waar u mee bezig bent. De mahcinetaalprogrammeur weet vaak meer van de geheugenindeling van zijn MSX. Waar deze programmeur echter altijd rekening mee dient te houden, is dat hij nooit programma's schrijft die het gebied van de systeemvariabelen overschrijft. Het zogenaamde eindadres van machinetaal dient dan ook nooit het geheugen "top" adres te overschrijden.

Onderstaande routine geeft u het "top"-adres van het geheugen in hexadecimale vorm. Wilt u dit adres in decimale vorm, dan kunt u dit verkrijgen op de volgende manier:

PRINT &H. . . .

Op de plaats van de puntjes dient u dan het adres (zonder H) te plaatsen. Hierna komt het decimale equivalent op het scherm.

```
10 REM ***************************
20 REM *                         *
30 REM * Top van het geheugen    *
40 REM * bepalen.                *
50 REM *                         *
60 REM ***************************
70 :
80 COLOR 15,1:SCREEN0:WIDTH40
85 A$=HEX$((PEEK(&HFC4A))+256*(PEEK(&HFC
4B)))
90 LOCATE 2,5:PRINT"Het hoogste geheugen
 adres is: ";A$+"H"
100 END
```

Een platte cursor

MSX BASIC kent twee verschillende cursors, een vierkante en een platte. De laatste ontstaat als de computer zich in de insert mode bevindt.
Door op adres &HFCAA een 1 te poken ontstaat een platte cursor. U bevindt zich dan echter niet in de insert mode. De cursor wordt weer normaal als u een enter/return geeft of als u adres &HFCAA met een 0 laadt.

```
10 '*********************************
20 '*                               *
30 '*        PLATTE CURSOR          *
40 '*                               *
50 '*********************************
60 :
70 A=1
80 :
85 ' ALS A=1   ==> PLATTE CURSOR
86 ' ALS A=0   ==> NORMALE CURSOR
87 :
90 POKE &HFCAA,A:'platte cursor
```

Tijdelijk het keyboard uitschakelen

MSX computers hebben de beschikking over een keyboard-buffer. Als u een reeks van toetsen indrukt, dan worden ze gedecodeerd in deze buffer opgeslagen zodat de computer de tijd heeft om ze op het scherm te printen.

Als het opslaan van tekens in de buffer ongewenst is, dan zouden we het toetsenbord uit moeten schakelen, er worden dan geen karakters meer naar de buffer gezonden.
De volgende routine zorgt hiervoor, ze schakelt het toetsenbord tijdelijk uit (maximaal 5 seconden).
De routine verandert geheugenlocatie &HF3F6. Hoe hoger de waarde op dit adres, hoe langer het toetsenbord uitgeschakeld blijft.

```
10 REM ****************************
20 REM *                          *
30 REM * Tijdelijk het keyboard   *
40 REM * uitschakelen.            *
50 REM *                          *
60 REM ****************************
70 :
80 X=255
90 POKE &HF3F6,X
```

De onuitwisbare tekst

Onder MSX BASIC kunt u in screen 0 kiezen uit 23 of 24 regels. 23 regels als het funktietoetsendisplay aan staat, 24 als ze uit staat.
De computer "weet" welke keuze u gemaakt heeft en moet deze keuze ook onthouden. Daarom wordt het regelaantal in de systeemvariabelen weggeschreven. Dit gegeven bevindt zich op adres &HF3B1.
Door de inhoud van dit adres te veranderen wordt het mogelijk om het regelaantal een waarde aan te laten nemen anders dan 23 of 24. Het regelaantal kan nu variëren van 0 t/m 255.

U bent nu in staat om een deel van het beeldscherm te reserveren voor tekst die onveranderd moeten blijven.
Dit gebeurt op de volgende wijze.

1. Stel het regelaantal vast.

2. Print een tekst op een plaats boven het vastgestelde regelaantal. (b.v. als het regelaantal is vastgesteld op 10 dan dient de tekst op regel 11 en hoger gezet te worden).

3. Verander het regelaantal door op adres &HF3B1 de gewenste waarde te POKEN.

In principe heeft het geen zin om een waarde groter dan 24 op adres &HF3B1 te zetten omdat het beeldscherm maximaal 24 regels kan verwerken. Maar experimenteren met grotere getallen kan geen kwaad.

```
1 '**********************************
2 '*                                *
3 '*     VASTSTAANDE TEKST          *
```

```
5 '*                                  *
6 '*                                  *
7 '************************************
8 :
10 KEYOFF:SCREEN0:WIDTH40
15 :
20 LOCATE 2,16:PRINT"DEZE TEKST KAN NIET
 OVERSCHREVEN DOOR   EEN ANDERE TEKST.
30 :
40 POKE &HF3B1,15 :' AANTAL REGELS=15
50 :
60 LOCATE 0,0
```

Lopende funktietoetsen

Met het KEYOFF-kommando is het mogelijk om het funktietoetsendisplay uit te schakelen, de funktietoetsen verdwijnen dan direkt van het beeldscherm.

Als u echter op adres &HF3DE een 0 laadt dan wordt het steeds opnieuw weergeven van de inhoud van funktietoetsen buiten werking gesteld. De funktietoetsen zijn dan nog wel in beeld maar ze veranderen niet als bijvoorbeeld de SHIFT-toets ingedrukt wordt.

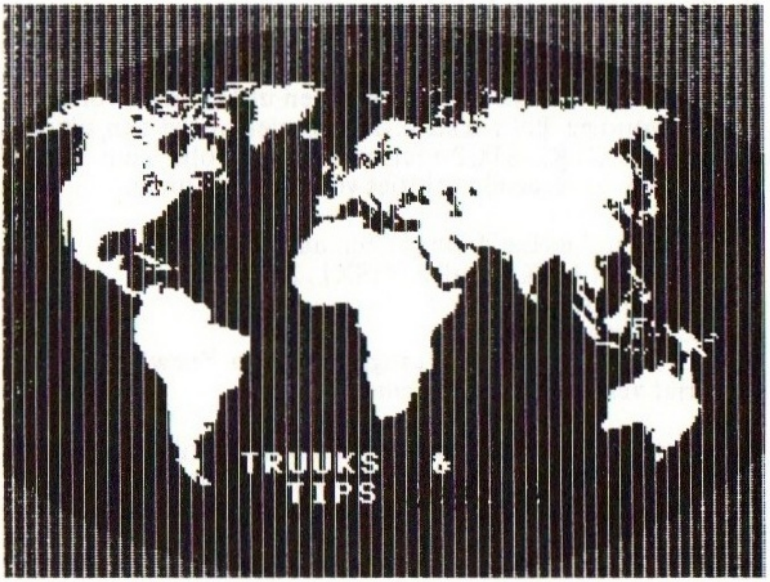
Wordt adres &HF3DE met een 1 geladen, dan heeft dit hetzelfde effekt als "KEYON".

Hiervan wordt gebruik gemaakt in onderstaande routine. De funktietoetsen worden in- en uitgeschakeld door de inhoud van adres &HF3DE steeds te veranderen. Daarnaast wordt er steeds een PRINT-opdracht gegeven. Het gevolg is dat de inhoud van de funktietoetsen naar boven loopt.

```
10 '**********************************
20 '*                                *
30 '*     LOPENDE FUNCTIE TOETSEN    *
40 '*                                *
50 '**********************************
60 :
70 SCREEN0:WIDTH40:KEYON
80 :
110 A=0
120 FOR T=1 TO 15
130 POKE&HF3DE,A:LOCATE 1,23
```

```
140 PRINT:POKE&HF3DE,1
150 KEY1,"color"
160 :
170 FOR M=1 TO 50:NEXT
180 NEXT
190 :
200 POKE&HF3DE,A
```

Utilities



Computers zijn eigenlijk nooit helemaal volmaakt. Er blijven bij de gebruikers altijd wel wensen over. De eerste huiscomputers waren meestal uitgerust met een zeer povere Basic interpreter.
Door middel van extra instrukties moest dit probleem dan later worden opgelost. Bij MSX computers is dit gelukkig veel beter gedaan, want de Basic bevat echt heel veel nuttige instrukties voor de hobbyprogrammeur.
Eén van de instrukties die MSX Basic rijk is, is bijvoorbeeld "ON STOP GOSUB." Deze instruktie werkt om onverklaarbare reden niet altijd voor 100% . Een utility in de vorm van een eenvoudige Poke-instruktie voorziet in het aan/uit zetten van de CTRL-STOP-toetsen zonder problemen. Verder hebben wij de volgende utilities voor u opgenomen.

- "Kijken" welke drives er zijn aangesloten
- De soort MSX bepalen (MSX1, 2 of 3)
- Een snelle RESET
- Een listing vertrager
- Een verzoek van bevestiging bij een "new"-opdracht.
- Het vervangen van statements

en nog vele andere makkelijke utilities. Wij zijn er van overtuigd, dat u veel gemak van deze hulpmiddelen kunt hebben als u ze op de juiste wijze toepast. Mochten er na het lezen van de truuks & tips delen 1 tot en met 4 nog dingen zijn die voor u nog niet zijn opgelost, schroom dan niet en stuur uw problemen op naar:

Uitgeverij Stark-Texel
Postbus 302
1794 ZG Oosterend

Misschien kunnen we uw probleem in een volgend deel dan aan de orde stellen en er een goede oplossing voor verzinnen.

CTRL-STOP uitschakelen

Het is mogelijk om met de funkties "STOP ON" en "ON STOP GOSUB regelnummer" een programma te beveiligen tegen inbraak volgens onderstaande methode:

```
10 STOP ON : ON STOP GOSUB 100

    het hoofdprogramma

100 RETURN
```

Deze manier van beveiligen is echter vaak lastig omdat het kommando "ON STOP GOSUB" niet altijd zonder problemen werkt. Zo treedt dit kommando buiten werking na een CLEAR-opdracht.

Gelukkig is dit probleem op een eenvoudige manier op te lossen.

Door adres &HFBB1 met de waarde 1 te laden, wordt de CTRL STOP funktie voor altijd uitgeschakeld. Indien dit adres een 0 bdvat, dan wordt de normale situatie weer verkregen.

Onderstaand voorbeeld toont hoe u voorgenoemd adres kunt veranderen.

```
10 '********************************
20 '*                              *
30 '* HET UITSCAKELEN VAN DE       *
40 '*    CTRL STOP FUNCTIE         *
50 '*          DEEL 2              *
60 '********************************
```

```
70 :
80 A=1
90 :
100 ' ALS A=1 DAN IS CTRL STOP UIT-
   GESCHAKELD. ALS A=0 DAN WERKT
   CTRL STOP WEER.
110 :
120 POKE&HFBB1,A
```

Drives aansluiting controle

MSX computers zijn zeer flexibel van opbouw. Zo kunt u zonder veel problemen twee of meer diskdrives aansluiten. Voor de programmeur zijn deze diskdrives een waar genoegen qua snelheid.

MSX-BASIC voorziet echter standaard niet in kommando's voor disk-gebruik. Deze kommando's bevinden zich in een extra 16 k disk-ROM. Een gevolg van deze methode is dat er wat vrij RAM-gebied verloren gaat.

Om te kijken hoeveel diskdrives er zijn aangesloten dient onderstaande routine. Hoe meer drives er zijn aangesloten hoe meer geheugen er verloren gaat. Veel software "kijkt" dan ook hoeveel drives er zijn aangesloten en aan de hand daarvan geeft men de boodschap: "Too much drives" als er teveel geheugen verloren gaat door de aangesloten drives.

Ook u kunt deze methode nu toepassen als u gebruik maakt van de onderstaande routine. Bovendien kunt u zien in welk "slot" de disk-ROM zich bevindt. Dit is vooral voor belang voor de machinetaalprogrammeur.

```
10 REM ***************************
20 REM *                         *
30 REM * Kijken of er een disk-  *
40 REM * drive is aangesloten.   *
50 REM *                         *
60 REM ***************************
70 :
100 COLOR 15,1:SCREEN0:WIDTH40
110 DIM A$(75):DIM B$(75)
120 A$=HEX$(PEEK(&HFFA7))
```

```
130 LOCATE ,5
140 IF A$="C9" THEN PRINT"- Diskdrive ni
et aangesloten." ELSE PRINT"- Diskdrive
aangesloten + Disk-ROM ":PRINT"  aktief.
150 :
160 REM ************************
170 REM *                      *
180 REM * Kijken hoeveel drives  *
190 REM * er zijn aangesloten.   *
200 REM *                      *
210 REM ************************
220 :
230 REM  * 1e interface *
240 :
250 A$(2)=HEX$(PEEK(&HFB21))
260 B$(2)=HEX$(PEEK(&HFB22))
270 :
280 REM  * 2e interface *
290 :
300 A$(3)=HEX$(PEEK(&HFB23))
310 B$(3)=HEX$(PEEK(&HFB24))
320 :
330 REM  * 3e interface *
340 :
350 A$(4)=HEX$(PEEK(&HFB25))
360 B$(4)=HEX$(PEEK(&HFB26))
370 :
380 REM  * 4e interface *
390 :
400 A$(5)=HEX$(PEEK(&HFB27))
410 B$(5)=HEX$(PEEK(&HFB28))
```

```
420 :
430 REM ***************************
440 REM *                         *
450 REM * Informatie over drives *
460 REM * printen.                *
470 REM *                         *
480 REM ***************************
490 :
500 PRINT:PRINTSTRING$(40,ASC("_")):PRIN
T
510 PRINT"- Aantal drives aan interface
1: ";A$(2)
520 PRINT"  Aangesloten op slot adres
 : ";B$(2)+"H"
530 :
540 PRINT"- Aantal drives aan interface
2: ";A$(3)
550 PRINT"  Aangesloten op slot adres
 : ";B$(3)+"H"
560 :
570 PRINT"- Aantal drives aan interface
3: ";A$(4)
580 PRINT"  Aangesloten op slot adres
 : ";B$(4)+"H"
590 :
600 PRINT"- Aantal drives aan interface
4: ";A$(5)
610 PRINT"  Aangesloten op slot adres
 : ";B$(5)+"H"
620 :
630 END
```

Welke soort MSX?

MSX-computers zijn er tegenwoordig in alle soorten en maten. De een heeft een ingbouwde diskdrive, de ander een RS-232-C interface etc.

Sinds enige tijd is er echter ook een werkelijke verbetering doorgevoerd wat betreft de MSX- standaard specificaties.

Deze verbetering wordt betiteld als MSX-2. De belangrijkste kenmerken van deze standaard zijn:

- 80 kolommen
- 512 kleuren (512 * 212 beeldpunten)

Om het nu mogelijk te maken voor software ontwikkelaars om te zien met wat voor MSX machine men te maken heeft, is er in het ROM een herkenningsbyte opgenomen. Dit herkenningsbyte bevindt zich op 2DH.

De byte kan verschillende waarden aannemen, te weten:

- 0 ==» MSX-1
- 1 ==» MSX-2
- 2 ==» Waarschijnlijk MSX-3

Onderstaande routine geeft een voorbeeld van het gebruik van deze "systeem herkennings byte".

```
10 REM ******************************
20 REM *                            *
30 REM * BEPALEN MET WELKE MSX      *
40 REM * VERSIE WE TE MAKEN HEBBEN *
50 REM *                            *
```

```
60 REM ******************************
70 :
80 COLOR 15,1:SCREEN0:WIDTH40
100 X=PEEK(&H2D)
110 LOCATE 5,5
120 IF X=0 THEN PRINT"Dit is een MSX-I c
omputer."
130 IF X=1 THEN PRINT"Dit is een MSX-II
computer."
140 IF X=2 THEN PRINT"Dit is waarschijnl
ijk een MSX-III":PRINT"computer.
150 END
```

Reset naar de BASIC editor

In het vorige deel publiceerden we een reset die door middel van de ESC-toets op te roepen is.
Die softwarematige reset is eigenlijk alleen nuttig voor MSX gebruikers die geen computer met een hardware reset bezitten. In tegenstelling tot de bovengenoemde reset is de nu volgende reset wel voor iedere gebruiker nuttig omdat ze veel sneller werkt. Na het resetten wordt namelijk direkt naar de BASIC editor gesprongen. Het vrij lange opstartverhaal blijft daardoor achterwege.

Onderstaande reset maakt wederom gebruik van een systeemvariabele; in dit geval bevindt het gegeven zich op adres &HFBB0.
Als dit adres een 1 bevat dan is een reset mogelijk. Als adres &HFBB0 een 0 bevat dan is geen reset mogelijk.
Tijdens het opstarten wordt dit adres geladen met een 0 en verder niet meer veranderd, maar we kunnen het natuurlijk zelf met een andere waarde laden.

Als de reset ingeschakeld is, dan kan ze uitgevoerd worden door gelijktijdig op de CTRL, SHIFT, GRAPH- en CODE-toetsen te drukken. De Ok prompt verschijnt dan direkt.
Het volgende programma toont u de werking.

OPMERKINGEN:

1. Als u een reset geeft terwijl screen mode 2 of 3 aktief is, dan lijkt het alsof de computer ophangt. Niets is echter minder waar, door het ingeven van "SCREEN 0" + ENTER/RETURN keert de computer naar het tekstscherm terug.
2. Ook deze reset werkt alleen als de "KEYBOARD SCAN" ingeschakeld is. Dit kunt u controleren door de CAPS-

LOCK-toets in te drukken. Als het lampje aan of uit gaat, dan is een reset mogelijk.

```
10 '*********************************
20 '*   RESET NAAR BASIC EDITOR     *
30 '*                               *
40 '* NA HET GELIJKTIJDIG INDRUKKEN *
50 '* VAN DE TOETSEN CTRL,SHIFT,GRP *
60 '* EN CODE VOLGT EEN RESET       *
70 '*********************************
80 :
90 STOPON:ONSTOP GOSUB 230
100 :
110 A=1
120 :
130 'ALS A=1 DAN IS DE RESET
     INGESCHAKELD. ALS A=0 DAN
     IS ZE UITGESCHAKELD
140 :
150 POKE &HFBB0,A
160 CLS
170 PRINT"PROBEER HET PROGRAMMA TE BREAK
EN"
180 PRINT"DIT ZAL NIET LUKKEN.
185 PRINT:PRINT:PRINT
190 PRINT"DRUK GELIJKTIJDIG OP DE CTRL,
SHIFT
200 PRINT"GRAPH EN CODE TOETSEN...."
210 GOTO 210
220 :
230 RETURN
```

Langzame lister

Het LIST-kommando stelt u in staat om een geheel BASIC programma te doorlopen. Na het ingeven van het woord LIST vertaalt de computer de gegevens die betrekking hebben op BASIC in een voor u begrijpelijke taal.
De computer weet dat het BASIC geheugen bij een 64K computer op adres 8000H begint. Verder weet hij ook dat een BASIC programma wordt beëindigd door een aantal nullen.
Na het LIST-kommando vliegen alle BASIC regels aan uw ogen voorbij. Tijd om ze op uw gemak te bekijken is er niet, tenzij u gebruikt maakt van de stop-toets. Maar ook dan bent u meestal te laat. Daarom hebben we onderstaand programma ontwikkeld. Het stelt u in staat om de LIST-snelheid naar wens in te stellen. Ga als volgt te werk.

1 RUN het programma
2 geef nu het LIST-kommando

De listsnelheid is nu op z'n laagst. U heeft alle tijd om een BASIC regel goed te bekijken.
Met behulp van "POKE &HE005,X" kunt u de listsnelheid veranderen. De variabele X mag tussen de 0 en de 255 variëren. Indien u X de waarde 0 laat aannemen, dan is de listsnelheid normaal. Als X=255 dan is de listsnelheid op z'n laagst.

```
1 REM ********************************
2 REM *     LANGZAME LISTING         *
3 REM * VERANDER DE LIST SNELHEID    *
4 REM * MET POKE&HE005,T. T MAG      *
5 REM * VARIEREN VAN 0-255. ALS      *
6 REM * T=255, DAN IS DE LISTING     *
```

```
7 REM * SNELHEID HET LAAGST.        *
8 REM ******************************
9 :
10 DATA f3              :'di
20 DATA e5              :'push hl
30 DATA f5              :'push af
40 DATA 21,ff,ff        :'ld hl,ffffh
50 DATA 2b              :'dec hl
60 DATA 7c              :'ld a,h
70 DATA fe,0            :'cp 0
80 DATA f5,f1           :'push af
90 DATA f5,f1           :'pop  af
100 DATA f5              :'push af
110 DATA f1              :'pop  af
120 DATA c2,06,e0        :'jp nz,e006h
130 DATA f1              :'pop af
140 DATA e1              :'pop hl
150 DATA fb              :'ei
160 DATA c9              :'ret
170 :
180 CLEAR 200,&HE000
190 :
200 FOR X=&HE000 TO &HE016:READA$
210 POKE X,VAL("&h"+A$):NEXT
220 :
230 POKE &HFF8F,&H0
240 POKE &HFF90,&HE0
250 POKE &HFF8E,&HCD
260 :
270 KEY 1,"POKE&HE005,"
```

New, zeker weten

Een van de eerste kommando's die u als MSX gebruiker heeft leren kennen, is "NEW". Het is u waarschijnlijk snel duidelijk geworden dat u hiermee voorzichtig dient om te gaan. Eén vergissing en uren werk kan verloren gaan.
Sommige BASIC versies beschikken over een kommando dat een NEW weer ongedaan maakt. MSX BASIC beschikt hier niet over. Daarom is het eigenlijk noodzakelijk om een extra beveiliging in te bouwen net zoals dat het geval is bij MSX DOS. Als u het kommando "DEL *.*" ingeeft dan wordt er gevraagd of u het zeker weet. Beantwoordt u de vraag met ja, dan wordt de inhoud van de gehele schijf gewist.
Onderstaand programma realiseert ook zo'n beveiliging. Als u, nadat het programma gelopen heeft, het kommando NEW ingeeft, dan wordt er om een bevestiging gevraagd. Het NEW kommando wordt alleen uitgevoerd als u de vraag met de kleine letter j beantwoordt. In alle andere gevallen blijft het programma behouden.

Uitleg voor gevorderden

Ook deze routine maakt gebruik van een Hook. In dit geval de "MAIN LOOP". Zodra de computer de TOKEN van NEW ontdekt, dan wordt er naar een subroutine gesprongen. Deze subroutine print de text "zeker weten" en wacht totdat er een letter "j" is, pas dan wordt het NEW-kommando gegeven. In alle andere gevallen wordt er terug gesprongen naar de BASIC.

```
10 ' ****************************
20 ' *                          *
30 ' *                          *
40 ' *     NEW zeker weten      *
```

```
50 ' *                              *
60 ' *                              *
70 ' *                              *
80 ' ********************************
90 :
100 DATA FE,94     :'CP 94
110 DATA CA,06,E0 :'JP Z,E006
120 DATA C9        :'RET
130 DATA 33        :'INC SP
140 DATA 33        :'INC SP
150 DATA 23        :'INC HL
160 DATA E5        :'PUSH HL
170 DATA 21,00,E1 :'LD HL,E100H
180 DATA CD,24,4A :'CALL 4A24
190 DATA CD,9f,00 :'CALL 009F
200 DATA FE,6A     :'CP 6AH  (j)
210 DATA E1        :'POP HL
220 DATA C0        :'RET NZ
230 DATA 3E,C9     :'LD A,C9H
240 DATA 32,43,FF :'LD (FF43),A
250 DATA 3E,94     :'LD A,94
260 DATA CD,46,46 :'CALL 4646H
270 DATA 3E,C3     :'LD A,C3H
280 DATA 32,43,FF :'LD (FF43),A
290 DATA c3,28,41 :'JP 4128H
300 DATA "**"
310 :
320 CLEAR 200,&HE000
330 T=&HE000
340 READ A$:IF A$="**" THEN 350 ELSE
POKE T,VAL("&H"+A$):T=T+1:GOTO340
350 POKE&HFF44,&H0:POKE&HFF45,&HE0
360 POKE &HFF43,&HC3
```

```
370 :
380 A$="Zeker weten j/n    "
390 A=LEN(A$):B=&HE101:POKE&HE113,0
400 FOR M=1 TO A: C$=MID$(A$,M,1)
410 POKE B,ASC(C$):B=B+1:NEXT
420 POKE&HE100,34:POKE&HE112,34
```

Statements vervangen

Als u in BASIC programmeert heeft u vast wel eens de behoefte aan het veranderen van BASIC statements in andere statements. Tekstverwerkingssystemen bieden u de mogelijkheid om woorden te zoeken en daarna door een ander woord te vervangen.

Uw MSX computer biedt door middel van onderstaande routine weliswaar niet al de mogelijkheden die een tekstverwerker u biedt, maar het geeft toch een behoorlijk alternatief.

Er wordt gebruik gemaakt van het feit dat MSX computers alle BASIC-statements opslaan in de vorm van tokens. Een voorbeeld wordt gegeven door het end-statement te veranderen in goto. Deze end-statements bevinden zich in regel 100 en lager.

Belangrijk is het feit, dat u het programma waarin u statements wilt vervangen onder de routine plaatst. De routine eindigt na regel 33.

Belangrijk: tikt u de routine PRECIES zo over als deze er staat (inklusief alle "REMS").

De tokens die hier gebruikt worden (in regel 20) zijn:

END: 81H
GOTO: 89H

Hieronder vindt u een lijst van overige BASIC tokens:

AUTO	A9H	AND	F6H

ABS	06H	ATN	0EH
ASC	15H	ATTR$	E9H
BASE	C9H	BSAVE	D0H
BLOAD	CFH	BEEP	C0H
BIN$	1DH	CALL	CAH
CLOSE	B4H	COPY	D6H
CONT	99H	CLEAR	92H
CLOAD	9BH	CSAVE	9AH
CSRLIN	E8H	CINT	1EH
CSNG	1FH	CDBL	20F
CVI	28H	CVD	2AH
COS	0CH	CHR$	16H
CIRCLE	BCH	COLOR	BDH
CLS	9FH	CMD	D7H
DELETE	A8H	DATA	84H
DIM	86H	DEFSTR	ABH
DEFINT	ACH	DEFSNG	ADH
DEFDBL	AEH	DSKO$	D1H
DEF	97H	DSKI$	EAH
DSKF	26H	DRAW	BEH
ELSE	A1H	END	81H
ERASE	A5H	ERROR	A6H
ERL	E1H	ERR	E2H
EXP	0BH	EOF	2BH
EQV	F9H	FOR	82H
FIELD	B1H	FILES	B7H
FN	DEH	FRE	0FH
FIX	21H	FPOS	27H
GOTO	89H	GO TO	89H
GOSUB	8DH	GET	B2H
HEX$	1BH	INPUT	85H
IF	8BH	INSTR	E5H
INT	05H	INP	10H
IMP	FAH	INKEY$	ECH
IPL	D5H	KILL	D4H

KEY	CCH	KEY	CCH
LPRINT	9DH	LLIST	9EH
LPOS	1CH	LET	88H
LOCATE	D8H	LINE	AFH
LOAD	B5H	LSET	B8H
LIST	93H	LFILES	BBH
LOG	0AH	LOC	2CH
LEN	12H	LEFT$	01H
LOF	2DH	MOTOR	CEH
MERGE	B6H	MOD	FBH
MKI$	2EH	MKS$	2FH
MKD$	30H	MID$	03H
MAX	CDH	NEXT	83H
NAME	D3H	NEW	94H
NOT	E0H	OPEN	B0H
OUT	9CH	ON	95H
OR	F7H	OCT$	1AH
OFF	EBH	PRINT	91H
PUT	B3H	POKE	98H
POS	11H	PEEK	17H
PSET	C2H	PRESET	C3H
POINT	EDH	PAINT	BFH
PDL	24H	PAD	25H
PLAY	C1H	RETURN	8EH
READ	87H	RUN	8AH
RESTORE	8CH	REM	8FH
RESUME	A7H	RSET	B9H
RIGHT$	02H	RND	08H
RENUM	AAH	SCREEN	C5H
SPRITE	C7H	STOP	90H
SWAP	A4H	SET	D2H
SAVE	BAH	SPC(	DFH
STEP	DCH	SGN	04H
SQR	07H	SIN	09H
STR$	13H	STRING$	E3H

SPACE$	E3H	SOUND	C4H
STICK	22H	STRIG	23H
THEN	DAH	TRON	A2H
TROFF	A3H	TAB(	DBH
TO	D9H	TIME	CBH
TAN	ODH	USING	E4H
USR	DDH	VAL	14H
VARPTR	E7H	VDP	C8H
VPOKE	C6H	VPEEK	18H
WIDTH	AOH	WAIT	96H
XOR	F8H		

Adres 3D26H in het ROM bevat de token-tabel voor afzonderlijke karakters.

+ F1H	- F2H	^ F5H	' E6H	= EFH	
* F3H	/ F4H	> FCH	< FOH	\ FCH	

```
1 REM ***************************
2 REM *                         *
3 REM * Statement vervangen     *
4 REM * in een BASIC programma.*
5 REM *                         *
6 REM ***************************
7 :
8  REM * Begin van BASIC bepalen *
9  :
10 A(1)=((PEEK(&HF676))+256*(PEEK(&HF677
)))
11 :
12 REM * Einde van BASIC bepalen *
13 :
14 A(2)=((PEEK(&HF6C2))+256*(PEEK(&HF6C3
)))
```

```
15 :
16 REM * tokens veranderen *
17 :
18 FOR X=A(1)+&H337 TO A(2)
19 A=PEEK(X)
20 IF A=&H81 AND PEEK(X-1)<>ASC("=") THE
N POKE X,&H89 :'end ==>> goto
21 NEXT
22 :
23 REM * einde dit programma:8338H *
24 :
25 REM ***************************
26 REM *                         *
27 REM * Plaats hieronder d.m.v.*
28 REM * een merge het programma*
29 REM * dat moet worden aange- *
30 REM * past.                   *
31 REM *                         *
32 REM ***************************
33 :
100 END  :REM ****  end ==> goto  ****
110 END  :REM ****  end ==> goto  ****
120 END  :REM ****  end ==> goto  ****
```

Twee nieuwe instrukties

Vaak is het wenselijk om of alleen hoofd- of alleen kleine letters tijdens een input routine te ontvangen. Als we kleine letters wensen, dan mag het CAPSLOCK lampje niet branden. Het vervelende is echter dat het niet zondermeer mogelijk is om te "kijken" of CAPSLOCK in werking is.
De volgende routine doet dit probleem verdwijnen. Ze voegt namelijk twee nieuwe instrukties toe.

De eerste instruktie: het ' teken.

De tweede instruktie: het ~ teken.

Met de eerste instruktie is het mogelijk om vanuit BASIC een BASIC programma de CAPSLOCK funktie in te schakelen (hoofdletters). Met de tweede instruktie kunt u de hoofdletters weer uitschakelen.
Onderstaand programma laat zien hoe u de nieuwe instrukties dient te gebruiken.

```
1 '*******************************
2 '*  TWEE NIEUWE INSTRUKTIES    *
3 '*Dit programma voegt 2 nieuwe *
4 '*instrukties toe. Met het '   *
5 '*teken worden de hoofdletters *
6 '*geactiveerd. Met het~ teken  *
7 '*worden de kleinde geactiveerd*
8 '*******************************
9 :
10 DATA FE,60     :'CP 60   (`)
20 DATA CA,0B,E0 :'JP Z,E00B
30 DATA FE,7E     :'CP 7E   (~)
40 DATA CA,16,E0 :'JP Z,E016
```

```
50 DATA C9          :'RET
60 DATA 33          :'INC SP
70 DATA 33          :'INC SP
80 DATA 23          :'INC HL
90 DATA E5          :'PUSH HL
100 DATA 3E,01      :'LD A,1
110 DATA 32,AB,FC:'LD (FCABH),A
120 DATA E1         :'POP HL
130 DATA C9         :'RET
140 DATA 33         :'INC SP
150 DATA 33         :'INC SP
160 DATA 23         :'INC HL
170 DATA E5         :'PUSH HL
180 DATA 3E,00      :'LD A,0
190 DATA 32,AB,FC:'LD (FCABH),A
200 DATA E1         :'POP HL
210 DATA C9,"**"  :'RET
220 :
230 CLEAR 200,&HE000
240 T=&HE000
250 READ A$:IF A$="**" THEN 260 ELSE
POKE T,VAL("&H"+A$):T=T+1:GOTO250
260 POKE&HFF44,&H0:POKE&HFF45,&HE0
270 POKE &HFF43,&HC3
280 :
290 `: REM dit teken zet de hoofd-
        letters aan.
300 :
310 ~: REM dit teken zet de hoofd-
          letters uit.
```

Funktietoetsen bewaren en oproepen

In het vorige deel bespraken we een routine om een set funktietoetsen naar diskette of cassette te schrijven. Het nadeel hiervan is dat dit niet snel genoeg gaat.
Met het onderstaande programma hebben we met die tijdvertraging afgerekend. De funktietoetsen worden namelijk in het RAM opgeslagen en kunnen direkt opgevraagd worden. Het is zelfs mogelijk om in no time over meer dan 15 verschillende sets funktietoetsen te beschikken. Het programma dient als volgt bediend te worden.
Nadat het programma gelopen heeft, kunt u met behulp van het "X=USR (0)"-kommando een set funktietoetsen wegschrijven. De computer kent aan elke weggeschreven set een nummer toe. De eerste weggeschreven set krijgt nummer 1 mee, de tweede nummer 2, de derde nummer 3 enz.
Met behulp van het kommando "X=USR1 (A*160)" kunt u iedere weggeschreven set direkt aanroepen. De variabele A in het kommando dient het gewenste setnummer te bevatten. Om het geheel wat te verduidelijken volgen hieronder enkele voorbeelden.

Het wegschrijven van een set:

Tik in: X=USR(0) + (ENTER/RETURN).

Setnummer 1 is nu weggeschreven.

Tik in : KEY1, "SET2" + (ENTER/RETURN).
Tik in: X=USR(0) + (ENTER/RETURN).

Setnummer 2 is nu weggeschreven.

Tik in: KEY1, "SET3" + (ENTER/RETURN).
Tik in: X=USR(0) + (ENTER/RETURN).

Setnummer 3 is nu weggeschreven.

Het terug halen van een set:

Tik in: X=USR1 (1*160) + (ENTER/RETURN).

Druk op de "SHIFT"-toets.
Als alles goed is verlopen, dan heeft u set 1 zojuist terug geroepen.

Tik in: X=USR1 (2*160) + (ENTER/RETURN).

Druk op de "SHIFT"-toets.
Nu heeft u set 2 opgeroepen.

Tik in: X=USR1 (3*160) + (ENTER/RETURN).

Als u op de "SHIFT" toets drukt, dan zal de derde set verschijnen die herkenbaar is aan de inhoud van de eerste funktietoets.

LET OP: Het is alleen mogelijk om een set terug te halen die eerst is weggeschreven. Probeert u een niet gedefinieerde set op te roepen, dan zullen de funktietoetsen onzin bevatten.

Heeft u een reeks funktietoetsen weggeschreven en wilt u ze op diskette of tape bewaren, dan dient u het volgende in te geven:

BSAVE "FUNCT" , &HD500, A*160 + &HD500

Hierbij stelt A het aantal weggeschreven sets voor. In het bovenstaande voorbeeld is de waarde van A gelijk aan 3.

Met de instruktie BLOAD "FUNCT" kunt u de sets weer in het geheugen laden.

```
10 '**********************************
20 '*                                *
30 '* FUNCTIE TOETSEN IN HET GEHEU- *
40 '* GEN OPLAAN EN TERUG ROEPEN.    *
50 '* MET X=USR(0) WORDT EEN SET     *
60 '* FUNCTIE TOETSEN WEGGESCHREVEN.*
70 '* X=USR1(A*160) HAALT SET A WEER*
80 '* TERUG.      0<A>15             *
90 '**********************************
100 :
110 DATA 21,00,D5:'LD HL,D500H
120 :
130 ' DIT ADRES (E1F0H) BEVAT DE
   PLAATS WAAR DE EERSTVOLGENDE
FUNCTIE TOETS SET HEEN KAN.
140 :
150 DATA 22,F0,E1:'LD (E1F0H),HL
160 DATA 3E,00    :'LD A,0
170 DATA 32,F2,E1:'LD (E1F2H),A
180 DATA C9       :'RET
190 :
200 ' DIT ADRES (E1F2H) BEVAT HET
   AANTAL OPGESLAGEN SETS.
210 :
220 :
230 CLEAR 200,&HD500
240 :
250 FOR X=&HE200 TO &HE20B:READ A$:POKE
X,VAL("&H"+A$):NEXT:DEFUSR=&HE200:X=USR(
0)
```

```
260 :
270 ' BEGIN VAN HET PROGRAMMA
280 :
290 ' OPSLAG ROUTINE
300 :
310 DATA F3           :'DI
320 DATA ED,5B,F0,E1:'LD DE,(E1F0H)
330 DATA 21,7F,F8     :'LD HL,F87FH
340 DATA 01,A0,00     :'LD BC,A0H
350 DATA ED,B0        :'LDIR
360 DATA 2A,F0,E1     :'LD HL,(E1F0H)
370 DATA 11,A0,00     :'LD DE,A0H
380 DATA 19           :'ADD HL,DE
390 DATA 22,F0,E1     :'LD (E1F0H),HL
400 DATA 3A,F2,E1     :'LD A,(E1F2H)
410 DATA 3C           :'INC A
420 DATA 32,F2,E1     :'LD (E1F2H),A
430 DATA FB           :'EI
440 DATA C9,"**"      :'RET
450 :
460 RESTORE 310:T=&HE210
470 READ A$:IF A$="**" THEN 480 ELSE   P
OKE T,VAL("&H"+A$):T=T+1:GOTO470
480 DEFUSR=&HE210
490 :
500 'OPHAALROUTINE
510 :
520 DATA F3           :'DI
530 DATA 2A,F8,F7     :'LD HL,(F7F8H)
540 DATA 11,60,D4     :'LD DE,D460H
550 DATA 19           :'ADD HL,DE
560 DATA 11,7F,F8     :'LD DE,F87FH
570 DATA 01,A0,00     :'LD BC,A0H
```

```
580 DATA ED,B0        :'LDIR
590 DATA FB           :'EI
600 DATA C9,"**"      :'RET
610 :
620 RESTORE 520:T=&HE240
630 READ A$:IF A$="**" THEN 640 ELSE   P
OKE T,VAL("&H"+A$):T=T+1:GOTO630
640 DEFUSR1=&HE240
```

Variabele BAUDsnelheden

Cassettegebruikers zullen het zeker op prijs stellen dat MSX computers het mogelijk maken om met twee verschillende snelheden programma's weg te schrijven. Er blijkt echter in de praktijk de behoefte te bestaan naar nog meer BAUD-snelheden.
MSX computers bewaren de gegevens omtrent BAUD-snelheden in het systeem RAM. Door in deze locaties andere waarden te "POKEN" kunnen we nu de BAUD-snelheid naar behoefte veranderen. Uw MSX computer 'ziet' later zelf weer met welke snelheid er moet worden ingelezen.

We hebben de volgende snelheden voor u geselecteerd:

- 600 BAUD (langzaam)
- 1200 BAUD (standaard)
- 1800 BAUD (gulden middenweg)
- 2400 BAUD (standaard snel)
- 5000 BAUD (zeer snel)

Als laatste opmerking dient nog wel te worden vermeld, dat 5000 BAUD niet kan worden verwerkt door een data-recorder. Hiervoor dient u uw MSX computer aan te sluiten op een hoogwaardig cassettedeck.

Veel plezier met het gebruik van deze routine.

```
10 REM ****************************
20 REM *                          *
30 REM * VERSCHILLENDE SNELHEDEN *
40 REM * WEGSCHRIJVEN VAN CASS-   *
50 REM * ETTE PROGRAMMA'S         *
60 REM *                          *
```

```
70 REM ****************************
80 :
100 COLOR 15,1:SCREEN0:WIDTH40:KEYOFF
110 LOCATE 5,2:PRINT" *  Baudsnelheden
*
120 LOCATE 9,5:PRINT"1.  600 Baud
130 LOCATE 9,6:PRINT"2. 1200 Baud
140 LOCATE 9,7:PRINT"3. 1800 Baud
150 LOCATE 9,8:PRINT"4. 2400 Baud
160 LOCATE 9,9:PRINT"5. 5000 Baud
170 :
180 PRINT:PRINT:PRINT"Welke snelheid (1-
6)?
190 A$=INPUT$(1):B=VAL(A$)
200 IF B<1 OR B>6 THEN 190
210 ON B GOSUB 230,340,450,560,670
220 END
230 :
240 REM **** 600 BAUD CASSETTE ****
250 :
260 POKE &HF406,INT(83*2.4)
270 POKE &HF407,92*2!
280 POKE &HF408,INT(38*2.7)
290 POKE &HF409,45*2!
300 :
310 POKE &HF40A,&HA
320 :
330 RETURN
340 REM **** 1200 baud cassette ****
350 :
360 POKE &HF406,&H53
370 POKE &HF407,&H5C
380 POKE &HF408,&H26
```

```
390 POKE &HF409,&H2D
400 :
410 POKE &HF40A,&HF
420 :
430 RETURN
440 :
450 REM **** 1800 baud cassette ****
460 :
470 POKE &HF406,&H3C
480 POKE &HF407,&H44
490 POKE &HF408,&H1A
500 POKE &HF409,&H21
510 :
520 POKE &HF40A,&H17
530 :
540 RETURN
550 :
560 REM **** 2400 baud cassette ****
570 :
580 POKE &HF406,&H25
590 POKE &HF407,&H2D
600 POKE &HF408,&HE
610 POKE &HF409,&H16
620 :
630 POKE &HF40A,&H1F
640 :
650 RETURN
660 :
670 REM **** 5000 baud cassette ****
680 :
690 POKE &HF406,&H10 :'low  1st half
700 POKE &HF407,&H16 :'low  2nd half
710 POKE &HF408,&H5  :'hi   1st half
```

```
720 POKE &HF409,&HA  :'hi   2nd half
730 :
740 POKE &HF40A,&H40 :'header cycle
                        count
750 :
760 RETURN
```

STOP OFF

Vindt u het ook niet vervelend dat sommige programma's niet te "breaken" zijn omdat er gebruik gemaakt wordt van "ON STOP GOSUB NR" en "STOP ON". Het nu volgende programma biedt een oplossing voor dit probleem.
Nadat onderstaand programma gelopen heeft, kunt u een op bovenstaande manier beveiligd programma eindelijk "breaken". De routine schakelt namelijk de "STOP ON" uit.
Hoe dient u te handelen?
Laad en RUN het onderstaande programma. De ESC-toets is nu geactiveerd. Nu kunt u in elk programma komen dat een "STOP ON" instruktie bevat. Als u zo'n programma heeft ingeladen, dan kunt u door het indrukken van de ESC-toets de "STOP ON" instruktie uitschakelen. Het programma is nu te "breaken" met CTRL STOP.

Verklaring van de werking

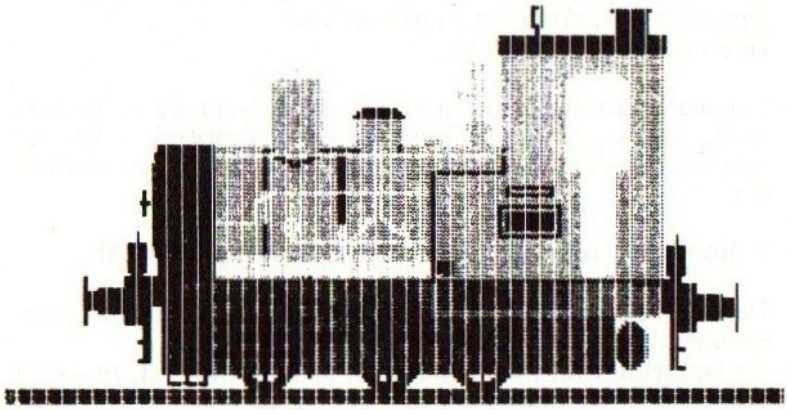
Zoals u waarschijnlijk weet staan er in het RAM-systeem variabelen die door de BASIC gebruikt worden. Een van deze variabelen is de STOP TRAP die zich op geheugen plaatsen FC6AH, FC6BH en FC6CH bevindt.
Als u onder BASIC de instrukties "ON STOP GOSUB 100: STOP ON" geeft, dan worden bovenstaande adressen als volgt geladen:
Geheugenplaats FC6AH wordt met een 1 geladen. Dit houdt in dat de "STOP ON"-funktie ingeschakeld wordt. Dit geheugenadres bevat een 0 als de "STOP ON"-funktie uitgeschakeld is. Adressen FC6BH en FC6CH bevatten het regelnummer waar naartoe gesprongen moet worden. In dit geval 100.
Onderstaande routine zorgt er nu voor dat adres FC6AH met een 0 geladen wordt als op de ESC-toets gedrukt wordt. Dit betekent dat de "STOP ON"-funktie is uitgeschakeld.

Het programma is dus te "breaken" met CTRL STOP.

```
10 '*********************************
20 '*                               *
30 '* Dit programma schakelt de     *
40 '* 'on stop gosub regel':'stop   *
50 '*  on' uit na het indrukken     *
60 '*  de ESC toets.                *
70 '*                               *
80 '*********************************
90 :
100 DATA FE,3A    :REM CP 3A
110 DATA 28,02    :REM JR Z,02
120 DATA C9       :REM RET
130 DATA f5       :REM PUSH AF
140 DATA 3E,00    :REM ld a,0
150 DATA 32,6A,FC:REM LD (FC6A),A
160 DATA f1       :REM POP AF
170 DATA C9       :REM RET
180 :
190 T=&HFFF2
200 FOR X=0 TO 12:READA$:POKE T+X,VAL("&
H"+A$):NEXT
210 :
220 REM *
230 REM * INITIALISEREN V/D HOOK
240 REM *
250 :
260 POKE&HFDCD,&HF2:POKE&HFDCE,&HFF
270 POKE&HFDCC,&HC3
280 :
290 STOP ON :ONSTOP GOSUB 340
300 CLS
```

```
310 PRINT"U kunt dit programma onmogelij
k"
320 PRINT"breaken alvorens u op de ESC t
oets":PRINT"hebt gedrukt.
330 GOTO 330
340 RETURN
```

Machinetaal
voor gevorderden



De laatste drie programma's van dit boek zijn bestemd voor hen die een redelijke kennis van machinetaal bezitten.
Er komen in deze programma's twee zeer handige ROM-routines voor. Hierbij dient wel vermeld te worden dat het geen standaard (BIOS) routines zijn. De routines zijn daarom ook niet volledig MSX compatible.

De eerste routine heeft betrekking op het printen van een string. Ze bevindt zich op adres 4A24H.
De routine werkt als volgt:

1. Laad registerpaar HL met het adres waar de te printen string opgeslagen ligt. (De string dient geopend te worden met aanhalingstekens en dient afgesloten te worden met een 0.)

2. Roep de routine aan door middel van CALL 4A24H.

De tweede routine stelt u in staat om Basic instrukties te gebruiken in een machinetaalprogramma.
De bedoelde routine bevindt zich op adres 4646H, ze wordt ook wel met de 'statement handler' aangeduid.
Eigenlijk is dit één van de belangrijkste routines die zich in het ROM bevinden.
De routine werkt als volgt:

1. Laad registerpaar HL met het beginadres van de uit te voeren instruktie. (De instruktie dient in gecodeerde vorm (tokenvorm) opgeslagen te worden.)

2. Roep de ROM-routine aan.

Voorbeelden die van bovenstaande routines gebruik maken, kunt u in de volgende programma's vinden: Circle in machinetaal en ML string printen.

Vanuit machinetaal naar BASIC

Voor de machinetaalprogrammeur is het belangrijkt om te weten hoe men dient over te gaan naar de direkte mode.

Allereerst dient de ROM (MSX-2 - de main- & sub-ROM) aangeschakeld te worden. Onderstaande routine geeft de slots aan waarin de ROM zich bevindt. Bij MSX-2 wordt tevens gekeken waar de sub-ROM zich bevindt.

Na het inschakelen van de ROM wordt er door middel van een "Jump" naar adres 409BH naar de direkte mode teruggesprongen. In het programma staat de juiste notatiewijze aangegeven.

```
10 REM ****************************
20 REM *                          *
30 REM * Terug naar BASIC van-    *
40 REM * uit een machinetaal      *
50 REM * programma.               *
60 REM *                          *
70 REM ****************************
80 :
90  REM ****************************
100 REM *                          *
110 REM * U dient allereerst de    *
120 REM * ROM in te schakelen.     *
130 REM *                          *
140 REM ****************************
150 :
160 REM          *  MSX-I    *
170 :
180 A=PEEK(&HFCC1)
```

```
190 PRINT"De hoofd-ROM bevindt zich in s
lot";A
200 :
210 REM         *  MSX-II  *
220 :
230 A2=PEEK(&HFAF8)
240 IF A2<>0 THEN PRINT"De sub-ROM bevin
dt zich in slot";A2
250 :
260 END
270 REM ************************
280 REM *                      *
290 REM * Terugkeren naar BASIC *
300 REM *                      *
310 REM ************************
320 :
330 C3,9B,40     ':JP 409BH
```

ML-string printen

Voor het printen van een enkel karakter bestaat er een ROM-routine. Deze routine vraagt om de ASCII-waarde van het te printer karakter in het A-register. Om een string te printen bestaat echter geen echte BIOSCALL.

In het BASIC-ROM bevindt zich bij de MSX-1 reeks van computers op locatie 4A24H een routine die een string print. De te printen string dient in het geheugen te worden "gepoked", zoals aangegeven in het onderstaande programma. Hierna dient het HL-register te worden geladen met het geheugenadres dat naar de string verwijst. Door middel van een CALL 4A24H wordt dan de string geprint.

```
1 REM ****************************
2 REM *                          *
5 REM * Tekst definiëren in      *
6 REM * regel 120                *
7 REM *                          *
8 REM ****************************
9 :
100 CLEAR 200,&HE200
110 T=&HE200
120 A$="Truuks & Tips deel 4
130 POKET,34
140 FOR X=1TOLEN(A$):M$=MID$(A$,X,1):POK
E X+T,ASC(M$):NEXT:POKEX+T,0
150 :
160 REM ****  DE ROUTINE  ****
170 :
180 DATA 21,00,E2 :'LD HL,E200H
190 DATA CD,24,4A :'CALL 4A24H
```

```
200 DATA C3,9B,40 :'JP 409BH
210 :
220 T=&HE220
230 FOR X=0 TO 8:READA$:POKE T+X,VAL("&H
"+A$):NEXT
```

Circle in machinetaal

(een truukje voor de machinetaal programmeur)

Het gebruik van machinetaal vereist veel kennis van uw MSX-computer. Het is dan ook niet zo eenvoudig om in machinetaal een cirkel te tekenen.

Als het nu echter toch nodig is om in een machinetaalprogramma een cirkel te tekenen, dan kunt u gebruik maken van de BASIC-ROM.

Onderstaande programma's laten achtereenvolgens zien hoe een cirkel kan worden getekend en hoe een PEEK-opdracht kan worden uitgevoerd.

N.B. De snelheid van deze routines is precies gelijk aan de BASIC equivalenten.

```
Hisoft GEN Assembler. Page      1.

Pass 1 errors: 00

D000              10          ORG   #D000
D000  2108D0      20          LD    HL,BASIC
D003  7E          30          LD    A,(HL)
D004  CD4646      45          CALL  #4646
D007  C9          50          RET
D008  BC          60 BASIC:   DEFB  188                ;STATEMENT HANDLER
D009  28313238    90          DEFM  "(128,96),60,15"  ;RETURN TO BASIC
                                                      ;CIRCLE

Pass 2 errors: 00

Table used:     27  from    118


Hisoft GEN Assembler. Page 1.

Pass 1 errors: 00

D000              10          ORG   #D000
D000  210AD0      20          LD    HL,BASIC
D003  7E          30          LD    A,(HL)
```

```
D004  CD4646     40          CALL #4646        ;STATEMENT HANDLER
D007  CD9B40     50          CALL #409B        ;RETURN TO BASIC
D00A  91         60 BASIC:   DEFB #91          ;PRINT
D00B  FF97       90          DEFB #FF,#97      ;PEEK
D00D  28264846  100          DEFM "(&HFFFF)"

Pass 2 errors: 00

Table used:    27  from   120
```

Nederlandstalige MSX handboeken

MSX BASIC handboek voor iedereen, door A.C.J. Groeneveld
Een kompleet nederlandstalig handboek voor iedere MSX computergebruiker. Dit handboek omvat een volledige behandeling van het MSX-basic in het Nederlands. Het handboek geeft een antwoord op elke vraag die een programmeur, van welke scholing ook, over het MSX-basic zou kunnen stellen. De volledige syntaxisbehandeling rekent af met onzekerheden of een bepaalde schrijfwijze nu wel of niet is toegestaan. De duidelijke beschrijving geeft per sleutelwoord aan, welke de funktie hiervan is. De laatste mogelijk nog aanwezig onduidelijkheden worden vervolgens door de opgenomen, zinvolle voorbeelden weggenomen
ISBN 90 6398 1007

MSX ZAKBOEKJE door Wessel Akkermans
Een vlot geschreven naslagwerk na of naast het handboek. U vindt er o.a. in: niet computergerichte tabellen; de MSX-BASIC instructieset; diverse tabellen die het BASIC-programmeren kunnen versnellen; de Z80 instructieset; hardware-gegevens (connectoren) en een aantal programmaatjes
ISBN 90 6398 888 5

MSX DISK handboek voor iedereen, door A.C.J. Groeneveld
Handboek voor diskdrivebezitters om naast het grote handboek te gebruiken. Een zeer volledige behandeling van het disk-gebeuren zelf en de specifieke disk kommando's, uitgebreid met voorbeelden, tabellen en overzichten. Het handboek is aangevuld met interessante programma's, waaronder een tekentafelprogramma en een basisprogramma voor basisonderhoud
ISBN 90 6398 407 3

MSX PRAKTIJKPROGRAMMA'S door Wessel Akkermans
Praktische programma's met waar nodig eerst een stukje theorie. Erg handig bij het maken van uw programma's. Een greep uit de onderwerpen: priemgetallen; zoeken en sorteren; trefwoordenlijsten; converteren van getallen; enz.
ISBN 90 6398 437 5

MSX QUICK DISK handboek voor iedereen, door A.C.J. Groeneveld
Hèt handboek voor iedere QUICK DISK gebruiker. Uitvoerige behandeling van de sleutelwoorden aangevuld met duidelijke voorbeelden met listing
ISBN 90 6398 254 2

MSX DOS handboek voor iedereen, door A.C.J. Groeneveld
Dit handboek geeft u op een heldere wijze een totaalbeeld van de mogelijkheden van het MSX-DOS. Ook is dit handboek voorzien van een inleiding op het begrip 'operating system' en dus echt een handboek voor iedereen
ISBN 90 6398 674 2

MSX LEERBOEKEN
door Wessel Akkermans en Piet den Heijer
De serie MSX leerboeken geeft een komplete kursus MSX-BASIC programmeren, in drie delen. Deze leerboeken zijn gericht op de beginnende programmeur. De moeilijkheidsgraad van de leerstof wordt dan ook slechts geleidelijk hoger. De gebruikte voorbeelden zijn zo praktisch mogelijk gekozen. Hierdoor kunnen al in een vroeg stadium bruikbare programma's worden gemaakt. Dit zal de lezer/leerling er toe aansporen om verder te gaan. Aan het eind van ieder deel is een groot voorbeeldprogramma opgenomen. Dit programma laat zien waartoe de lezer/leerling na bestudering van het betreffende leerboek in staat zal zijn.
Bij ieder leerboek is een afzonderlijk –Opdrachten en uitwerkingen– boekje te verkrijgen. In deze boekjes staan, in volgorde van de hoofdstukken uit het leerboek, vragen en opdrachten met antwoorden en uitwerkingen. Een unieke serie leerboeken voor een ieder die meer over MSX wil weten en het betere werk met zijn computer wil maken.

MSX Basic leerboek deel 1 - ISBN 90 6398 649 1
Opdrachten bij deel 1 - ISBN 90 6398 596 7
MSX Basic leerboek deel 2 - ISBN 90 6398 769 2
Opdrachten bij deel 2 - ISBN 90 6398 556 8
MSX DOS leerboek deel 3 - ISBN 90 6398 519 3
Opdrachten bij deel 3 - ISBN 90 6398 516 9

MSX Verder uitgediept door H. Klopper
Eindelijk een Nederlandstalig boek over het altijd in de mist gehulde onderwerp – PEEKS EN POKES. In dit boek staan alle belangrijke RAM en VRAM adressen. De video chip en zijn registers worden volledig uitgelegd. Maar ook hoe men een machinetaal programma van cassette naar disk kan schrijven. Bovendien een diskloader utility en een uiterst geavanceerde programma beveiliging. Tenslotte zijn er een aantal interessante programma's opgenomen, waaronder een wereldkaart, waarmee verder kan worden geëxperimenteerd. Elke MSX gebruiker kan in dit boek iets van zijn gading vinden en nieuws leren.

ISBN 90 6398 447 2

MSX Machinetaal handboek door H. Klopper en M. Le Belle
Hoewel een MSX computer over een krachtig Basic beschikt, is het toch handig tijdens het programmeren de grondbeginselen van machinetaal te kennen. Daarvoor is dit boek een goede gids. De zaken worden niet puur theoretisch maar ook aan de hand van duidelijke voorbeelden, die direkt bruikbaar zijn, uitvoerig uitgelegd. Enkele onderwerpen zijn verder – scroll routine –machinetaal software (ook in disk Basic) op casette zetten –disassembler –Z80 essembler instrukties –lijst van ROM-routines –alle hook-adressen –bespreking van Basic tokens en een kompleet token-overzicht. Hèt handboek voor iedere MSX programmeur die zijn computer ten volle wil benutten.

ISBN 90 6398 735 8

MSX TRUUKS EN TIPS
door A.C.J. Groeneveld
Hoe laat ik de computer een cirkel arceren, hoe tover ik mijn computer om in een elektronisch orgeltje, hoe maak ik een mooie intro voor een spelletje. Allemaal vraagstukken die zich lastig laten programmeren maar die iedere MSX-er toch graag opgelost wil zien.
Dit boekje staat boordevol truuks en tips, allemaal in gewoon MSX basic geschreven. Bladerend door dit boek komt u tot de ontdekking dat er voornamelijk korte maar uiterst krachtige en bijzonder goed bruikbare routines zijn opgenomen. Dit boekje geeft kort maar krachtig een antwoord op al uw programmeervragen.

deel 1 ISBN 90 6398 900 8
deel 2 ISBN 90 6398 340 9
deel 3 ISBN 90 6398 910 5

SOFTWARE PLUS IN MSX

INTROTAPE MSX door A.C.J. Groeneveld
Heeft u nog maar net een MSX computer gekocht en wilt u graag weten wat de computer kan en hoe u hem kunt leren programmeren? Deze cassette introduceert MSX op een uiterst vriendelijke en onderwijzende manier. U krijgt instructies hoe u de computer aan moet sluiten en de tape laden. Daarna volgt een demonstratie van de mogelijkheden in MSX, zoals het tekenen van sprites en het werken met de driestemmige toongenerator. Het geheel wordt afgesloten met twee 'les' gedeeltes. In anderhalf à drie uur weet u wat de MSX computer is, wat hij kan, en heeft u haast ongemerkt al wat regels geprogrammeerd.

ISBN 90 6398 148 1

MSX SCRIPT door Ton Weijters
Een menugestuurde nederlandstalige tekstverwerker. Het programma is geschikt om efficiënt grotere of kleinere teksten te bewerken. Paginaindeling (regellengte, paginalengte, marge, inspringen, centreren, enz.) wordt door het programma verzorgd. Dit geldt ook voor de paginatelling, toptitel en het eventueel uitvullen van de regels. Ook corrigeren, zoeken, string-substitutie, blokken tekst verplaatsen, copiëren of verwijderen, onderstrepen en vet zetten, is mogelijk met dit programma.

ISBN 90 6398 189 9

MSX DRAWS door A.C.J. Groeneveld
Een tekenprogramma in MSX basic, waarmee u al binnen 10 minuten uw eerste tekening kunt maken. Draws werkt erg vriendelijk en maakt gebruik van alle grafische mogelijkheden van de MSX computer. U kunt met Draws zowel technisch als creatieve tekeningen maken. Het programma heeft een effectief bereik van ruim 30.000 bij 30.000 puntjes met mogelijkheden als lijnen, cirkels, krommen, inkleuren, vergroten, verkleinen, verschuiven, verdraaien en andere tekeningen invoegen

ISBN 90 6398 754 4

MSX2 Basic handboek
A.C.J. Groeneveld, ISBN 90 6398 221 6, 507 pagina's, prijs f 56,50
Wie denkt over de aanschaf van een MSX2 computer, of er al een heeft, kan niet zonder het 507 pagina's tellende MSX2 Basic handboek. Alles over MSX2 Basic, de grafische- en geluidsmogelijkheden en de computer zelf. Met 288 voorbeeldprogramma's.

MSX2 Disk/Dos Uitbreidingshandboek
A.C.J. Groeneveld, ISBN 90 6398 222 4, 172 pagina's, prijs f 37,50
Omvat een volledige behandeling van het MSX2 Disk Basic en het MSX DOS operating system, voorafgegaan door een zeer duidelijke inleiding tot de fenomenen disk en operating system. Verder praktische tabellen, duidelijke afbeeldingen en zinvolle voorbeelden. Een standaardwerk dat naast elke MSX2 Disk computer zou moeten liggen.

MSX2 Utility-Toepassingshandboek
A.C.J. Groeneveld, ISBN 90 6398 223 2, 144 pagina's, prijs f 29,75
Een verzameling programma's die voor elke MSX-er onontbeerlijk zijn. Een aantal van de mogelijkheden met deze programma's: bestandsonderhoud met lijstwerk in iedere vorm op schijf en tape, staaf- en taartdiagrammen, programma's samenstellen met sprites en geluidseffekten, binair manipuleren binnen blokken op schijf. Alle programma's in dit boek zijn geschikt voor zowel MSX als MSX2 computers.

truuks en tips deel 4

Zoals bijna elke computer is de MSX-range van huiscomputers voorzien van een Basic-vertaler. De MSX-vertaler geeft de (hobby)programmeur de gelegenheid om door middel van zeer krachtige kommando's goede programma's te schrijven. De MSX truuks en tips reeks heeft als doel om u bij de ontwikkeling van deze programma's te ondersteunen.
In navolging van de eerste drie delen, kunt u in dit boekje weer de nodige nuttige routines en programma's vinden die u nergens anders zult aantreffen. Naast de truuks voor de Basic-programmeur, hebben we nu ook een hoofdstukje opgenomen met truuks en tips voor de MSX-machinetaalprogrammeur.
Een kleine selektie uit de inhoud:

- Foutmeldingen in het nederlands (assembler programma)
- "NEW/KILL" voorzorgsmaatregelen
- Vijfde sprite-detektieroutine
- Variabele kassettesnelheden (600 - 5000 baud)
- Langzame listing generator

Dit is slechts een zeer kleine greep uit de vele uitgekookte truuks en tips die ook dit deel weer bevat.
MSX truuks en tips deel 4 geeft u weer meer het gevoel de computer volledig te beheersen.

ISBN 90 6398 897 4

uitgeverij Stark-Texel