zakboekje MSX

MSX2, BASIC, DISK-BASIC, DOS



a. van utteren

kluwer technische boeken

A. van Utteren Zakboekje MSX

Zakboekje MSX

Aaldrik van Utteren

Zakboekje MSX

MSX2
BASIC
DISK BASIC
DOS



Kluwer Technische Boeken B.V.
Deventer - Antwerpen

Omslag: W. Niessink

ISBN 90 201 2002 6
D/1987/0108 180
NUGI 434

© 1987 Kluwer Technische Boeken B.V. Deventer

1e druk 1987

Niets uit deze uitgave mag worden verveelvoudigd en/of openbaar gemaakt door middel van druk, fotokopie, microfilm of op welke andere wijze ook, zonder voorafgaande schriftelijke toestemming van de uitgever.

No part of this book may be reproduced in any form, by print, photoprint, microfilm or any other means without written permission from the publisher.

Ondanks alle aan de samenstelling van de tekst bestede zorg, kan noch de redactie noch de uitgever aansprakelijkheid aanvaarden voor eventuele schade, die zou kunnen voortvloeien uit enige fout, die in deze uitgave zou kunnen voorkomen.

Woord vooraf

Inmiddels is de MSX-computer alom bekend en niet meer weg te denken uit het steeds groter wordende aanbod van home-computers. Met de komst van MSX2 werd het mogelijk videobeelden te digitaliseren, een stap dichter bij de integratie van de computer in het audio- en videogebeuren.
Steeds meer BASIC-commando's staken de kop op, te veel om te onthouden. Maar ook het gebruik van de verschillende commando's werd steeds ingewikkelder. Vandaar dit zakboekje, een hulpmiddel voor hen die veel programmeren, met een compleet overzicht van alle MSX-commando's.
Er is bewust een splitsing gemaakt tussen BASIC en Disk BASIC, vooral voor hen die nog steeds met de cassetterecorder werken. Voor degene die wel een diskdrive of MSX2 ter beschikking heeft, biedt het aparte hoofdstuk met alle Disk BASIC-commando's een makkelijk en snel toegankelijk overzicht.

Ook MSX-DOS, het operating system voor de professionele gebruiker, is niet vergeten. Ideaal voor veel nuttige toepassingsprogramma's, zoals tekstverwerkers, communicatieprogramma's e.d.

Ik hoop dat u veel gebruik maakt van dit MSX-zakboekje.

Blaricum, februari 1987 Aaldrik van Utteren

INHOUD

1 Hardware 8
1.1 MSX-specificaties 8
1.1.1 De Z-80A microprocessor 9
1.1.2 I-8255 periferal interface 9
1.1.3 AY-3-8910 audiochip 9
1.1.4 TMS-9918A videochip 9
1.2 Geheugenstructuur 10
1.2.1 Primaire slots 10
1.2.2 Secundaire slots 11
1.3 MSX-diskdrives 11
1.3.1 MSX-Disk BASIC 11
1.3.2 Diskettes 11
1.3.3 Formatteren 12
1.3.4 Single of Double Sided 12
1.3.5 Single of Double Density 12
1.3.6 Tracks 12
1.3.7 Sectoren 12
1.4 Disk Operating System 12
1.4.1 MSX-DOS 13
1.5 De klokchip 13
1.6 Memory-disk 13

2 BASIC-commando's 14
3 BASIC-functies 52
4 Disk BASIC-commando's 64
5 Disk BASIC-functies 72
6 DOS-commando's 76
7 BATCH-commando's 83

Appendix A Systeemvariabelen 85
Appendix B Speciale tekens en uitdrukkingen 89
Appendix C Control- en kleurcodes 91
Appendix D RS232C-interface 94
Appendix E Foutmeldingen 103

1 Hardware

1.1 MSX-specificaties

Het MSX-ontwerp heeft zowel op de hardware als op de software betrekking, waardoor het mogelijk is programma's en randapparaten tussen de verschillende merken uit te wisselen.
Er zijn minimale voorwaarden op hardware-gebied, waarbij iedere fabrikant eigen opties mag toevoegen. Alle MSX-computers voldoen aan de volgende (minimale) eisen:

- Z-80A, 8 bits-processor;
- TMS-9918A videochip (of overeenkomstig type);
- 16K RAM-videogeheugen;
- AY-3-8910 audiochip (of overeenkomstig type);
- I-8255 periferal interfacechip (of overeenkomstig type);
- 32K ROM-geheugen, dat de MSX BASIC-interpreter bevat;
- minimaal 8K RAM-geheugen;
- standaard-karakterset (afhankelijk van het land);
- standaard-toetsenbord (afhankelijk van het land);
- interface voor:
  cassetterecorder;
  minimaal één joystick;
  minimaal één uitbreidingsslot.

Deze minimale eisen worden door de meeste fabrikanten uitgebreid met bijvoorbeeld een printerinterface, tweede joystick-aansluiting, tweede uitbreidingsslot, muziek-synthesizer, lichtpenaansluiting en/of geheugenuitbreiding tot 16K, 32K of 64K.
Met behulp van het uitbreidingsslot kan de computer worden uitgebreid met onder andere een RS-232C interface, diskette-interface, 64K RAM-insteekmodule of programma-modules. Mocht het aantal aansluitingen van de computer te klein zijn dan kunnen deze via het uitbreidingsslot worden uitgebreid. Via de joystickpoort kan ook een 'graphic tablet', een 'touch-pad' of een paddle worden aangesloten.
Wat de software betreft dient iedere MSX-computer natuurlijk dezelfde MSX BASIC-instructieset te bevatten. BASIC-program-

ma's kunnen dus zonder problemen worden uitgewisseld. Eveneens kunnen machinetaalprogramma's worden uitgewisseld, mits deze programma's gebruik maken van de interpreter en niet rechtstreeks naar I/O-poorten lezen of schrijven! Bij programma-uitwisseling moet de gebruiker letten op de geheugenruimte die het programma inneemt.

1.1.1 De Z-80A microprocessor

Het hart van de MSX-computer wordt gevormd door de Z-80A microprocessor. Hoewel tegenwoordig de 16-bitter voorkeur geniet, heeft het gebruik van de Z-80A voordelen, onder andere een veel lagere kostprijs, verschil in verwerkingssnelheid en de grote hoeveelheid bestaande software. Een nadeel van de Z-80A is het feit dat hij slechts 64K geheugen kan adresseren. Meer dan 64K geheugen betekent dat 'bank switching' moet worden toegepast.

1.1.2 I-8255 periferal interface

Deze chip heeft drie poorten (A, B en C) van ieder acht bits. Poort A wordt bij MSX gebruikt voor bank switching, terwijl poort B voor het return-signaal van het toetsenbord dient. Poort C wordt gebruikt voor het scan-signaal van het toetsenbord, cassette controle-signaal, cassette write-signaal en het CAPS-lampsignaal.
Maak bij machinetaalprogramma's nooit rechtstreeks gebruik van deze poorten, maar gebruik de sprongvectoren van de interpreter.

1.1.3 AY-3-8910 audiochip

De geluidschip heeft zestien registers van acht bits. De eerste veertien registers zijn voor het produceren van geluid, terwijl de laatste twee registers als poorten voor de cassetterecorder en de joysticks worden gebruikt. Het is met deze chip mogelijk om driestemmige muziek te maken, waarbij iedere stem acht octaven beslaat.
Ruis kan worden gegenereerd en gemengd met de drie stemmen. Voor elke stem kunnen het volume en de geluidsvorm afzonderlijk worden ingesteld.

1.1.4 TMS-9918A videochip

De videochip heeft negen registers van acht bits. Deze chip communiceert met het videogeheugen en kent vier verschillende schermindelingen:

Tekstmode 1:
maximaal 40 tekens per beeldschermregel.

Tekstmode 2:
32 tekens per beeldschermregel. Het verschil tussen mode 1 en 2 is de weergave van de grafische karakters die via het toetsenbord kunnen worden ingevoerd. Deze grafische tekens zijn opgebouwd uit 8 bij 8 beeldpunten, terwijl de ASCII-tekens (alfabet, cijfers en leestekens) uit een 6 bij 8 matrix zijn opgebouwd. In tekstmode 1 worden alle tekens op het beeldscherm weergegeven in een matrix van 6 bij 8 beeldpunten. De grafische tekens worden dan onvolledig weergegeven.

Grafische mode 1:
hierbij wordt het beeldscherm verdeeld in 256 bij 192 beeldpunten, de zogenoemde high resolution mode.

Grafische mode 2:
de lowresolution mode, waarbij het scherm wordt verdeeld in 64 bij 48 blokken, waarbij ieder blok 4 bij 4 beeldpunten groot is.

De videochip kan zestien kleuren genereren. In tekstmode 1 zijn er echter slechts twee beschikbaar, één voor de tekens en één voor de achtergrondkleur.

1.2 Geheugenstructuur

In de 32K ROM bevindt zich het MSX-interpreter-programma en kan de computer 16K, 32K of 64K RAM-geheugen bevatten. Via het uitbreidingsslot is het mogelijk om 64K geheugen of een programma in ROM toe te voegen. De Z-80A is met 'bank switching' in staat om deze extra 64K te adresseren, waarbij de 16K voor de video buiten beschouwing blijft. Deze is alleen door de videochip bereikbaar en niet door de processor. Voor MSX-DOS dient men echter over minimaal 64K te beschikken.

1.2.1 Primaire slots
Bij MSX kunnen vier 'banks' (slots) van elk 64K worden geadresseerd. Ieder slot is verdeeld in vier 'pages' van elk 16K.
De BASIC-interpreter bevindt zich dus in de eerste twee pages van het eerste slot (altijd in de MSX-computer). De overige drie slots kunnen óf in de MSX-computer óf in een uitbreidingsslot zijn ge-

plaatst. Elke page van ieder slot kan RAM, ROM of een interface bevatten, waarbij de eerste twee bytes in de pages aangeven wat die page bevat.
Voordat de Z-80A uit het geheugen gegevens leest of opslaat, wordt in het selectieregister gekeken in welk slot de desbetreffende page zich bevindt. Via het selectieregister kan men in machinetaal bank switching toepassen.
In BASIC laat de interpreter dit niet toe.

1.2.2 Secundaire slots

Ieder primair slot kan worden uitgebreid met nog eens vier slots voorzien van vier pages van elk 16K. Deze slots worden secundaire slots genoemd. In dit geval is het primaire slot niet meer bruikbaar; alleen het hoogste adres van het primaire slot wordt toegepast als slot-selectieregister voor het secundaire slot. In principe zijn dus zestien slots van elk vier pages (16K) mogelijk.
In de praktijk kunnen echter alleen de primaire slots 1 en/of 2 worden uitgebreid met secundaire slots. Immers bij de 64K computer zitten slot 0 (interpreter) en slot 3 (64K RAM) in de computer, terwijl slot 1 en/of slot 2 als uitbreidingsslot zijn gebruikt. Via dit slot kunnen secundaire slots worden aangesloten.

1.3 MSX-diskdrives

Aan iedere MSX-computer kunnen één tot vier diskdrives worden gekoppeld. De aansluiting gebeurt via het uitbreidingsslot. In dit slot wordt de interface gestoken; deze wordt via een kabel verbonden met de diskdrive. Een tweede diskdrive kan met een tweede kabel aan de eerste worden verbonden.

1.3.1 MSX-Disk BASIC

De diskdrive kan tijdens het programmeren in BASIC volledig worden gebruikt voor de opslag van programma's, sequentiële files en random-files. Het is zelfs mogelijk om ook het 16K videogeheugen naar diskette weg te schrijven en later terug te lezen. Voor MSX Disk BASIC is het niet nodig dat er 64K geheugen aanwezig is (in tegenstelling tot MSX-DOS).

1.3.2 Diskettes

De diskettes zijn tegenwoordig in drie maten verkrijgbaar, namelijk 8, 5,25 en 3,5 inch. Iedere maat vereist een diskdrive die op dat formaat is afgestemd.

1.3.3 Formatteren

De diskette moet, voordat zij voor een computer leesbaar is, worden geformatteerd. De diskette wordt dan zodanig ingedeeld dat de computer de schijf kan gebruiken. Het formatteren wordt in de meeste gevallen gedaan door een programma. MSX-computers hebben daar een specifiek commando voor.

1.3.4 Single of Double Sided

Tijdens het formatteren wordt vastgelegd of één zijde van de diskette wordt gebruikt (single sided) of beide zijden (double sided).

1.3.5 Single of Double Density

Tijdens het formatteren wordt ook bepaald met welke schrijfdichtheid de diskette wordt gebruikt: enkelvoudige dichtheid (single density) of dubbele dichtheid (double density).

1.3.6 Tracks

Op de diskette wordt tevens vastgelegd uit hoeveel sporen (tracks) de diskette wordt opgebouwd, 40 of 80 per diskette.

1.3.7 Sectoren

Iedere track wordt weer onderverdeeld in blokken (sectoren), 8 of 9. Elke sector kan afhankelijk van de formattering een aantal tekens bevatten variërend van 128, 256 tot 512 tekens.

1.4 Disk Operating System

Om gegevens naar een diskette te schrijven of van een diskette te lezen, moeten er een aantal handelingen worden verricht: er moet worden bepaald waar de gegevens staan (sector, track), de inhoudsopgave (directory) van de diskette moet worden bijgehouden enz. Deze handelingen worden uitgevoerd door een disk operating system. Een operating system is altijd voor een bepaalde processor geschreven, maar is niet afhankelijk van de overige hardware. Ieder operating system heeft zijn eigen wijze van schrijven en lezen en ook de plaats en de indeling van de directory zijn per operating system verschillend.
Het MSX-ontwerp houdt rekening met het feit dat er eventueel met een disk operating system kan worden gewerkt. Voorwaarde is dan dat de machine 64K geheugen moet bevatten.
Het is niet gelukt overeenstemming over de te gebruiken diskettes te bereiken met de MSX-computerfabrikanten. Zo is het gebruik

van zowel 8, 5,25, 3,5 als 3 inch diskettes toegestaan, zodat uitwisseling van programma's op diskettes moeilijkheden kan geven.

1.4.1 MSX-DOS

Microsoft ontwikkelde speciaal voor de MSX-computers een nieuw disk operating system: MSX-DOS.

1.5 De klokchip

De MSX-computers uit de eerste generatie waren niet allemaal voorzien van een interne klok. Met de komst van MSX2 is er een standaard-klokchip opgenomen die niet alleen de tijd en datum kan vasthouden, maar bijvoorbeeld ook een 'wachtwoord'.
Ook als de computer wordt uitgezet, zal de chip alle opgeslagen gegevens bewaren en bij het aanzetten zijn die gegevens direct beschikbaar. Behalve het wachtwoord kan men ook de 'prompt' veranderen. Dit is in BASIC het woord 'Ok', dat telkens op uw beeldscherm komt na het voltooien van een programma of commando. Er is ook ruimte voor een eigen titel die verschijnt bij het opstarten van de computer. Een speciaal commando zorgt ervoor dat u het gehele weergavescherm kunt verplaatsen, bijvoorbeeld een stukje naar rechts als de eerste kolommen niet zichtbaar zijn op uw beeldscherm.
Ook het geluid dat wordt geproduceerd door het commando BEEP kunt u vooraf instellen en opslaan in de klokchip. Normaal start de computer op met het tekstscherm, als u een andere standaardinstelling wenst te gebruiken, kunt u deze opslaan in de chip.

1.6 Memory-disk

Bij MSX2 wordt niet het gehele geheugen gebruikt voor BASIC en voor de programma's. De overgebleven geheugenruimte kan men benutten als een extra opslagmedium, een zogenoemde 'memory-disk'.
Daar een memory-disk veel sneller werkt dan een normale diskdrive biedt dit veel voordelen bij o.a. het sorteren, tijdelijk opslaan van gegevens die snel beschikbaar moeten zijn e.d.
Voor men deze vrije geheugenruimte als memory-disk wil gebruiken, dient men deze als het ware te 'formatteren' met een speciaal commando, 'CALL MEMINI'. Voor meer informatie hierover verwijzen we u naar het hoofdstuk over BASIC-commando's.

2 BASIC-commando's

In dit hoofdstuk treft u een overzicht aan van alle MSX-BASIC commando's. De naam van het commando vindt u in de marge; in voorkomende gevallen is de syntax ervan vetgedrukt zodat u in één oogopslag kunt zien hoe u het commando dient te gebruiken. Verder vindt u bij de meeste commando's een voorbeeld.

AUTO

AUTO [<regelnummers>],[,[stapgrootte]]
Dit commando genereert automatisch regelnummers om het invoeren van een programma te vereenvoudigen. Wordt er geen regelnummer opgegeven, dan zal het nummeren beginnen met regelnummer 10. Na het afsluiten van de regel met een <RETURN> zal de eerstvolgende regel met 10 worden verhoogd.

```
AUTO 100,20
```
Nu begint het automatisch nummeren bij regel 100 en zullen de volgende regels steeds met 20 toenemen. Als een reeds aanwezig regelnummer wordt gegenereerd, ziet men een asterisk als waarschuwing achter het regelnummer. Het commando wordt met <CONTROL>-C of <CONTROL>-STOP beëindigd.

BEEP

BEEP
Geeft een korte (0,04 seconde) pieptoon, komt overeen met de opdracht PRINT CHR$(7).

BLOAD

BLOAD "<dev>:[<filenaam>]"[,R][,<verschuiving>]
Laadt een machinetaalprogramma dat met het commando BSAVE is opgeslagen.

BLOAD "CAS:PROG.ASM",R
Geeft men een R op achter de naam van het programma dan wordt het programma na het laden onmiddellijk gestart.

BLOAD "CAS:PROG.ASM",&H00FF
Geeft men een adres op achter de naam van het programma dan wordt het programma vanaf dit adres in het geheugen van de computer geplaatst.

BSAVE **BSAVE "<dev>:[<filenaam>]",<beginadres,eindadres>[,<beginadres voor start van het programma>]**
Hiermee wordt een in machinetaal geschreven programma opgeslagen op een cassette (CAS:) of op een diskette (A:).

CALL **CALL <naam>[(<string>,<string> ...)]**
Commando om bepaalde subroutines die in de ROM-cartridge zijn opgeslagen, uit te voeren. Achter CALL dient men een naam op te geven, achter de naam van de CALL kan men variabelen opnemen. Bij MSX2 zijn er een aantal CALL's opgenomen voor het instellen van de memory-disk.

CIRCLE Dit commando werkt alleen in de grafische mode, SCREEN 2 of 3.

CIRCLE (X,Y),R,C
Tekent een cirkel op het beeldscherm. Het middelpunt wordt opgegeven met een X- en Y-coördinaat terwijl R de straal aangeeft. De C geeft de kleur aan; indien men hier geen waarde opgeeft, wordt de cirkel getekend in de op dat moment ingestelde voorgrondkleur.

CIRCLE (X,Y),R,C,BH,EH
Men kan een gedeelte van een cirkel tekenen door een begin- (BH) en een eindhoek (EH) op te geven (in radialen). Achter de straal komt óf het kleurnummer óf een extra komma als men de kleur ongewijzigd wil laten.

CLEAR **CLEAR [<geheugenruimte voor strings>][,<hoogste adres>]]**
CLEAR 'reset' alle variabelen en reserveert ruimte voor stringvariabelen. CLEAR heeft tevens tot gevolg dat alle eventueel nog geopende bestanden worden gesloten.
Zonder CLEAR reserveert de computer slechts 200 bytes voor het opslaan van karakters.

```
CLEAR 500,&HF300
```
Het adres specificeert de hoogste geheugenlocatie die in BASIC kan worden gebruikt. In dit voorbeeld wordt de geheugenruimte tussen &HF300 en &HF380 vrijgehouden voor door uzelf geschreven machinetaalroutines.

CLOAD **CLOAD ["<filenaam>"]**
Commando om een programma van cassette te laden. Men dient nu niet meer de term 'CAS:' op te nemen.

```
CLOAD"PROG1"
```
Heeft tot gevolg dat een programma met de naam PROG1 van cassette wordt geladen. Als er iets mis gaat, dient men het laden met CTRL/STOP te onderbreken, de band terug te spoelen en het laden opnieuw te starten.
Laat men de programmanaam weg dan zal het eerste programma dat op de band wordt aangetroffen, worden gelezen.

CLOAD? ["<filenaam>"]
Commando om een ingelezen programma te vergelijken met het programma op band. Komen de gegevens overeen dan verschijnt er 'Ok'. Bij een fout verschijnt 'Verify error'.
Men dient de band natuurlijk eerst terug te spoelen.

CLOSE **CLOSE [#][<filenummer>][,[#]<filenummer> ...]**
Sluit het bestand dat overeenkomt met het opgegeven bestandsnummer. Geeft men geen nummer op dan worden alle bestanden gesloten. Men mag achter het

eerste filenummer meer nummers opgeven, gescheiden door een komma.

CLS **CLS**

Maakt het beeldscherm schoon. Indien de functie 'KEY ON' is ingesteld, blijft de functie-omschrijving van de functietoetsen onderin het beeld zichtbaar.

COLOR **COLOR [<voorgrond>][,<achtergrond>][,<randgebied>]**

Hiermee kan men de weergavekleur van het beeldscherm bepalen. De kleur geeft men steeds met een cijfer aan, de intensiteit geeft aan uit welke basiskleuren de kleur is samengesteld.

Cijfer	*Kleur*	*Intensiteit (RGB)*
0	transparant	0 0 0
1	zwart	0 0 0
2	groen	1 6 1
3	lichtgroen	3 7 3
4	donkerblauw	1 1 7
5	lichtblauw	2 3 7
6	donkerrood	5 1 1
7	cyaan	2 6 7
8	rood	7 1 1
9	lichtrood	7 3 3
10	donkergeel	6 6 1
11	lichtgeel	6 6 4
12	donkergroen	1 4 1
13	magenta	6 2 5
14	grijs	5 5 5
15	wit	7 7 7

De standaard-instelling komt overeen met COLOR 15,4,7, witte voorgrond, donkerblauwe achtergrond en een cyaankleurige rand. De volgorde van de kleuren achter COLOR is voorgrond, achtergrond en als laatste de rand.

In SCREEN 6 (MSX2) kan men de rand weergeven met een gestreept patroon; dit gaat met de volgende code:

Nummer	*Menging van kleurnummer*
16	0 en 0
17	0 en 1
18	0 en 2
19	0 en 3
20	1 en 0
21	1 en 1
22	1 en 2
23	1 en 3
24	2 en 0
25	2 en 1
26	2 en 2
27	2 en 3
28	3 en 0
29	3 en 1
30	3 en 2
31	3 en 3

In SCREEN 8 (MSX2) wordt de kleur bepaald door een waarde tussen 0 en 255 en wel volgens de formule:

kleurnummer = 4 * R + 32 * G + B

De letters R, G en B staan voor de intensiteit van respectievelijk de kleuren rood, groen en blauw. Deze intensiteit mag voor rood en groen liggen tussen 0 en 7. De intensiteit van blauw moet tussen 0 en 3 liggen.

COLOR 5
Alleen de voorgrond wordt lichtblauw. Een nieuw opgegeven achtergrondkleur verandert niet eerder dan na het 'CLS'-commando.

COLOR= **COLOR=(<kleurnummer>,<intensiteit>)**
Bij MSX2 kan men de standaard-kleurnummers qua intensiteit wijzigen om daarmee eigen kleuren te kunnen gebruiken.

```
COLOR=(2,6,1,6)
```
Kleur '1' is nu niet meer groen maar paars-achtig.

COLOR=NEW
Hiermee komen de standaard-intensiteitinstellingen terug na het veranderen van één of meer kleuren.

COLOR=RESTORE
Hiermee worden de in het videogeheugen opgeslagen intensiteiten ingesteld. Men kan een zelf te definiëren intensiteit voor alle kleuren en screenmodes op diskette vastleggen en later weer laden met het commando BLOAD ...,S. Zie ook bij 'Systeemvariabelen' voor de geheugenindeling.

COLOR SPRITE

COLOR SPRITE (<nummer>)=<kleur>
Met dit commando geven we de sprite (nummer) een eigen kleur, alleen mogelijk bij de screenmodes 4 t/m 8 (MSX2).
De kleur ligt tussen 0 en 15; verhogen we die waarde met 32 dan zal de instructie ON SPRITE GOSUB niet meer actief zijn. Verhogen we de kleur met 64 dan zal de prioriteit worden genegeerd en wordt er een logische OR-functie uitgevoerd om de kleur te bepalen.

COLOR SPRITE$

COLOR SPRITE$ (<kleur>)=<string>
Dit commando maakt het mogelijk om een deel van de sprite een eigen kleur te geven. Dit kan alleen bij de screenmodes 4 t/m 8 (MSX2). De kleur is een nummer tussen 0 en 15, verhogen we die waarde met 32 dan zal er niet meer worden gelet op botsende sprites. Verhogen we de waarde met 64 dan wordt de kleur bepaald door een logische OR-bewerking. Verhogen we de waarde met 128 dan zal de lijn van de sprite 32 pixels naar links schuiven.

CONT

CONT
Dit commando laat de uitvoering van een programma hervatten vanaf de plaats waar het werd onderbroken

met CTRL/STOP of met het STOP- of END-commando.

COPY TO

Commando om delen van het grafische scherm te kopiëren naar een file of array, alleen bij screenmodes 5 t/m 8 (MSX2). Eerst dient men de X- en Y-coördinaten van het beginpunt op te geven, daarna de eindcoördinaten. Hierna kan men de 'bron'-pagina opgeven.
Achter TO komen de X- en Y-coördinaten die het beginpunt aangeven van de kopie, daarachter de bestemmingspagina.
Achter TO kan men in plaats van coördinaten de naam van een array of file opnemen. Ook het kopiëren van een array naar een file en andersom is mogelijk.

COPY SCREEN

COPY SCREEN [<X>]
Dit commando (MSX2)dient om een videosignaal te digitaliseren. Als men de waarde 0 achter dit commando opneemt, wordt het aangeboden signaal gedigitaliseerd en direct weergegeven op het beeldscherm.
Gebruikt men de waarde 1 dan worden er twee videobeelden gedigitaliseerd. De eerste wordt direct weergegeven en het tweede beeld wordt opgeslagen op een lagere pagina.

CSAVE

CSAVE "<filenaam>"[,<snelheid in baud>]
Commando om een programma op cassette op te slaan (zie ook CLOAD). De naam van het programma mag niet meer dan zes karakters bevatten.
BASIC-programma's die met dit commando zijn opgeslagen, kunnen niet worden geMERGEd met andere programma's, zie hiervoor SAVE.

```
CSAVE "PROG",2
```

De toevoeging achter de naam van het programma heeft betrekking op de registratiesnelheid (baud):
1 1200 baud
2 2400 baud
Laat men deze aanduiding weg dan wordt de snelheid

van 1200 baud gebruikt. De baud-snelheid kan ook worden ingesteld met SCREEN.

DATA

DATA <n1>[,<n2> ...]
Achter dit commando kan men in een programma een lijst met vaste waarden opnemen die één voor één kunnen worden gelezen met het READ-commando. De gegevens achter DATA mogen uit numerieke waarden bestaan en uit strings, door komma's gescheiden. De strings moeten tussen aanhalingstekens (") staan. READ leest de waarden sequentieel (één voor één). Men kan tijdens het lezen weer vooraan beginnen bij de DATA-gegevens door eerst het RESTORE-commando te geven.

DEF FN

Hiermee kan men eigen functies definiëren. De naam waaronder de functie wordt aangeroepen, is de functienaam achter FN. Bij stringfuncties eindigt de naam op $. Eventueel kan men de functie argumenten geven die in de functiedefinitie worden gebruikt. De namen van deze argumenten dienen om de functie te definiëren en hebben geen invloed op variabelen met dezelfde naam die in het programma voorkomen. De argumenten moeten door een komma worden gescheiden.
In de functiedefinitie mogen ook variabelen voorkomen die niet als argument worden opgegeven, maar vóór het aanroepen van de functie een bepaalde waarde hebben gekregen in het programma. De functiedefinitie mag niet langer zijn dan één regel.

```
DEF FNKUBUS(x)=X^3
```

De functie kubus verheft de waarde van x tot de derde macht, om deze functie op te roepen, gebruikt men de volgende programmaregel:

```
30 PRINT FNKUBUS(A%)
```

De variabele A% wordt nu tot de derde macht verheven en op het beeldscherm weergegeven.

DEF...

DEF<type> <letter>
Dit commando zorgt ervoor dat alle variabelen die beginnen met de opgegeven letter(s) van het daarvoor opgegeven type zijn. Een normaal type-declaratieteken (dus %, #, $ of !) heeft voorrang op het DEF-commando.

```
DEFINT A
```
Alle variabelen beginnende met een A zijn integervariabelen.

```
DEFSGN A
```
Alle variabelen beginnende met de letter A zijn realvariabelen.

```
DEFDBL A-E
```
Alle variabelen die beginnen met A, B, C, D of E zijn dubbele-precisievariabelen.

```
DEFSTR A
```
Alle variabelen die beginnen met de letter A zijn stringvariabelen.

DEFUSR

DEFUSR[<getal>]=<geheugenadres>
Hiermee legt men het startadres van een machinetaalsubroutine vast. De definitie wordt gevolgd door een getal dat tussen 0 en 9 moet liggen. Laat men het getal weg dan wordt 0 aangenomen. Op grond van dit getal en met behulp van de USR-functie kan men nu de desbetreffende subroutine in machinetaal aanroepen.
Een voorbeeld:

```
DEFUSR1=&HF300
X=USR1(9)
```

DELETE

Verwijdert alle opgegeven regels, of alleen de laatste regel. Na uitvoering van dit commando keert men altijd terug in de BASIC-commando.

```
DELETE 10
```
verwijdert regel 10

```
DELETE 10-40
```
verwijdert de regels 10 t/m 40

```
DELETE -40
```
verwijdert alle regels vanaf 1 t/m 40

```
DELETE 40-
```
niet toegestaan

```
DELETE
```
verwijdert laatste regel

DIM

Reserveert geheugenruimte voor gespecificeerde array(s) en zet de array-elementen op nul. Indien men naar een array refereert die niet eerst door een DIM-commando is gecreëerd, wordt 10 als maximum index aangenomen. De laagste index is altijd 0. Met ERASE kan men een array wissen.

```
DIM A(5,5,5)
```
A is nu gedimensioneerd als een driedimensionale array.

DRAW

DRAW <string>

Met behulp van DRAW kan men rechte lijnen weergeven door gebruik te maken van eenvoudige codes (ook wel Graphics Macro Language genoemd). De codes vormen samen een string. Het commando werkt niet in de grafische modes 1 en 2.

```
DRAW "D60R120"
```
Geeft twee aanéénsluitende rechte lijnen. De eerste gaat over 60 beeldpunten naar beneden ('down 60' of afgekort D60) en de tweede 120 punten naar rechts (R120).

```
DRAW "C12D60C3R120"
```
Voor ieder lijnstuk mogen we ook een kleurnummer opgeven voorafgegaan door de letter C.
De volgende tabel geeft een overzicht van de GML:

Code *Betekenis*

A Na A komt 0, 1, 2 of 3 te staan. Hiermee kunnen we het coördinatenstelsel in stappen van 90° verdraaien. Zonder een opgave zal de computer geen verdraaiing toepassen.

B Met de letter B voor één van de commando's wordt er een blanco lijn getekend, dus niet zichtbaar op het beeldscherm.

C Geeft de kleur van het lijnstuk aan.

D D slaat op 'down' en trekt een lijn naar beneden.

E Voor een lijn die onder een hoek van 45° naar rechtsboven wordt getrokken. E60 geeft als resultaat: een lijn die vanaf het laatste punt naar een punt gaat dat 60 beeldpunten naar rechts en 60 beeldpunten hoger ligt.

F Voor een lijn die onder een hoek van 45° naar rechtsonder wordt getrokken.

G Voor een lijn die onder een hoek van 45° naar linksboven wordt getrokken.

H Voor een lijn die onder een hoek van 45° naar linksonder wordt getrokken.

L L slaat op 'left' en trekt een lijn naar links.

M Voor het tekenen van een schuine lijn. Na M komen twee getallen gescheiden door een komma, bijvoorbeeld:

```
M20,40
```

Nu zal er een schuine lijn worden getrokken vanaf de laatste positie, tot het punt dat we krijgen door bij de x-coördinaat 20 en bij de y-coördinaat 40 op te tellen. De waarden x en y kunnen ook negatief zijn.

N De letter N voor een van de commando's trekt normaal een lijn, maar de cursor keert terug naar het beginpunt van de lijn.

R R slaat op 'right' en trekt een lijn naar rechts.

S Geeft de schaal aan. Na S volgt een getal tussen 0 en 255. De schaalfactor komt overeen met 'getal/4'. S2 geeft dus een schaalfactor van

½ en tekent dan lijnen die de helft zijn van de opgegeven waarden.

U U slaat op 'up' dat wil zeggen 'naar boven' en trekt een lijn naar boven.

END **END**

Beëindigt een programma. Dit commando mag overal in het programma voorkomen. Een END aan het einde van een programma is niet verplicht. Bij dit commando worden alle nog geopende files automatisch gesloten.

ERASE **ERASE <array-naam>[,<array-naam> ...]**

Met ERASE kunnen we een array wissen waardoor weer geheugenruimte vrijkomt (zie ook DIM).

```
10 DIM A(200), X$(40)
. . .
500 ERASE A, X$
```

FOR/ NEXT **FOR I=X TO Y: ... :NEXT**

De instructies die tussen FOR en NEXT zijn opgenomen, worden herhaald uitgevoerd totdat de teller de opgegeven waarde overschrijdt. De waarde van de teller (X) wordt iedere keer als alle instructies tussen 'FOR' en 'NEXT' zijn uitgevoerd met één verhoogd, tot de waarde van Y is bereikt.

FOR ... TO ... STEP: ... :NEXT

Achter het commando FOR ... NEXT kan men een stapgrootte opgeven; de teller wordt dan steeds verhoogd met de waarde van die stap.
FOR...NEXT-commando's mogen op hun beurt andere FOR...NEXT-commando's bevatten.

GET DATE **GET DATE <string>[,A]**

Commando (MSX2) voor het opvragen van de datum die is opgeslagen in de klokchip. Neemt men achter de variabele de letter A op, dan wordt de alarmdatum opgehaald.

```
10 GET DATE DD$
20 PRINT DD$
```

GET TIME

GET TIME <string>[,A]
Commando (MSX2) voor het opvragen van de tijd die is opgeslagen in de klokchip. Neemt men achter de variabele de letter A op dan wordt de alarmtijd ingelezen.

```
10 GET TIME TT$
20 PRINT TT$
```

GOSUB... RETURN

GOSUB <regelnummer> RETURN
Deze commando's maken het mogelijk om naar een subroutine te springen. Het eerste regelnummer van de subroutine wordt opgegeven. Een RETURN moet de subroutine afsluiten en zorgt ervoor dat wordt teruggekeerd naar de eerstvolgende instructie na het GOSUB-commando. Subroutines mogen ook naar andere subroutines wijzen, iedere subroutine dient men af te sluiten met een RETURN.

GOTO

GOTO <regelnummer>
De uitvoering van het programma wordt vervolgd bij het opgegeven regelnummer.

IF... THEN... ELSE

Tussen IF en THEN wordt een logische vergelijking opgenomen, waarin de volgende tekens (operators) kunnen voorkomen:

Teken	*Betekenis*
<	kleiner dan
<=	kleiner dan of gelijk aan
>	groter dan
>=	groter dan of gelijk aan
=	gelijk aan
<>	ongelijk aan

De volgende logische operators zetten de waarde eerst om in een binaire waarde en volgen dan de regels van de Booleaanse algebra:

Teken	*Betekenis*
AND	om twee vergelijkingen op te nemen
EQV	om twee waarden te vergelijken
IMP	impliceert twee waarden
NOT	om het tegengestelde van een vergelijking te testen
OR	om twee vergelijkingen te vergelijken
XOR	vergelijkt twee waarden volgens de exclusive or methode

Als de hele expressie (uitdrukking) tussen IF en THEN waar is, worden de commando's na THEN uitgevoerd. Volgt na THEN geen commando maar een regelnummer dan wordt verdergegaan met de opgegeven regel.
Als de hele expressie tussen IF en THEN niet waar is, worden de commando's na ELSE uitgevoerd of wordt naar de opgegeven regel gesprongen. Is geen ELSE aanwezig dan wordt verder gegaan met de regel die direct volgt na het
IF...THEN-commando.
De commando's na THEN of ELSE mogen ook een volgend IF...THEN-commando bevatten, mits het geheel op één programmaregel van maximaal 255 karakters staat.

INPUT

INPUT ["<tekst>"],<variabele 1>[,<variabele 2>...]

Vraagt om gegevens die door de gebruiker via het toetsenbord moeten worden ingevoerd. Op het beeldscherm verschijnt er een vraagteken (?) om aan te geven dat er om gegevens wordt gevraagd.
De ingetoetste waarde of string wordt opgeslagen in een variabele. Er moeten net zoveel waarden, gescheiden door een komma, worden ingetoetst als er variabelen voorkomen in het INPUT-commando. Bij stringvariabelen mogen er geen komma's in de string voorkomen en ze mogen niet langer zijn dan 200 karakters, tenzij er eerst met CLEAR meer ruimte is toegekend aan de variabele. Treedt er in de ingetoetste waarden een fout op dan verschijnt de foutmel-

ding '?Redo from start' en wordt het INPUT-commando opnieuw uitgevoerd.

INPUT # **INPUT #<filenummer>,<stringvariabele>**
Dit commando komt overeen met INPUT, maar het gaat nu niet om het lezen vanaf het toetsenbord. In plaats daarvan worden er waarden vanuit een file ingelezen. De waarden in de file moeten in dezelfde volgorde staan als de variabelen in het INPUT #-commando. Tijdens het inlezen van een string wordt het einde van een string bepaald door een komma, RETURN of line feed. Is het eerste teken van de string een aanhalingsteken ("), dan wordt er een tweede aanhalingsteken aan het einde van de string opgenomen.

INTERVAL **INTERVAL ON**
INTERVAL OFF
INTERVAL STOP
Hiermee kan men de onderbreking die met het commando ON INTERVAL GOSUB is opgegeven, inschakelen, uitschakelen of stop zetten.

```
10 ON INTERVAL=50 GOSUB 100
20 IF INKEY$="j" THEN INTERVAL ON
30 IF INKEY$="n" THEN INTERVAL OFF
40 GOTO 20
...
...
100 REM INTERVAL SUBROUTINE
110 BEEP
120 PRINT "Subroutine test"
130 RETURN
```

Na het starten van het programma zal de computer wachten op een toetsaanslag. Bij het indrukken van de 'j' zal de interval-subroutine worden ingeschakeld. Na het verstrijken van de tijd (hier dus 1 seconde) springt het programma naar regel 100. Na een korte pieptoon wordt regel 120 uitgevoerd die de boodschap op het beeldscherm weergeeft. Het programma

keert nu terug naar regel 30, na de instructie 'INTERVAL ON'.
Het programma zal nu steeds om de seconde naar de subroutine gaan, tot men de 'n'-toets indrukt waardoor de interval-subroutine wordt uitgezet.

KEY Commando voor het definiëren van een opdracht (serie commando's) en het toekennen aan een functietoets. Het nummer van de functietoets kan 1 t/m 10 zijn. De opdracht moet tussen aanhalingstekens (") staan en mag maximaal 15 schrifttekens bevatten. De opdracht kan ook in een stringvariabele staan. Ook een RETURN kan men in zo'n stringvariabele opnemen door het gebruik van '+CHR$(13)'.

KEY 5,"RUN"+CHR$(13)
Wordt nu functietoets 5 ingedrukt dan wordt het programma dat zich in het geheugen bevindt direct gestart.

KEY LIST
Geeft op het beeldscherm een overzicht van de commandostrings die men kan activeren met de 10 functietoetsen.

KEY ON
KEY OFF
Hiermee kunnen we de betekenis van de functietoetsen al dan niet op het beeldscherm laten verschijnen. Eventueel kan men de toetsen apart aangeven:

KEY (*n*)ON
KEY (*n*)OFF
KEY (*n*)STOP

Hiermee kan men vaststellen of één van de functietoetsen werd ingedrukt. Met ON schakelt men deze toestand in en met OFF schakelt men deze toestand uit.
Met STOP zal de computer niet onmiddellijk naar de subroutine volgens ON KEY GOSUB springen, maar dit pas doen zodra KEY (*n*) ON word gegeven.

Tussen haakjes vertegenwoordigt *n* het nummer van de functietoets.

LET — Dit commando kent een waarde aan een variabele toe. Het commando LET kan men eventueel weglaten.

```
10 LET X=100
```

Dit commando komt dus overeen met:

```
10 X=100
```

LINE — Met dit commando kan men op het beeldscherm lijnen tekenen.

`LINE (X1,Y1)-(X2,Y2)`
Tekent een lijn tussen (X1,Y1) en (X2,Y2). De op te geven coördinaten zijn absolute waarden voor X tussen 0 en 255 en voor Y tussen 0 en 191.

`LINE (X1,Y1)-10-(X2,Y2)`
Met het getal tussen de begin- en eindcoördinaten kunnen we aangeven dat het coördinatenstelsel moet worden verplaatst.

`LINE (X1,Y1)-10-(X2,Y2),3,B`
Deze instructie tekent een vierkant dat overeenkomt met de opgegeven diagonaal.
De letter B geeft aan dat er een vierkant moet worden getekend in de kleur ... (3). Door de toevoeging BF wordt het vierkant ingekleurd.

LINE INPUT — **LINE INPUT <string>**
Leest een string die wordt ingetoetst via het toetsenbord. De string mag uit alle mogelijke schrifttekens bestaan en een maximumlengte hebben van 255 tekens.

LINE INPUT # — **LINE INPUT #<filenummer>,<string>**
Dit commando komt overeen met LINE INPUT, maar leest nu een string in van een file.

```
10 LINE INPUT #1,A$
```
De desbetreffende file dient eerst te worden geopend met een filenummer; dit filenummer moet men ook gebruiken bij dit commando. De string wordt ingelezen tot aan een RETURN.

LIST

LIST [<regelnummer>[-[<regelnummer>]]]
Dit commando zal de programmaregels op het beeldscherm weergeven. Men kan de listing laten stoppen door de STOP-toets in te drukken.

```
LIST
```
Het complete programma wordt weergegeven.

```
LIST 200
```
Alleen regel 200 wordt weergegeven.

```
LIST 200-350
```
Alle regels van 200 tot en met 350 worden weergegeven.

```
LIST -300
```
Alle regels van 0 tot en met 300 worden weergegeven.

```
LIST 400-
```
Alle regels vanaf 400 worden weergegeven.

```
LIST.
```
Alleen de laatst behandelde regel wordt weergegeven.

LLIST

LLIST
Komt overeen met LIST, alleen worden de regels nu door de printer afgedrukt.

LOAD

LOAD"<dev>:<filenaam>"[,R]
Voor het laden van een programma dat met het SAVE-commando is weggeschreven.

```
LOAD "CAS:PROG"
```
Laadt van cassette het programma dat onder de naam PROG is opgeslagen.

LOAD "A:PROG"
Laadt het programma van diskdrive A: en slaat dit op in het geheugen.

LOCATE **LOCATE [<X>][,[<Y>][,<cursor aan/uit>]]**
Verplaatst de cursor naar de opgegeven plaats.

LOCATE x,y,1
De cursor kunnen we aan- of uitzetten door een 1 of een 0 aan te geven. X mag variëren tussen 0 en 39 en Y tussen 0 en 23.

LPRINT **LPRINT USING**
Als PRINT en PRINT USING maar nu wordt de printer aangestuurd.

MAX-FILES **MAXFILES=<variabele>**
Bepaalt het maximale aantal gelijk geopende files (bestanden) en reserveert voor ieder bestand een buffer.

MERGE **MERGE "<dev>:[<filenaam>]"**
Een op een extern geheugen opgeslagen programma wordt toegevoegd aan een in het geheugen aanwezig programma. Het toe te voegen programma moet wel in ASCII-formaat zijn opgeslagen.
De programmaregels worden in volgorde van de regelnummers toegevoegd. Hebben regels hetzelfde nummer dan wordt de regel in het geheugen overschreven door de regel van het ingelezen programma. Het aldus ontstane programma blijft in het geheugen staan.

MOTOR **MOTOR ON**
MOTOR OFF
Met dit commando kan men de aangesloten cassetterecorder vanuit de computer besturen. MOTOR ON heeft het effect alsof men bij de recorder op de PLAY-toets drukt; MOTOR OFF schakelt de recorder weer uit. Het commando MOTOR zal de motor beurtelings in- en uitschakelen.

NEW

NEW
Een in het geheugen opgeslagen programma wordt gewist.

ON ... GOTO

ON ERROR GOTO <regelnummer>
Deze instructie zal bij een optredende fout in het programma het programma voortzetten op het regelnummer achter GOTO. Hierdoor heeft men de mogelijkheid om eigen fout-afhandelingsroutines te maken. In deze routine kan men gebruik maken van de variabelen ERR en ERL.

ON N GOTO <regelnummer>
Hiermee kan men de programma-uitvoering voortzetten op een regelnummer dat wordt bepaald door de waarde van N. Heeft N bijvoorbeeld de waarde 2 dan zal het programma worden voortgezet op het tweede regelnummer dat na GOTO is gespecificeerd. De waarde van N mag niet negatief zijn. Is N groter dan het aantal regelnummers dan gaat het programma naar de eerstvolgende programmaregel.
De regelnummers achter GOTO worden gescheiden door komma's.

ON ... GOSUB

Werkt net als het commando ON N GOTO, maar nu moeten de regelnummers naar een subroutine verwijzen.

```
ON INTERVAL=200 GOSUB 500
```

Hiermee wordt aangegeven dat na een door INTERVAL aangegeven wachttijd het programma naar de aangegeven subroutine moet springen.
Het getal '200' geeft de duur van de onderbreking in eenheden van 1/50 seconde aan. Na dit commando dient men de INTERVAL-functie aan te zetten met het commando INTERVAL ON. Als binnen de opgegeven tijd het commando INTERVAL OFF niet wordt gegeven, zal de programma-uitvoer vanaf de opgegeven subroutine verder gaan, zie ook bij het commando INTERVAL.

ON KEY GOSUB <regelnummer>
Met dit commando kan men naar een subroutine springen als één van de functietoetsen (F1 t/m F10) wordt ingedrukt.

```
10 ON KEY GOSUB 60,50
20 KEY (1) ON
30 KEY (2) ON
40 GOTO 40
50 BEEP
55 FOR I=1 TO 100 : NEXT
60 BEEP
70 RETURN
```

Bij dit testprogramma zal het indrukken van functietoets 1 tot gevolg hebben dat het programma naar regel 60 springt. Men krijgt dan één pieptoon te horen. Drukt men functietoets 2 in dan hoort men twee pieptoontjes.

ON SPRITE GOSUB <regelnummer>
Met dit commando kan men naar een subroutine springen zodra twee sprites elkaar overlappen. Eerst dient wel het SPRITE ON-commando te worden gegeven.

ON STOP GOSUB <regelnummer>
Met dit commando kan men naar een subroutine springen zodra het programma met CTRL/STOP wordt onderbroken. Met het commando STOP ON wordt de instructie actief.
Deze subroutine kan het programma zichzelf dus laten voortzetten.

ON STRIG GOSUB <regelnummer>
Met dit commando kan men naar een subroutine springen aan de hand van de vuurknoppen van de joystick. STRIG kan de volgende waarden aannemen:

0 spatiebalk
1 joystick 1, vuurknop 1
2 joystick 1, vuurknop 2
3 joystick 2, vuurknop 1
4 joystick 2, vuurknop 2

De desbetreffende vuurknop/spatiebalk moet wel met het commando STRIG (*n*) ON worden geactiveerd.

OPEN

OPEN <apparaat:filenaam> FOR <mode> AS# <filenummer>
Hiermee wordt een bestand geopend naar één van de volgende apparaten:

CAS:	cassetterecorder;
COM:	RS232-interface;
CRT:	tekstbeeldscherm;
GRP:	grafisch beeldscherm;
LPT:	printer;
MEM:	memory-disk (MSX2);
A t/m D:	diskdrive;
E en F:	diskdrive (MSX2).

Men kan verschillende modes opgeven:

INPUT	lezen van gegevens uit een file
OUTPUT	invoeren van gegevens in een file
APPEND	alleen bij disk BASIC, zie aldaar

Het filenummer is het logische filenummer van de geopende file. Als men 'FOR' weglaat, wordt er een random file geopend.

OUT

OUT <poortnummer>,<uitdrukking>
Hiermee kan men gegevens naar een bepaalde poort sturen.

PAINT

Met dit commando kan men een bepaald gebied inkleuren. Met STEP kan het coördinatenstelsel worden verplaatst.

PAINT (X,Y),3

Met X en Y wordt een punt opgegeven binnen het vlak dat moet worden ingekleurd. Als het vlak niet volledig gesloten is, wordt het gehele scherm ingekleurd. X ligt tussen 0 en 255 en Y tussen 0 en 191.
Bij het grafische scherm in hoge resolutie (SCREEN 2) geldt dat als de kleur van het vlak en de kleur van de reeds gegeven randlijnen niet gelijk zijn, het hele scherm wordt ingekleurd. Dit geldt ook voor SCREEN 4 (MSX2).
In screenmode 6 (MSX2) dient het kleurnummer tussen 0 en 3 te liggen, in screenmode 8 tussen 0 en 255 (zie COLOR) en in de overige screenmodes tussen 0 en 15.

PLAY

PLAY <string>

Commando om geluid voort te brengen volgens een bepaalde code. Als aanduiding voor de maten kan men een aantal letters en/of cijfers gebruiken die de volgende betekenis hebben:

A t/m G — Geeft de toon aan. Achter de toon kan men de volgende tekens opnemen; # of + betekent SHARP (de toon wordt verhoogd); − betekent FLAT (de toon wordt verlaagd). Een cijfer achter de toon geeft het octaaf aan.

L — Geeft de duur van een toon aan en wel met de kwartnoot als ritmische eenheid:

Duur	*Code*	
4 tellen	L1	hele noot
2 tellen	L2	halve noot
1 tel	L4	kwartnoot
½ tel	L8	achtste noot
¼ tel	L16	enz.
⅛ tel	L32	
1/16 tel	L64	

Zonder nadere aanduiding neemt de computer L4 aan.

M — Geeft de mate aan waarin de klankkleur varieert. Zonder nadere aanduiding

neemt de computer M255 aan. M ligt tussen 1 en 65535.

N — Met N geeft men de toon aan, maar nu gespecificeerd met een getal tussen 0 en 96. Deze getallen omvatten alle noten over de acht beschikbare octaven.

O — Geeft het desbetreffende octaaf aan, bijvoorbeeld PLAY 'O5CDE' wil zeggen: speel C, D en E in octaaf 5. Voor het getal achter O geldt: groter dan of gelijk aan 1 en kleiner dan of gelijk aan 8. Zonder nadere aanduiding neemt de computer O4 aan.

R — Geeft de rust weer in tellen, zie ook 'L'. R4 komt neer op een rust ter lengte van een hele noot en R1 op een rust ter lengte van een kwartnoot. De opeenvolgende mogelijkheden zijn dan: R1, R2, R4, R8, R16, R32 en R64.

S — Geeft de klankkleur aan, het getal achter S moet groter dan of gelijk aan nul en kleiner dan of gelijk aan 15 zijn. Zonder nadere aanduiding neemt de computer S1 aan.

T — Geeft het tempo aan. De waarde van het getal achter T bepaalt het aantal kwartnoten in een minuut. Dit kan liggen tussen 32 en 255. De automatisch ingestelde standaardwaarde is 120.

V — Geeft het volume aan: het getal 15 komt met 'maximaal volume' overeen. Zonder nadere aanduiding neemt de computer V8 aan.

Verlengt de duur van een noot met een factor 1,5. Dit komt overeen met de duur van een noot die (in notenschrift) van een punt is voorzien.

POKE — **POKE <geheugenplaats>,<waarde>**

Plaatst een gegeven in de opgegeven geheugenplaats. Dit commando is het complement van de functie PEEK.

PRESET Zet of wist een punt op het grafische scherm. Als een kleur is opgegeven, is het effect van PRESET gelijk aan dat van PSET. Zonder deze aanduiding wordt een reeds aanwezig punt gewist.

PRESET <STEP> (X,Y)
Zet het beeldpunt op de aangegeven coördinaten terug in de achtergrondkleur, waardoor het punt voor ons oog van het beeldscherm is gewist. Met STEP kan men eventueel het coördinatenstelsel verplaatsen.

PRINT Geeft op het scherm de waarde van variabelen, numerieke uitdrukkingen of strings. Strings moeten tussen aanhalingstekens (") staan.

```
10  A$="Boodschap  tussen  aanha-
lingstekens"
20 PRINT A$
```

In programmaregel 20 wordt de variabele A$ op het beeldscherm weergegeven.

```
PRINT CHR$(65)
```
Dit zal de letter 'A' weergeven op het beeldscherm.

PRINT # Dit commando dient voor het schrijven naar andere apparaten, zoals bijvoorbeeld het grafische beeldscherm. Men dient eerst een kanaal te openen om de gegevens te kunnen schrijven. Dit commando wordt ook gebruikt voor het schrijven van gegevens naar cassette of diskette.
Ook te gebruiken in combinatie met het commando PRINT USING, dus PRINT # USING.

PRINT USING Dit commando is een uitbreiding van het PRINT-commando, waarbij men zelf met behulp van een code kan opgeven in welke vorm de gegevens moeten worden weergegeven. Bij het weergeven van strings kan men één van de volgende codes kiezen:

!	Alleen het eerste schriftteken van de string wordt weergegeven.

*n*spaties Nu worden de eerste 2+*n* schrifttekens van de string weergegeven.

& De string wordt volledig weergegeven.

Geeft het aantal posities voor en na de komma aan. Te grote getallen worden afgerond. Te kleine getallen worden door middel van nullen en spaties passend gemaakt. Is het getal te groot voor de gespecificeerde code, dan wordt er een procentteken (%) voor het getal weergegeven.

@ Plaatst een opgegeven string op de plaats van '@' in een andere string.

```
PRINT USING "ABC@DEF";"MSX"
ABCMSXDEF
```

Er mogen nooit meer dan 24 plaatsen worden gedimensioneerd.

```
PRINT USING "##.##";1.2345
```

Dit wordt weergegeven als 1.23.

```
PRINT USING "##.##";99.996
```

Dit wordt weergegeven als %100.00.

\+ Een plusteken voor of aan het einde van de code zorgt ervoor dat een plus- of minteken voor of aan het einde van het getal wordt weergegeven.

\- Een minteken aan het einde van de code zorgt ervoor dat bij negatieve getallen een minteken aan het einde wordt weergegeven.

** Een dubbele asterisk aan het begin van de code zorgt ervoor dat eventuele aan het getal voorafgaande spaties met asterisktekens worden aangegeven. Tevens tellen deze twee asterisk-tekens als twee extra posities.

$$ Een dubbel $-teken aan het begin van de code zorgt ervoor dat een $-teken voor het getal wordt weergegeven. De wetenschappelijke notatie kan niet worden gebruikt in combinatie met $$.

**$ Resulteert in een combinatie van de twee hiervoor genoemde effecten.
Een komma aan de linkerkant van de decimale punt zorgt ervoor dat er om de drie posities een komma wordt weergegeven, gerekend vanaf de decimale plaats naar de positie met de hoogste waarde.

```
PRINT USING "####,.##";1234.5
```
Dit wordt weergegeven als 1,234.50.
Een komma aan het einde van de code wordt gewoon weergegeven.

```
PRINT USING "####.##,";1234.5
```
Dit wordt dus weergegeven als 1234.50,.
Door vier dakjes aan het einde van de code te plaatsen, wordt het getal in wetenschappelijke notatie weergegeven.

```
PRINT USING "##.## ^^^^";123.45
```
Dit wordt weergegeven als 1.23E+0.2.
Tekst voor of achter de code, mits van de code gescheiden door een spatie, wordt gewoon weergegeven.

```
PRINT USING "FL ##.## GULDEN";
24.56
```
De tekst wordt normaal weergegeven en het resultaat zal er dus zó uitzien:

```
FL 24.56 GULDEN
```

PSET

PSET [STEP](x,y)[,<Z>[,<operatie>]]
Plaatst punten op het beeldscherm, het commando komt overeen met PRESET.

PUT SPRITE

PUT SPRITE <vlaknummer>,(x,y),<kleur>,<patroonnummer>
Voor een 8×8-sprite kan men als nummer een getal

tussen 0 en 255 nemen. Voor een 16×16-sprite een getal tussen 0 en 63.
Indien twee sprites op dezelfde positie op het scherm staan, is de sprite met het laagste prioriteitsnummer zichtbaar.
Het beeldscherm kent de volgende coördinaten:

X = −32 tot 255

Y = −32 tot 191

In de screenmodes 2 en 3 kunnen maximaal vier sprites naast elkaar op één regel worden weergegeven. In de screenmodes 4 t/m 8 (MSX2) kunnen maximaal acht sprites op één regel.

READ

READ <variabele>
READ wordt altijd gebruikt in combinatie met DATA en kent de door DATA gespecificeerde waarden toe aan de opgegeven variabelen. Zie ook DATA.

REM

Hiermee is het mogelijk een programma te voorzien van commentaar om de verschillende routines te verduidelijken. Alle tekst na REM wordt door BASIC genegeerd. REM kan worden afgekort met het symbool '.

RENUM

RENUM <nieuw regelnummer>,<oud regelnummer>,<stapgrootte>
Dit commando dient voor het opnieuw nummeren van de programmaregels.
Het opgegeven nieuwe regelnummer is het beginnummer van de eerste regel. Specificeert men dit niet dan wordt met 10 begonnen.
Het oude regelnummer dient te worden opgegeven om de computer duidelijk te maken vanaf welke regel het hernummeren moet beginnen. Geeft men dit niet op dan wordt met de eerste regel begonnen.
Tevens dient men de stapgrootte waarmee het regelnummer steeds moet worden verhoogd, op te geven. De standaard-stapgrootte is 10.

RENUM ,,20
Dit commando zal het programma in het geheugen opnieuw nummeren beginnende bij de eerste regel en zal deze eerste regel het regelnummer 10 geven, de stapgrootte wordt in dit geval 20.

RE-STORE **RESTORE [<regelnummer>]**
Met dit commando wordt deldata-pointer gereset, zodat deze naar het begin van de DATA wijst. Door een regelnummer achter het commando op te geven, wordt er naar de desbetreffende regel met DATA gesprongen.

RESUME **RESUME [<regelnummer>]**
Het RESUME-commando wordt alleen in fout-afhandelingsroutines gebruikt. Het heeft dezelfde functie als RETURN in subroutines.

RESUME 100
Het regelnummer duidt de regel aan waar de programma-uitvoering moet worden hervat. Resume neemt men op aan het einde van een fout-afhandelingsroutine.

RETURN **RETURN**
Commando behorende bij GOSUB-routines. Alle GOSUB-routines dient men te beëindigen met het commando RETURN.

RUN **RUN [<regelnummer>]**
RUN ["<dev>:<filenaam>"[,R]
Start de programma-uitvoering bij de eerste programmaregel. Geeft men een regelnummer op dan start de uitvoering bij de opgegeven programmaregel. Geopende files worden gesloten, tenzij de optie R wordt opgegeven.

RUN"CAS:"
Dit commando start een bepaald programma dat op cassette is opgeslagen. Door ,R achter de naam op te nemen, geeft men aan dat eventueel geopende files niet worden gesloten.

SAVE

Met dit commando wordt een bestand weggeschreven naar één van de volgende apparaten:

CAS:	cassetterecorder (in ASCII-code);
COM:	RS-232 interface;
CRT:	tekstbeeldscherm;
GRP:	grafisch beeldscherm;
LPT:	printer;
MEM:	memory-disk (MSX2);
A t/m D:	diskdrive;
E en F:	diskdrive (MSX2).

```
SAVE "CAS:PROG"
```
Slaat een BASIC-programma onder de naam PROG op cassette op.

```
SAVE "A:PROG"
```
Slaat een programma op onder de naam PROG, maar nu op de diskette-eenheid A.

```
SAVE "A:PROG",A
```
Het programma 'PROG' wordt nu als een ASCII-file weggeschreven naar de diskette. Het voordeel hiervan is dat programma's die op deze manier zijn opgeslagen, kunnen worden samengevoegd. Zie hiervoor ook bij MERGE.

SCREEN

SCREEN[<X>[,<Y>[,<Z>[,<YY>[,<ZZ>]]]]]
Met het SCREEN-commando bepaalt men de indeling van het beeldscherm. Om het grafische scherm in hoge resolutie aan te duiden, geeft men SCREEN 2 op. Zonder nadere specificatie neemt de computer SCREEN 0,0,1,1,0 aan.
Hierbij hebben de vijf getallen respectievelijk de volgende betekenis:

Mode:

0 Tekstscherm: 40×24 regels.
1 Tekstscherm: 32×24 regels.
2 Grafisch scherm (hoge resolutie = 256×192 pixels).
3 Grafisch scherm (lage resolutie = 64×48 pixelblokken).

Formaat sprite:

0	8×8	Zonder vergroting.
1	8×8	Met vergroting, factor 2.
2	16×16	Zonder vergroting.
3	16×16	Met vergroting, factor 2.

Geluid:

1	Wél geluid bij indrukken van een toets.
0	Geeft geen piepsignaal bij indrukken van een toets.
Ander getal	Geeft wél een piepsignaal bij indrukken van een toets.

Snelheid bij in/uitvoer van cassette:

1	1200 baud.
2	2400 baud.

Soort afdrukeenheid:

0	MSX-type.
1	Ander type.

Bij MSX2 is het SCREEN-commando uitgebreid met de volgende modus:

Mode:

4	Grafisch scherm (hoge resolutie 256×192 pixels).
5	Grafisch scherm (hoge resolutie 256×212 pixels).
6	Grafisch scherm (hoge resolutie 512×212 pixels).
7	Grafisch scherm (hoge resolutie 512×212 pixels).
8	Grafisch scherm (hoge resolutie 256×212 pixels).

Afhankelijk van de screenmode zijn er meer of minder kleuren beschikbaar:

Mode	*Aantal kleuren*
0	2
1	2
2	16
3	16
4	16
5	16
6	4
7	16
8	256

Bij MSX2 kan men achter het volledige SCREEN-commando nog een waarde opgeven waar men de weergave kan instellen:

Display	*Omschrijving*
0	Normaal;
1	Interlaced (zelfde pagina);
2	Normaal, eerst even en daarna oneven pagina;
3	Interlaced, eerst even en daarna oneven pagina.

SET

SET ADJUST <x,y>
Met dit commando (MSX2) kan men het beeld zoals dat op de monitor of televisie verschijnt iets verplaatsen. Dit is vooral makkelijk indien onze monitor of televisie het computerbeeld niet goed in het midden van het scherm weergeeft. De X- en Y-coördinaten dienen tussen de –7 en +8 te liggen. De waarden die we opgeven, zullen in de klokchip worden opgeslagen en gelden als standaard-instellingen bij het aanzetten van de computer.

SET BEEP <toonhoogte>,<volume>
Met dit commando (MSX2) kan men de pieptoon van de computer instellen. De eerste waarde achter het commando ligt tussen 1 en 4 en geeft de toonhoogte aan, de tweede waarde gescheiden door een komma geeft het volume aan en ligt tussen 1 en 4. De gegevens worden opgeslagen in de klokchip en bewaard, ook als de computer wordt uitgezet.

SET DATE <string>
Hiermee kunnen we de datum opgeven en veranderen (MSX2). Afhankelijk van de versie die u heeft, dient de datum als volgt te worden opgegeven:

MM/DD/YY
DD/MM/YY
YY/MM/DD

Nemen we achter de datum de letter A op, dan gaat het om de datum dat de computer ons moet alarmeren.

SET PAGE <variabele>
MSX2-commando om pagina's in het videogeheugen op te slaan of op te roepen. Afhankelijk van de gekozen screenmode hebben we twee of vier pagina's ter beschikking. De pagina's hebben de volgende nummers:

SCREEN	*Pagina's*
5	0, 1, 2 en 3
6	0, 1, 2 en 3
7	0 en 1
8	0 en 1

SET PAGE 0,1
Hiermee geven we aan dat pagina '0' op het beeldscherm moet worden weergegeven, pagina '1' wordt in het geheugen opgeslagen, ook tijdens het opbouwen van de afbeelding.

SET PASSWORD <string>
Dit MSX2-commando zorgt ervoor dat er een 'wachtwoord' in het geheugen wordt opgeslagen. Het wachtwoord mag uit maximaal 255 karakters bestaan. Telkens als de computer is aangezet, zal deze vragen om het wachtwoord.

SET PROMPT <string>
Dit MSX2-commando laat u de prompt wijzigen die

na ieder commando wordt weergegeven. De standaardinstelling is de prompt 'Ok'.

SET SCREEN
Bij MSX2 kan men vaste waarden opslaan in de klokchip. De computer werkt na het aanzetten dan in eerste instantie met deze standaard-waarden. De volgende instellingen zijn mogelijk:

SCREEN-mode	0 of 1
WIDTH	1 t/m 80
COLOR	0 t/m 15 voor zowel voorgrond, achtergrond als rand
KEY	ON of OFF
PRINTER	MSX of geen MSX
Snelheid	1200 of 2400 baud (cassette)
Display-mode	0 t/m 3
Toetsaanslag	Wél of geen geluid

Het commando SET SCREEN wordt gegeven meteen nadat de bewuste waarden zijn opgegeven. Een voorbeeld:

```
10 SCREEN 0:WIDTH 80:KEY OFF
20 SET SCREEN:END
```

SET TIME <string>,[A]
MSX2-commando voor het instellen van de tijd, in de volgorde HH:MM:SS. Geeft men een A op achter de tijd, dan wordt deze als alarmtijd opgeslagen.

SET TITLE <string>
MSX2-commando om een eigen titel te laten weergegeven bij het opstarten van de computer. Achter de string kan men een kleurnummer opnemen waarin de titel moet worden weergegeven. Is de opgegeven titel langer dan vijf karakters dan zal het systeem wachten met het weergeven tot men een toets indrukt.
Van de drie commando's SET PASSWORD, SET PROMPT, SET TITLE wordt alleen de laatst gegeven commando-string door de computer onthouden.

SET VIDEO <beeld>,<intensiteit>,<kleuren>, <synchronisatie>,<audio>,<video>, <scart>
MSX2-commando dat alleen werkt als uw computer is uitgerust met de optie 'super imposed', waarmee beelden kunnen worden gemengd.

Beeld	*Beeld komt:*
0	van de computer
1	van de computer
2	samen met een ander beeld (superimposed)
3	van de video-ingang

Bij beeld 0 is er geen externe synchronisatie mogelijk en bij de overige beelden is er geen gemengde uitvoer mogelijk.

Intensiteit

0	half
1	vol

Kleuren	*Besturing*
0	alleen naar de uitgang
1	alleen voor de ingang

Synchronisatie

0	intern
1	extern

Audio

0	alleen van computer
1	mengen computer met externe ingang rechts
2	mengen computer met externe ingang links
3	mengen computer met beide externe kanalen

Video

0	RGB-scartaansluiting
1	TV-aansluiting

Scart

0	aansluiting actief
1	aansluiting niet actief

Standaard neemt de computer altijd 'nul' aan.

SOUND **SOUND<registernaam,getal>**
Met dit commando wordt een bepaald geluid gegenereerd. Hierbij gelden de volgende codes:

Register	*Bereik*	*Betekenis*
0	0-255	Frequentie kanaal A
1	0-15	Frequentie kanaal A
2	0-255	Frequentie kanaal B
3	0-15	Frequentie kanaal B
4	0-255	Frequentie kanaal C
5	0-15	Frequentie kanaal C
6	0-31	Frequentie ruis
7	0-63	Keuze kanaal: toon en ruis
8	0-15(16)	Volume kanaal A
9	0-15(16)	Volume kanaal B
10	0-15(16)	Volume kanaal C
11	0-255	Frequentie van patroon-variatie
12	0-255	Frequentie van patroon-variatie
13	0-15	Keuze patroon

SPRITE **SPRITE ON**
SPRITE OFF
SPRITE STOP
Met het commando SPRITE ON geeft men de computer opdracht om op botsende sprites te letten. Door het SPRITE STOP-commando onderbreekt men de situatie. Met het SPRITE ON-commando wordt er weer naar de door ON SPRITE GOSUB opgegeven subroutine gesprongen.

STOP Dit commando onderbreekt de programma-uitvoering. Door middel van het CONT-commando gaat de programma-uitvoering weer door. Let op: dit is het *commando* STOP en niet de *toets* STOP.

STOP ON
STOP OFF
STOP STOP
STOP STOP houdt de onderbrekingstoestand aan die werd veroorzaakt door een CTRL/STOP. Pas als er een STOP ON wordt gedetecteerd, springt het programma naar de door ON STOP GOSUB opgegeven subroutine.

STRIG **STRIG ON**
STRIG OFF
STRIG STOP
STRIG STOP houdt de onderbrekingstoestand aan bij een door de joystick of spatiebalk veroorzaakte onderbreking. Nadat er een STRIG ON-commando is gedetecteerd, wordt er naar de door ON STRIG GOSUB-opgegeven subroutine gesprongen. Achter STRIG dient men de te controleren waarde op te geven, zie ON STRIG GOSUB.

SWAP **SWAP<variabele>,<variabele>**
Met dit commando kan men de waarden van twee variabelen verwisselen op voorwaarde dat de variabelen van hetzelfde type zijn.

```
10 A=100 : B=200
20 SWAP A,B
30 PRINT A,B
```

Het resultaat hiervan is dat de variabelen A en B in respectievelijk 200 en 100 worden veranderd.

TROFF **TROFF, TRON**
TRON Met het TRON-commando laat de computer tijdens de programma-uitvoering tevens het regelnummer zien van de regel waarmee hij bezig is. Zodoende kunnen fouten makkelijk worden opgespoord. Met het TROFF-commando kan men deze situatie weer ongedaan maken.

VPOKE **VPOKE <adres,gegeven>**
Dit commando werkt als POKE, maar het adres verwijst nu naar een adres in het videogedeelte van het RAM en kan variëren tussen 0 en 16383. De geheugeninhoud kan variëren tussen 0 en 255.

WAIT **WAIT <poortnummer,uitdrukking 1>[,<uitdrukking 2>]**
Met dit commando wordt het programma onderbroken en wordt er gewacht op gegevens uit de door het poortnummer aangegeven poort.

WIDTH **WIDTH (<aantal kolommen>)**
Met dit commando kan men het aantal kolommen van het tekstscherm instellen. Zonder nadere specificatie wordt het scherm in 40 kolommen ingedeeld. Bij MSX2 kan men tot 80 kolommen instellen.

```
WIDTH 30
```
Betekent dat er per regel maximaal 30 karakters kunnen worden afgebeeld. In tekst-mode 1 varieert de waarde tussen 1 en 40; in tekst-mode 2 tussen 1 en 32.

3 BASIC-functies

Bij de BASIC-functies worden af en toe twee notaties gehanteerd met de volgende betekenis:

X Een numerieke variabele of uitdrukking die men zelf kan invullen.
X$ Een stringvariabele of uitdrukking die men zelf kan bepalen.

ABS **ABS(<variabele>)**
Resulteert in de absolute waarde.

```
PRINT ABS(-1)
```
Levert de absolute waarde op van '–1', dit wordt dus '1'.

ASC **ASC(<string>)**
Resulteert in de ASCII-code van het eerste karakter van de string.

ATN **ATN(<variabele>)**
Geeft de arctangens in radialen. Het resultaat ligt tussen $-c/2$ en $c/2$.

BIN$ **BIN$(<variabele>)**
Zet een getal om in de binaire notatie.
Een voorbeeld:

```
PRINT BIN$(12)
```
Het resultaat hiervan is de string '1100'.

CDBL **CDBL(X)**
Zet X om in een getal met dubbele precisie.

CHR$ **CHR$(<variabele>)**

```
PRINT CHR$(65)
```
Resulteert in het karakter dat behoort bij de waarde (65): de letter A.

CINT **CINT(<variabele>)**
Rondt de waarde af op een geheel getal (een integer). De waarde van het opgegeven getal moet liggen tussen –32768 en 32767.

```
PRINT CINT(12.34)
```
Drukt de integer van 12.34 af: 12.

COS **COS(<variabele>)**

```
PRINT COS(1.369435)
```
Het getal moet in radialen worden opgegeven. De cosinus van 1.36935 wordt '.2'.

CSNG **CSNG(<variable>)**

```
PRINT CSNG(1.2345678)
```
Zet de waarde om in een getal met enkele precisie, hier dus '1.23457'.

CSRLIN **CSRLIN**
Geeft de Y-coördinaat (regelnummer) van de positie waarop de cursor zich bevindt, dus een getal van 1 t/m 23.

EOF **EOF(<filenummer>)**
Geeft aan of het einde van de desbetreffende file is bereikt, de waarde –1 betekent dat het einde van de file is bereikt.

ERL **ERL**
Geeft het regelnummer waarin een fout is opgetreden.

ERR **ERR**

Resulteert in een code die overeenkomt met de opgetreden fout.

ERROR **ERROR**

Plaatst de foutmelding die bij het opgegeven foutnummer hoort op het beeldscherm. Het foutnummer moet groter zijn dan 0 en kleiner dan 255.
Niet alle foutnummers tussen 0 en 255 worden door MSX gebruikt, zie ook appendix F. Let op bij het plaatsen van een cartridge; deze kan ook gebruik maken van vrije foutcodes. De vrije nummers kan men gebruiken om eigen foutboodschappen te genereren. Men kan ook een zelf opgestelde foutboodschap laten weergeven.

```
10 ON ERROR GOTO 300
20 INPUT "Getal= ",A
30 IF A>100 THEN ERROR 222
. . .
. . .
300 IF ERR=222 THEN PRINT "Getal
kleiner dan 100!"
310 RESUME 20
```

EXP **EXP(X)**

Resulteert in een macht van e. De waarde X mag niet groter zijn dan '145.062860858624', anders treedt er een overflow error op.

FIX **FIX(<variabele>)**

`PRINT FIX(7.2)`
Geeft het integer-deel van het getal, in dit geval '7'. Ook bij negatieve getallen wordt alleen het integerdeel genomen zonder af te ronden.

`PRINT FIX(-7.5)`
Hieruit volgt de waarde '–7'. De functie INT zou hier resulteren in '–8'.

FRE **FRE(<dummy>)**

```
PRINT FRE(0)
```
Toont u de vrije ruimte in het geheugen dat kan worden gebruikt voor BASIC-programma's.

```
PRINT FRE("")
```
Geeft u de geheugenruimte voor stringvariabelen in bytes. Dit geheugengebied kan met CLEAR worden aangepast.

HEX$ **HEX$(<variabele>)**

```
PRINT HEX$(43980.3)
```
Resulteert in een string die de hexadecimale waarde van het getal vertegenwoordigt. Het getal moet tussen –32768 en 65535 liggen en wordt eerst naar boven afgerond tot een geheel getal. De hexadecimale waarde in het voorbeeld wordt 'ABCD'.

INKEY$ **INKEY$**
Deze functie controleert of er een toets is ingedrukt en kent in het positieve geval het betreffende karakter toe aan een stringvariabele. Wordt er geen invoer vanaf het toetsenbord gedetecteerd dan resulteert dit in een lege string oftewel een nul-string.

```
500 T$=INKEY$:IF T$="" THEN 500
```

Zolang er geen toets wordt ingedrukt, zal T$ leeg zijn en de vergelijking stuurt het programma naar het begin van de regel om nogmaals te controleren of er een toets is ingedrukt.

INP **INP(<poortnummer>)**
Leest een byte vanuit een ingangspoort; het nummer is een integer tussen 0 en 255.

INPUT$ **INPUT$(X)**
Deze functie leest een string met een lengte van X karakters in vanaf het toetsenbord.

INPUT$(X,#<filenummer>)
Een string met een lengte van X karakters wordt uit een file gelezen. Alle karakters zijn toegestaan uitgezonderd CTRL/C.

INSTR **INSTR(<begin>,<string>,<string>)**
INSTR zoekt vanaf het begin van X$ of Y$ daarin voorkomt en geeft dan de desbetreffende positie op. Met het specificeren van het begin kan men eventueel een andere startpositie opgeven. Men krijgt de waarde nul als Y$ niet voorkomt en de waarde één als Y$ een lege string is.

INT **INT(X)**
Met deze functie kan het grootste gehele getal dat kleiner is dan of gelijk is aan X worden berekend.

LEFT$ **LEFT$(X$,A)**
Deze functie resulteert in een string die uit de voorste (linkse) A karakters van X$ bestaat. A mag hierbij niet groter zijn dan 255. De waarde nul voor A resulteert in een lege string.

LEN **LEN(X$)**
Berekent het aantal karakters waaruit X$ bestaat inclusief de spaties en de controlkarakters.

LOG **LOG(X)**
Berekent de natuurlijke logaritme van X, waarbij X groter moet zijn dan 0.

LPOS **LPOS(<dummy>)**

```
PRINT LPOS(0)
```

LPOS berekent de positie van de printer in de printbuffer, waarbij het niet noodzakelijk is dat deze overeenkomt met de positie van de printkop. Deze functie werkt alleen indien er een printer is aangesloten.

MID$ **MID$(A$,P,A)**

MID$(A$,4,2)
Herleidt een stringgedeelte van A$ (stel "ABC-DEF") met een lengte van A karakters (2) vanaf positie P (4), in dit geval resulteert dat in 'DE'. Zonder opgave van aantal karakters A neemt de computer alle karakters vanaf positie P.

```
MID$(A$,4)="YZ"
```
Vervangt vanaf de vierde positie de stringkarakters door 'YZ'. In dit voorbeeld en het vorige zal het resultaat zijn 'ABCYZF'.

MOD **MOD**
Geeft de rest na een deling.

```
90 Y=INT(10/3)
100 X=10MOD3
110 PRINT Y;" Rest ";X
```

De waarde Y is 3 en X zal 1 zijn.

OCT$ **OCT$(<variabele>)**

```
PRINT OCT$(20)
```
Resulteert in een string die de octale waarde van 20 vertegenwoordigt, dus '24'.

PAD **PAD(X)**
Met deze functie kan men de status van diverse accessoires vaststellen. Eerst dient men de juiste accessoires aan te duiden.

Waarde	*Accessoires*
0 t/m 3	aanraakpaneel op gamepoort 1
4 t/m 7	aanraakpaneel op gamepoort 2
8 t/m 11	lichtpen
12 t/m 15	muis of 'track ball' op gamepoort 1
16 t/m 19	muis of track ball op gamepoort 2

Informatie over het aanraakpaneel wordt op de volgende wijze verkregen:

Waarde X	*Betekenis*
0 of 4	0 = paneel niet aangeraakt –1 = paneel aangeraakt
1 of 5	X-coördinaat van aangeraakte plaats
2 of 6	Y-coördinaat van aangeraakte plaats
3 of 7	0 = schakelaar niet ingedrukt –1 = schakelaar ingedrukt

Informatie over de lichtpen wordt op de volgende wijze verkregen:

Waarde X	Betekenis
8	0 = pen is niet gereed –1 = pen is gereed
9	X-coördinaat, PAD(8) moet eerst –1 worden
10	Y-coördinaat, PAD(8) moet eerst –1 worden
11	0 = schakelaar niet ingedrukt –1 = schakelaar ingedrukt

Informatie over de muis of track ball wordt op de volgende wijze verkregen:

Waarde X	*Betekenis*
12 / 16	altijd –1, moet wél worden opgevraagd
13 / 17	X-coördinaat
14 / 18	Y-coördinaat
15 / 19	niet van toepassing

PDL

PDL(X)

Bepaalt de stand van de paddle (1 t/m 12). Bij oneven getallen neemt de computer aan dat de joystick via connector 1 is aangesloten, bij even getallen via connector 2. De functie werkt alleen bij een aangesloten paddle.

PEEK **PEEK(<geheugenplaats>)**
Deze functie leest de inhoud van geheugenplaatsen.

```
10 FOR I=3493 TO 3622
20 A=PEEK(I)
30 PRINT A,CHR$(A)
40 NEXT I
```

Dit programma zal een groot deel van de karakterset op het beeldscherm brengen.
Deze functie is het complement van de functie POKE.

PLAY **PLAY(X)**
Met deze functie kan men het gebruikte muziekkanaal herleiden: 0 = alle kanalen, 1 = kanaal 1, 2 = kanaal 2, 3 = kanaal 3. De waarde nul krijgt men als er geen van de kanalen in gebruik is, anders krijgt men de waarde –1.

POINT **POINT(X,Y)**
X = positie op de regel (0-255).
Y = regel (0-191).
Deze functie geeft een getal dat de kleur aangeeft van het punt op het scherm met de coördinaten X,Y. Deze functie werkt alleen in de grafische mode. In de modes 0 en 1 geeft deze functie de waarde '0'. Vallen de coördinaten buiten het beeldscherm dan verkrijgen we de waarde '1'.
Bij MSX2 zijn de waarden van X en Y afhankelijk van de ingestelde SCREEN-mode.

POS **POS(<dummy>)**

```
PRINT POS(0)
```

Resulteert bij de tekstmode in de positie (kolom) van de cursor op de regel, waarbij de meeste linkse positie nul is. Het getal '0' in de functie heeft geen speciale betekenis.

RIGHT$ **RIGHT$(X$,A)**
Resulteert in een string die uit de achterste (rechtse) A karakters van X$ bestaat.

RND **RND(X)**
Genereert een getal tussen nul en één uit een tabel die na RUN steeds hetzelfde is. Met X kan men een bepaald getal kiezen uit deze tabel.

X > 0	geeft het eerstvolgende getal
X = 0	geeft het laatste getal
X < 0	geeft een getal uit een nieuwe tabel
RND(-TIME)	resulteert iedere keer in andere getallen

SGN **SGN(X)**
Bepaalt het wiskundige teken van de waarde van X op de volgende manier:

X = positief	> SGN(X) = 1
X = 0	> SGN(X) = 0
X = negatief	> SGN(X) = –1

SGN(–123) resulteert dus in '–1'.

SIN **SIN(X)**
Resulteert in de sinus van X, waarbij X in radialen moet worden opgegeven.

SPACE$ **SPACE$(X)**
Resulteert in een string die uit X spaties bestaat. X mag niet groter zijn dan 255.

SPC **SPC(X)**
Resulteert in X spaties op het scherm of printer. Deze functie mag alleen in combinatie met een PRINT- of LPRINT-commando worden gebruikt. Ook hier mag X niet groter zijn dan 255.

SQR **SQR(X)**
Berekent de wortel uit X, waarbij X nooit negatief mag zijn.

STICK **STICK(X)**
Vertegenwoordigt de stand van de joystick (X=1 of 2). Geeft men X de waarde nul dan worden de cursortoetsen gelezen. De waarden die deze functie geeft, hebben de volgende betekenis.

0 neutraal
1 middenboven
2 rechtsboven
3 rechts
4 rechtsonder
5 middenonder
6 linksonder
7 links
8 linksboven

STR$ **STR$(X)**
Zet de numerieke waarde van X om in een string.

```
A$=STR$(987)
```
De variabele A$ vertegenwoordigt nu de string "987".

STRIG **STRIG(X)**
Bepaalt of de vuurknop van de joystick is ingedrukt met als resultaat: –1 = ja, 0 = nee.

X	*Betekenis*
0	Toetsenbord; spatiebalk
1 of 3	Joystick 1
2 of 4	Joystick 2

STRING$ **STRING$(A,K)**
Resulteert in een string die uit A identieke karakters bestaat met als ASCII-code K.

STRING$(A,K$)
Resulteert in een string die uit A identieke karakters bestaat. De karakters zijn alle hetzelfde als het eerste karakter van K$.

TAB **TAB(X)**
De cursor wordt verplaatst naar positie X (0-255) van dezelfde regel. Is de cursor al voorbij die positie dan gebeurt er niets. Deze functie werkt alleen in combinatie met een PRINT- of een LPRINT-commando.

TAN **TAN(X)**
Berekent de tangens van X, waarbij X in radialen moet worden opgegeven.

USR2 **USR2 (X)**
Hiermee wordt een machinetaalprogramma gestart, waarbij 2 het nummer van het betreffende machinetaalprogramma (toegewezen door DEF USR) en X het argument is. Als men het nummer (2) weglaat, gaat de computer van USR0 uit.
Indien men van het argument gebruik maakt, kan men de volgende notaties gebruiken:

$ voor stringvariabelen
% voor integervariabelen
! voor normale numerieke variabelen met enkele precisie
voor numerieke variabelen met dubbele precisie

Indien de machinetaalroutine een waarde aan het BASIC-programma teruggeeft, dient de variabele van de juiste soort te zijn ($, %, ! of #). Met de instructie CLEAR kan men geheugenruimte reserveren voor de machinetaal-routines.

VAL **VAL(X$)**
Resulteert in de numerieke waarde van X$. VAL is gelijk aan 0 als het eerste karakter van X$ geen +, –, & of cijfer is.

VARPTR **VARPTR(<variabele>)**
Aan de variabele moet een waarde zijn toegewezen voordat VARPTR het geheugenadres kan berekenen van de eerste byte waar de variabele is opgeslagen.

VARPTR(#<filenummer>)
Nu wordt het eerste adres van het file-controleblok van de opgegeven file berekend. Resulteert de waarde in een negatief getal dan dient men voor het berekenen van het juiste adres de waarde met 65536 te vermeerderen.

VPEEK **VPEEK(X)**
Met deze functie kan men de inhoud van een adres in het video-RAM-geheugen opzoeken. X vertegenwoordigt het adres en moet tussen 0 en 16383 liggen.

4 Disk BASIC-commando's

Er zijn veel MSX-BASIC-commando's die tevens betrekking hebben op het gebruik van de diskdrive, zoals LOAD, SAVE enz. De betekenis en het gebruik van die commando's kunt u terugvinden in het hoofdstuk BASIC-commando's. Hier behandelen we alleen de 'extra' en specifieke commando's voor gebruik bij de aansturing van één of meer diskdrives.
Bij de eerste MSX-versie zijn alleen de diskdrives A t/m D beschikbaar. MSX2 heeft de beschikking over de diskdrives A t/m F en een extra diskdrive die in het geheugen van de computer wordt geïnstalleerd, de 'memory-disk'. U dient er wel aan te denken dat het benoemen van een andere diskette binnen de aanhalingstekens direct vóór de filenaam moet gebeuren; bovendien moeten de opdrachten worden gescheiden door een dubbele punt (:).

APPEND **APPEND**
Wordt gebruikt om nieuwe data aan een bestaand sequentieel bestand toe te voegen.

```
OPEN "DATA" FOR APPEND AS #1
```
Het bestaande bestand 'DATA' wordt geopend voor het toevoegen van gegevens. Het wegschrijven van gegevens naar het DATA-bestand gebeurt verder op dezelfde manier als bij een sequentieel bestand.

BLOAD **BLOAD "<dev>:[<filenaam>]"[,<verschuiving>]**
Met dit commando kan een in machinetaal geschreven programma in het geheugen worden geladen.
Bij de screenmodes 5 t/m 8 (MSX2) wordt de inhoud van de actieve pagina geladen.

BSAVE

BSAVE "<dev>:[filenaam>]",<beginadres,eindadres>[,<beginadres voor start van het programma>][,S]

Dit commando schrijft een in machinetaal geschreven programma weg naar een cassetterecorder (CAS:) of een diskette (A:). Door het opgeven van een begin- en eindadres geven we het geheugengebied aan dat naar het externe medium moet worden weggeschreven.

Door het opnemen van ',S' achter het eindadres kan een deel van het videogeheugen worden weggeschreven. Dit kan echter alleen naar diskette.

Een BSAVE-commando kan er als volgt uitzien:

```
BSAVE "A:PROG.PIC",&HA100,&HA200,S
```

Begin- en eindadres geven het geheugengebied aan dat naar het externe medium moet worden gekopieerd. Eventueel kan men achter het eindadres een executie-adres opgeven. De letter S verwijst naar het videogeheugen. In de screenmodes 5 t/m 8 (MSX2) wordt de inhoud van de actieve pagina weggeschreven.

CALL

CALL FORMAT

Dit commando maakt de diskette gereed voor het ontvangen en opslaan van programma's en dergelijke onder MSX Disk BASIC.

Na dit commando zal de computer de volgende vraag stellen:

```
Drive name? (A,B)
```

Zijn er twee diskdrive-interfaces op uw computer aangesloten dan zal de vraag luiden:

```
Drive name? (A,B,C,D,)
```

Zodra het formatteren gereed is, meldt de computer dit met:

```
Format complete
Ok
```

Diskettes die u onder MSX-DOS heeft geformatteerd, kunt u zonder meer gebruiken voor Disk BASIC.

CALL MEMINI (waarde)
Met dit MSX2-commando reserveert men geheugenruimte voor een memory-disk. Neemt men een waarde op achter het commando dan wordt er een geheugengebied gereserveerd volgens de volgende formule:

Waarde = (G + 767)/256

Wil men een geheugenruimte voor de memory-disk hebben van 20K dan dient men een waarde van 81 op te geven. De naam van de memory-disk is 'MEM:'. De instructie CALL MEMINI mag natuurlijk maar éénmaal gebeuren na het aanzetten van de computer. De instructie is te vergelijken met de instructie CALL FORMAT voor gewone diskettes.

CALL MFILES
Dit MSX2-commando toont de files die zijn opgeslagen in de memory-disk. Er ontstaat een foutmelding 'disk offline' als de memory-disk niet is geïnitialiseerd.

CALL MKILL ("filenaam")
Dit commando (MSX2) wist de genoemde file uit de memory-disk.

CALL MNAME ("filenaam" AS "filenaam")
Dit MSX2-commando komt overeen met het Disk BASIC-commando NAME AS.

CALL SYSTEM
Specifiek MSX-DOS-commando dat alleen werkt bij MSX Disk BASIC.

Als men vanuit MSX-DOS met het commando 'BASIC' is overgegaan naar Disk BASIC, kan men zonder de computer uit te zetten of te 'resetten' met één van deze twee commando's terugkeren naar het MSX-DOS-besturingssysteem. De systeemdiskette voorzien van de files MSX-DOS.SYS en COMMAND.COM dient eerst te worden geplaatst in diskdrive A. De commando's werken niet als er is opgestart in BASIC of Disk BASIC.

CALL VERIFY
Met dit commando kan alle data die naar de diskette wordt weggeschreven, worden gecontroleerd.

CALL VERIFY ON
Nu wordt de verify-mode (controleren) ingeschakeld. Alle data die naar de diskette wordt gestuurd, zal nadat het op de diskette staat weer worden gelezen en gecontroleerd aan de hand van de inhoud van het computergeheugen.
Indien er bij het schrijven naar diskette een fout is opgetreden, bij het wegschrijven naar de diskette, verschijnt er een foutmelding op het beeldscherm. Na het opstarten van MSX Disk BASIC staat de verify-mode altijd op OFF.

COPY TO **COPY TO**
Dit commando dient voor het kopiëren van één of meer files van de ene naar de andere diskette. Dit commando werkt zowel bij één als meer diskdrives.

COPY "A:TESTPROG.BAS" TO "B:TEST.BAS"
Dit commando zal het programma TESTPROG.BAS kopiëren van diskette A: naar diskette B:, maar onder een andere naam, namelijk TEST.BAS.
De tweede naam achter 'TO' is een optie en kan worden vervangen door alléén een letter om daarmee een andere diskette of diskdrive aan te geven, waardoor het programma onder dezelfde naam naar de opgegeven diskette wordt gekopieerd.

Het COPY TO-commando kan tevens worden gebruikt om ASCII-files te kopiëren naar andere apparatuur.

COPY "TEKST.TXT" TO "CON"
Dit commando zal de inhoud van de file TEKST.TXT kopiëren naar CON (het CONsole, in dit geval het beeldscherm). U krijgt op het beeldscherm de tekst te zien waaruit deze file bestaat. In plaats van CON kunt u ook andere apparatuur invullen (AUX) en natuurlijk de printer, LST of PRN.

DSKO$ **DSKO$ <diskdrive-nummer>,<sectornummer>**
Met deze instructie is het mogelijk een deel van het geheugen naar een bepaalde diskettesector te schrijven. Het geheugengebied dat wordt weggeschreven, is vastgelegd in de geheugenplaatsen &HOF351 en &HOF352. De diskdrives zijn genummerd van 0 t/m 6, respectievelijk diskdrive A t/m F.

FIELD **FIELD**
Dit commando dient voor het creëren van ruimte in een buffer voor variabelen ten behoeve van een Random Access File.

```
FIELD #1,20 AS NAAM$,30 AS ADRES$
```
Hiermee worden de velden binnen een record van een Random Access File gedimensioneerd. Het eerste veld krijgt ruimte voor 20 karakters en de variabele heet 'NAAM$', het tweede veld heeft 30 karakters en de variabele heet ADRES$ enz.

FILES **FILES en LFILES**
LFILES Deze commando's tonen u alle 'files' (inhoudsopgave) van de diskette respectievelijk op het beeldscherm en via de printer.

GET **GET**
Dit commando dient voor het lezen van een record uit een Random Access File. Het record wordt opgeslagen in een random filebuffer.

GET# **GET#<filenummer>,<recordnummer>**
Nadat de file is geopend, kan men met dit commando een record lezen. Direct achter het GET-commando komt het 'logische filenummer' waar de Random Access File mee werd geopend. Achter de komma staat het recordnummer. Het in dit voorbeeld genoemde commando leest dus van file '#1' het twaalfde record.
Geeft men geen recordnummer op dan zal het GET-commando het eerstvolgende record lezen.
Na een GET-commando kan men met INPUT# en LINE INPUT# de gegevens uit de random filebuffer lezen.

KILL **KILL <filenaam>**
Commando voor het wissen van één of meer files van de diskette. Achter het commando wordt de te wissen file gespecificeerd.

```
KILL "TESTPROG.BAS"
```

De file TESTPROG.BAS wordt van de diskette gewist. Zolang er geen drive wordt opgegeven, zal dat altijd van de diskette in diskdrive A: zijn. Door vóór de filenaam de diskdrive te benoemen, wordt de file van de genoemde diskette gewist.
Gebruikt u dit commando terwijl er een sequentieel bestand of Random Access File is geopend dan zal de volgende mededeling op uw beeldscherm verschijnen:

```
File still open
Ok
```

Er kan in dit geval niets van de diskette worden gewist.

LEN **LEN**
Dit commando dient voor het dimensioneren van de recordlengte bij Random Access Files.

```
OPEN "A:ADRES.BES" AS #1 LEN=75
```

Met deze programmaregel wordt op diskette A: het

BEStand ADRES geopend onder het logische bestandsnummer #1 en krijgt ieder record een lengte van 75 karakters.
Hierna dient men de afzonderlijke velden binnen de records te dimensioneren met het commando FIELD.

LSET
RSET

LSET en RSET
Commando's om de gegevens vanuit het werkgeheugen naar de filebuffer te schrijven.

```
LSET N$=NAAM$
```
De variabele N$ wordt opgeslagen in de gereserveerde ruimte NAAM$ die zich in de filebuffer bevindt en wel aan de linkerzijde van de gereserveerde ruimte. Met andere woorden; de stringvariabele N$ wordt opgeslagen in NAAM$ als een reeks karakters die bestaat uit N$ aangevuld met spaties.

```
RSET N$=NAAM$
```
De variabele N$ wordt nu aan de rechterzijde opgeslagen in de gereserveerde ruimte en er staan spaties vóór N$. Beide commando's behoren tot bewerkingen die nodig zijn bij de Random Access Files.

NAME AS

NAME <oude filenaam> AS <nieuwe filenaam>
Verandert de filenaam (Engels reNAME) van een te specificeren file in een nieuwe filenaam.

```
NAME "TEKST.TXT" AS "TEKST1.TXT"
```
De bestaande file TEKST.TXT krijgt de nieuwe naam TEKST1.TXT. De inhoud van de file TEKST.TXT blijft ongewijzigd en op dezelfde plaats op de diskette staan; alleen in de inhoudsopgave wordt de naam veranderd. Staat de file die van naam moet veranderen niet op de default-diskette dan dient men voor de eerste filenaam de juiste diskette of diskdrive op te geven.

PUT

PUT #<filenummer>,<recordnummer>
Hiermee schrijven we de inhoud van de filebuffer naar de diskette.

```
PUT #1, R%
```
Nu worden alle gegevens die zich in de filebuffer bevinden naar de diskette geschreven en wel naar het bestand met nummer '#1' en naar 'record' nummer 'R%'. R% is in dit geval een variabele maar kan ook een normaal geheel getal zijn!

5 Disk BASIC-functies

Behalve de specifieke Disk BASIC-commando's kent Disk BASIC nog een aantal functies die na het oproepen een bewerking uitvoeren. Deze functies dienen hoofdzakelijk voor het lezen en schrijven in Random Access Files.

CVI
CVS
CVD

CVI, CVS en CVD
Deze functies zetten de inhoud van een 'numerieke string' om in een numerieke waarde. Deze numerieke string dient te zijn gemaakt met de functies MKI$, MKS$ of MKD$.

```
X=CVI (TEL$)
```
CVI verwerkt een 2-bytes string (TEL$) en zet die om naar een integer (geheel getal). Omdat het een 2-bytes string is, kan de integer niet groter zijn dan '32767' of kleiner dan '–32768'.

```
Y=CVS (A$)
```
CVS verwerkt een vierbytes-string (A$) tot een waarde met 'enkele' precisie. Uit deze bewerking kunnen getallen met maximaal zes cijfers voortkomen.

```
Z=CVD (B$)
```
CVD verwerkt een achtbytes-string (B$) tot een waarde met 'dubbele' precisie. De functie werkt hetzelfde als CVS, doch gebruikt acht bytes voor de opslag van het getal in plaats van vier bytes. Het getal bestaat maximaal uit 14 cijfers.

DSKF

```
PRINT DSKF (1)
```
Deze functie leest uit de inhoudsopgave de vrije opslagruimte die zich nog op de diskette bevindt.

Het getal tussen haakjes is het nummer van de diskdrive en correspondeert met:

0 default diskdrive
1 diskdrive A:
2 diskdrive B:
. . .
6 diskdrive F:

Het getal dat deze functie van de diskette leest en op het beeldscherm weergeeft, is uitgedrukt in kilobytes. Een waarde van 179 wil dus zeggen dat er 179 × 1024 = 183296 bytes beschikbaar zijn voor opslag van gegevens.

DSKI$ **DSKI$<drive>,<sectornummer>**
Bij het gebruik van deze functie wordt de aangegeven sector van de opgegeven diskdrive in het geheugen geladen, het gebruikte geheugengebied wordt aangegeven door de geheugenplaatsen 0F351 en &HOF352.

```
DSKI$(1,1)
```

Nu wordt van diskdrive 1 (A:) de eerste sector gelezen en in het geheugen geplaatst. Eventueel kan men met de instructie DSKO$ deze sector naar een andere diskdrive wegschrijven.
Het eerste cijfer staat voor de te lezen diskdrive:

0 default diskdrive
1 diskdrive A:
6 diskdrive F:

Het tweede getal geeft het sectornummer aan, afhankelijk van de gebruikte diskettes kunnen dat er maximaal 720 zijn.

EOF **EOF(<bestandsnummer>)**
Bepaalt of het einde van een bestand is bereikt en wel tijdens het inlezen van de gegevens. Bijvoorbeeld:

```
IF EOF(2) THEN CLOSE #2
```

LOC

LOC(<bestandsnummer>)
Deze functie geeft het laatst gelezen of geschreven recordnummer aan bij een Random Access File. Bij een sequentieel bestand zal de functie het totale aantal ingelezen of weggeschreven records aangeven.
Het nummer dient overeen te komen met het bestandsnummer van het geopende bestand. Is er nog niet gelezen of geschreven in een geopende Random Access File dan krijgt men de waarde 0.
Bij een sequentieel bestand krijgt men in dit geval de waarde 1, omdat bij het openen van een sequentieel bestand het eerste record altijd wordt gelezen. Probeert men deze functie zonder dat het genoemde bestand is geopend, dan resulteert dat in een foutmelding.

LOF

LOF(<bestandsnummer>)
Deze functie geeft de totale lengte van een bestand in bytes.
Denk eraan dat ieder gegeven in een sequentieel bestand wordt afgesloten met twee codes, tevens heeft het bestand nog een begincode. Het getal dat deze functie genereert, bestaat uit het aantal ingevoerde karakters plus één beginkarakter en twee karakters extra per ingevoerd 'item'.

MKI$
MKS$
MKD$

MKI$, MKS$ en MKD$
Deze functies zetten een numerieke waarde om in een numerieke string voor opslag in een Random Access File.
Het voordeel van deze functies is de ruimtebesparing, MKI$ beslaat slechts twee bytes, MKS$ vier bytes en MKD$ acht bytes.

```
TEL$=MKI$(A%)
```
MKI$ zet integers tussen de '32767' en '−32768' om in een numerieke string die alleen gelezen kan worden met CVI.

```
A$=MKS$(B!)
```
MKS$ zet een getal met enkele precisie om naar een numerieke string, hierbij wordt het getal indien nodig

afgerond op zes cijfers. Het afronden gebeurt op de normale manier: 123,4567 wordt 123,457 en 123,4562 wordt 123,456. De numerieke waarde kan worden gelezen met de CVS-functie.

B$=MKD$(C#)
MKD$ zet getallen met dubbele precisie om; ze kunnen met de CVD-functie worden teruggelezen.

6 DOS-commando's

BASIC

BASIC
Commando om MSX-DOS te verlaten en over te gaan naar MSX Disk BASIC.
Indien men achter het commando BASIC een naam van een BASIC-programma opneemt, zal dat BASIC-programma automatisch worden geladen en gestart.

```
BASIC TESTPROG.BAS
```
Nadat BASIC is geactiveerd, zal het programma TESTPROG.BAS automatisch worden geladen en gestart. De programmanaam moet volledig worden opgegeven, dus inclusief de extensie; het gebruik van de wildcards * en ? is bij dit commando niet toegestaan.
Denk eraan dat dit commando de geheugenindeling van de computer verandert, daar die indeling voor MSX Disk BASIC en MSX-DOS verschillend is.

COPY

COPY <bron> <bestemming>
Commando voor het kopiëren van één of meer files van de ene naar de andere diskette. Dit commando werkt zowel bij één als twee diskdrives.

```
COPY TESTPROG.BAS TEST.BAS
```
Dit commando zal het programma TESTPROG.BAS kopiëren op dezelfde diskette maar onder de naam TEST.BAS.

```
COPY TESTPROG.BAS B:
```
De naam achter het te kopiëren programma is een optie en kan worden vervangen door een letter om

daarmee een andere diskette of diskdrive aan te geven, waardoor het programma onder dezelfde naam naar de opgegeven diskette wordt gekopieerd.

```
COPY C:TESTPROG.BAS
```

Het programma TESTPROG.BAS wordt nu vanaf diskette C gekopieerd naar de default diskette gekopieerd. Indien men van een bepaalde diskette naar een andere diskette wil kopiëren, dient men beide diskdrives op te geven.

```
COPY A:TEST1.TXT + B:TEST2.TXT A:TO-
TAAL.TXT
```

De files TEST1.TXT van diskette A en TEST2.TXT van diskette B worden samengevoegd tot één nieuwe file en weggeschreven naar diskette A onder de naam TOTAAL.TXT.
Hiermee zijn bijvoorbeeld verschillende tekstfiles samen te voegen tot één grote tekstfile. In het genoemde voorbeeld zal TEST2.TXT achter TEST1.TXT worden opgenomen in de nieuwe file.
Normaal gesproken worden alleen ASCII-files aan elkaar gekoppeld: tekstfiles gemaakt met een tekstverwerkend programma. Ook BASIC-programma's kunnen als ASCII-files worden opgeslagen, ze moeten dan worden geSAVEd met de A-optie. Het einde van een ASCII-file wordt altijd gekenmerkt door een '<CONTROL>-Z' ('Z) code; dat is de MSX-karaktercode 26.
Binaire files en programma's hebben geen 'Z-code aan het einde. De lengte van een dergelijke file wordt voor het kopiëren uit de inhoudsopgave gelezen. Om dat aan te geven gebruikt men de '/B'-switch.

```
COPY/B TESTA.COM + TESTB.COM
```

De lengtes van de files TESTA.COM en TESTB.COM worden nu uit de inhoudsopgave gelezen. In dit voorbeeld wordt de file TESTB.COM achter TESTA.COM geplaatst en heet de nieuwe file TESTA.COM. De originele file TESTA.COM bestaat dus niet meer en is vervangen door een samenvoeging van beide files.

```
COPY TESTA.COM/B TESTB.COM/A
```
De lengte van de file TESTA.COM wordt uit de inhoudsopgave gelezen. Na het kopiëren wordt er een 'Z-code toegevoegd aan de nieuwe file TESTB.COM.
De '/A'-switch zal één 'Z-code toevoegen aan de nieuwe file. Het toevoegen van een tweede 'Z-code aan een file kan alleen bij tekstfiles.

```
COPY *.TXT COMBI.TEX
```
Alle files voorzien van de extensie .TXT worden eerst samengevoegd en dan weggeschreven naar een nieuwe file COMBI.TEX. De files die worden gekopieerd, zullen natuurlijk ook als afzonderlijke files op de diskette blijven staan.

```
COPY TEKST?1.TXT TEKST?1.TTT
```
Met dit commando worden alle files die beginnen met 'TEKST' met daarachter nog één karakter en eindigend op 1.TXT gekopieerd naar nieuwe files met dezelfde naam, maar met een andere extension.
Gebruik bij het kopiëren nooit één of meer vraagtekens aan het einde van een filenaam. Alle filenamen worden opgeslagen in de vorm van acht karakters, zo zal bijvoorbeeld de naam TEST3.BAS opgeslagen zijn als 'TEST3 .BAS'.
Tussen de drie en de punt bevinden zich (in de directory) nog drie spaties. Een spatie voldoet aan de kenmerken van een willekeurig karakter. Voor het vraagteken als wildcard maakt het dus niets uit of er een spatie of een ander karakter op die bewuste plaats staat.

```
COPY TEKST.TXT LST
```
Dit zal de tekstfile naar de printer sturen en alle informatie op papier zetten. In plaats van de aanduiding 'LST' kan men ook 'PRN' gebruiken. Denk eraan dat alleen ASCII-files op papier kunnen worden afgedrukt.

```
COPY CON TEKST.TES
```
Dit commando zal alles dat u via het toetsenbord invoert, overbrengen naar de default diskette onder de naam TEKST.TES. Het wegschrijven gebeurt pas na het afsluiten van de tekst met een 'Z-code daar alleen ASCII-files met dit commando rechtstreeks op diskette kunnen worden weggeschreven.

DATE

DATE [<datumcode>]
Dit commando dient voor het controleren en wijzigen van de datum. De computer zal de laatst ingevoerde datum opnemen in de inhoudsopgave bij iedere nieuwe file die u op diskette zet.

```
DATE 03-31-1986
```
Hiermee wordt de datum direct opgeslagen in het geheugen zonder dat u de vorige datum op het beeldscherm krijgt met de vraag naar de nieuwe datum.

DEL

DEL <filenaam>
Dit commando dient voor het wissen (Eng. DELete) van één of meer files van de diskette. Achter het commando wordt de te wissen file gespecificeerd.

```
DEL B:*.TXT
```
Deze opdracht zal alle files met de extension .TXT die zich op diskette B bevinden, wissen. Om alle files ongeacht hun naam of extension van een diskette te wissen, kunnen we voor de extension ook een 'asterisk' opnemen. Om vergissing te voorkomen, vraagt de computer voor alle zekerheid of deze 'totale' wisopdracht inderdaad moet worden uitgevoerd.

DIR

DIR <opties>
Dit commando toont u de directory (inhoudsopgave) van de diskette. U heeft de keuze uit een compleet overzicht inclusief de filegegevens of uit een verkorte inhoudsopgave die alleen de filenamen laat zien.

```
DIR /W
```
Dit commando toont alleen de filenaam met zijn ex-

tensie. De filenamen worden op het beeldscherm in twee kolommen weergegeven (alleen bij MODE 29 of hoger).

```
DIR /P
```

Dit commando toont maximaal 23 files op het beeldscherm. Zijn er meer files op de diskette aanwezig dan krijgt u de volgende mededeling:

```
Strike a key when ready . . .
```

Nadat u een willekeurige toets heeft ingedrukt, zal de computer het vervolg van de inhoudsopgave lezen en op uw beeldscherm weergeven.

ERASE **ERASE <filenaam>**
Dit commando is gelijk aan DEL en wist informatie van de diskette. Zie voor verdere bijzonderheden het commando DEL.

FORMAT **FORMAT [<drive>]**
Dit commando 'formatteert' de diskette voor het ontvangen en opslaan van MSX-files.

MODE **MODE <kolombreedte>**
Hiermee kunt u de kolombreedte van het beeldscherm instellen. Afhankelijk van de kolombreedte wordt het beeldscherm ingesteld op de tekstmode of de grafische mode. Deze schermmode wordt bepaald door het aantal karakters (kolommen) per regel. De grafische mode verkrijgt u door een karakterbreedte van 32 of minder. Een karakterbreedte van 33 tot 40 zet het beeldscherm in de tekstmode.
In de grafische mode kunt u de grafische karakterset van uw toetsenbord goed en volledig weergeven op het beeldscherm. In de tekstmode zullen de grafische karakters niet in hun geheel worden weergegeven.

REN **REN <oude filenaam> <nieuwe filenaam>**
Verandert de filenaam (Engels REName) van een te specificeren file in een nieuwe filenaam. Staat de file die van naam moet veranderen niet op de default

diskette dan dient men vóór de eerste filenaam de juiste diskette of diskdrive op te geven.
Geeft men voor de nieuwe file een filenaam op die overeenkomt met een reeds op de diskette aanwezige file dan zal er een foutmelding op het beeldscherm verschijnen.

RENAME **RENAME <oude filenaam> <nieuwe filenaam>**
Gelijk aan REN.

TIME **TIME [<tijdcode>]**
Dit commando dient voor het controleren en wijzigen van de tijd. Het systeem zal de tijd opslaan in de inhoudsopgave bij iedere nieuwe file die u op diskette zet. Na het TIME-commando kunt u direct de nieuwe tijd invoeren.
Voert u geen tijd in dan zal de computer de juiste tijd weergeven (indien die eerder werd ingevoerd) en vragen om een nieuwe tijd. De tijd wordt uitsluitend met cijfers ingevoerd. Tussen de uren, minuten en de eventuele seconden dient een dubbele punt (':') te staan.

H = 00-24 in uren
M = 00-59 in minuten
S = 00-59 in seconden
C = 00-99 in honderdsten van seconden

De seconden en honderdsten van een seconde zijn optioneel en mogen worden weggelaten. Tussen de seconden en honderdsten van een seconde mag geen dubbele punt staan, maar moet men een enkele punt ('.') gebruiken.
Indien uw computer niet is voorzien van een ingebouwde klok zal dit commando niet functioneren en een foutmelding tot gevolg hebben.

TYPE **TYPE <filenaam>**
Geeft de inhoud van een gespecificeerde file weer op het beeldscherm. Eventuele tabsprongen worden omgezet in spaties tot de eerste standaard-tabkolom

van het beeldscherm. Deze beeldschermtabs staan om de acht kolommen.
Het weergeven van een binaire file zal gepaard gaan met het weergeven van alle eventuele control-karakters uit die binaire file en zijn onleesbaar op het beeldscherm.

VERIFY

VERIFY
Met dit commando kan alle data die naar de diskette wordt weggeschreven, worden gecontroleerd.

`VERIFY ON`
Nu wordt de verify-mode (controleren) ingeschakeld. Alle data die naar de diskette wordt gestuurd, zal nadat het op de diskette staat, weer worden gelezen en gecontroleerd aan de hand van de inhoud van de computer.
Indien er een fout is opgetreden bij het wegschrijven naar de diskette, verschijnt er een foutmelding op het beeldscherm.

`VERIFY OFF`
Na het opstarten van MSX-DOS staat de verify-mode altijd op OFF.

7 BATCH-commando's

Alle MSX-DOS-commando's kunnen worden gebruikt in een BATCH-programma. Daarnaast heeft MSX-DOS nog twee speciale commando's die alléén in BATCH-programma's kunnen worden toegepast.

PAUSE

PAUSE [<tekst>]
Commando dat kan worden gebruikt om een pauze in te lassen, bijvoorbeeld ten behoeve van het verwisselen van een diskette.

```
PAUSE Plaats diskette B
```

Na het commando PAUSE kan men een boodschap opnemen die tijdens het uitvoeren van het BATCH-programma moet worden getoond. Als scheidingstekens kunt u tussen PAUSE en de mededeling gebruik maken van de spatie, de tab-sprong of de komma. Na ieder PAUSE-commando verschijnt tijdens het uitvoeren van het BATCH-programma eerst de opgenomen mededeling en daarna verschijnt automatisch de volgende regel op het beeldscherm:

```
Strike a key when ready . . .
```

Iedere toets kan worden gebruikt om het BATCH-programma verder te laten gaan, uitgezonderd <CONTROL>-C.
Indien u <CONTROL>-C indrukt, kunt u het BATCH-programma onderbreken.
U kunt het PAUSE-commando ook gebruiken om een BATCH-programma op bepaalde punten vroegtijdig af te breken.

REM

REM [<commentaar>]
Het commando REM (REMark) maakt het mogelijk om commentaar en opmerkingen op te nemen die worden weergegeven op het beeldscherm tijdens het uitvoeren van het BATCH-programma.

Appendix A Systeemvariabelen

MSX-BASIC kent een aantal systeemvariabelen, die een gereserveerde naam hebben. Ondanks het feit dat het systeem er zelf waarden aan toekent, kan men ze desgewenst 'overschrijven'.

BASE **BASE(X)**

Bevat het eerste adres van de zogenaamde Video Display Processor (VDP). X heeft de volgende betekenis:

0 naamtabel	tekstmode 1
1 geen betekenis	
2 patroontabel	tekstmode 1
3 geen betekenis	
4 geen betekenis	
5 naamtabel	tekstmode 2
6 kleurtabel	tekstmode 2
7 patroontabel	tekstmode 2
8 sprite-attribuuttabel	tekstmode 2
9 sprite-patroontabel	tekstmode 2
10 naamtabel	grafische mode 1
11 kleurtabel	grafische mode 1
12 patroontabel	grafische mode 1
13 sprite-attribuuttabel	grafische mode 1
14 sprite-patroontabel	grafische mode 1
15 naamtabel	grafische mode 2
16 kleurtabel	grafische mode 2
17 patroontabel	grafische mode 2
18 sprite-attribuuttabel	grafische mode 2
19 sprite-patroontabel	grafische mode 2

Deze adressen zijn alleen te overschrijven met een waarde. Let hierbij op dat een verandering voor bij-

voorbeeld screenmode 2 ook geldt voor de screenmodes 4 t/m 8.
MSX2 maakt tevens gebruik van de adressen 20 t/m 44:

20 naamtabel	grafische mode 3
21 kleurtabel	grafische mode 3
22 patroontabel	grafische mode 3
23 sprite-attribuuttabel	grafische mode 3
24 sprite-patroontabel	grafische mode 3
25 naamtabel	grafische mode 4
26 kleurtabel	grafische mode 4
27 patroontabel	grafische mode 4
28 sprite-attribuuttabel	grafische mode 4
29 sprite-patroontabel	grafische mode 4
30 naamtabel	grafische mode 5
31 kleurtabel	grafische mode 5
32 patroontabel	grafische mode 5
33 sprite-attribuuttabel	grafische mode 5
34 sprite-patroontabel	grafische mode 5
35 naamtabel	grafische mode 6
36 kleurtabel	grafische mode 6
37 patroontabel	grafische mode 6
38 sprite-attribuuttabel	grafische mode 6
39 sprite-patroontabel	grafische mode 6
40 naamtabel	grafische mode 7
41 kleurtabel	grafische mode 7
42 patroontabel	grafische mode 7
43 sprite-attribuuttabel	grafische mode 7
44 sprite-patroontabel	grafische mode 7

SPRITE$ **SPRITE$(<nummer>)=<string>**
Dit commando dient om de vorm van een sprite te definiëren.
Nummer is het sprite-nummer, dat afhankelijk van het sprite-formaat maximaal 63 of 255 kan zijn. De vorm van de sprite bepalen we door middel van een string. Dit wordt meestal gedaan aan de hand van een binaire of hexadecimale definitie.

```
10 SCREEN 2,0
20 B$=""
30 FOR I=1 TO 8
40 READ A$:B$=B$+CHR$(VAL("&B"+A$))
50 NEXT I
60 SPRITE$(0)=B$
70 DATA 00011000, 00111100,
   01111110, 11111111, 00100100,
   00100100, 01000010, 10000001
```

TIME

TIME

De variabele TIME bevat de tijd van de interne klok waarmee de MSX-computer is uitgerust. Deze variabele is vooral geschikt om echt willekeurige getallen in combinatie met de RND-functie te genereren. Tijdens het schrijven naar of het lezen van de cassetterecorder krijgt de variabele TIME geen interrupt van de VDP, hierdoor verandert TIME niet tijdens de werking van de cassetterecorder.

VDP

VDP (X)

Deze variabele bevat de gegevens van de Video Display Processor. De waarde van X ligt tussen 0 en 8, waarbij de inhoud van het achtste register niet mag worden veranderd.
Bij MSX2 kan de waarde van X liggen tussen 0 en 47, de waarden 25 t/m 32 kunnen niet worden gebruikt.

Videogeheugen betreffende de intensiteit

De intensiteit bij MSX2 van de verschillende kleuren is opgeslagen in het videogeheugen. Iedere screenmode heeft zijn eigen tabel die zich bevindt op de volgende adressen:

SCREEN	*adres*
0 (40 kol)	&H0400 - &H0420
0 (80 kol)	&H0F00 - &H0F20
1	&H2020 - &H2040
2	&H2020 - &H2040
3	&H2020 - &H2040
4	&H2020 - &H2040
5	&H7680 - &H76A0
6	&H7680 - &H76A0
7	&HFA80 - &HFAA0
8	&HFA80 - &HFAA0

Daar men bij MSX2 in de screenmodes 5 t/m 8 meer pagina's tegelijk kan instellen met het commando SET PAGE, heeft men voor iedere pagina de ruimte voor een intensiteitstabel. Deze tabellen bevinden zich op het volgende adres:

screenmodes 5 en 6	paginanummer X &H08000 + &H7680
screenmodes 7 en 8	paginanummer X &H10000 + &HFA80

Appendix B
Speciale tekens en uitdrukkingen

Bewerkingstekens (operators)

Hierna volgt een overzicht van de tekens die in rekenkundige bewerkingen worden gebruikt. Het laatste cijfer geeft de prioriteitswaarde aan: uitdrukkingen worden volgens deze prioriteitswaarde uitgewerkt (van het laagste tot het hoogste cijfer). Bij gelijke prioriteitswaarde wordt de uitdrukking van links naar rechts uitgewerkt.

Teken	*Betekenis*	*Voorbeeld*	*Prioriteit*
+	optellen	3+5	6
−	aftrekken	5−3	6
*	vermenigvuldigen	5*3	2
/	delen	5/3	3
^	machtsverheffen	2^3	1
−	verandert teken	−3	4
MOD	rest bij integerdeling	12 MOD 3	5

Tekens in logische uitdrukkingen

De volgende symbolen kunnen in wiskundige uitdrukkingen voorkomen, waarbij wordt nagegaan of ze al of niet waar zijn (bijv. na de term IF).

Teken	*Betekenis*	*Voorbeeld*
=	is gelijk aan	IF A=B THEN...
<	kleiner dan	IF A<B THEN...
>	groter dan	IF A>B THEN...
<> of ><	ongelijk aan	IF A<>B THEN...
<= of =>	kleiner dan of gelijk aan	IF A<=B THEN...
>= of =>	groter dan of gelijk aan	IF A>=B THEN...

De volgende logische operators zetten de waarde eerst om in een binaire waarde en volgen dan de regels van de Boolese algebra:

Teken	*Betekenis*
AND	om twee vergelijkingen op te nemen
EQV	om twee waarden te vergelijken
IMP	impliceert twee waarden
NOT	om het tegengestelde van een vergelijking te testen
OR	om twee vergelijkingen te vergelijken
XOR	vergelijkt twee waarden volgens de exclusive or methode

Appendix C
Control- en kleurcodes

Controlcodes onder BASIC

Een aantal toetsen hebben in combinatie met de CTRL-toets een specifieke functie. Als men bijvoorbeeld met ingedrukte CTRL-toets ook de L-toets indrukt, zal het scherm worden gewist. We schrijven dit als CTRL L. CTRL U betekent dan: op U drukken terwijl de CTRL-toets is ingedrukt. In het nu volgende overzicht zijn alle mogelijkheden opgenomen:

CTRL A	De nu volgende toets roept een symbool op uit de alternatieve symbolenreeks
CTRL B	Verplaatst cursor naar het vorige woord
CTRL C	Stopt AUTO-commando
CTRL E	Alle symbolen vanaf de cursor worden gewist
CTRL F	Verplaatst de cursor naar het volgende woord
CTRL G	BEEP
CTRL H	Komt met de BS-toets overeen
CTRL I	Komt met TAB overeen: naar volgende tabulatorstop
CTRL J	Verplaatst de cursor naar het begin van de volgende regel
CTRL K	Verplaatst de cursor naar de hoek linksboven
CTRL L	Komt met CLS overeen
CTRL M	Komt met de RETURN-toets overeen
CTRL N	Verplaatst de cursor naar het einde van de regel (d.w.z. achter het laatste karakter op die regel)
CTRL R	Komt met de INS-toets overeen
CTRL U	Wist de regel
CTRL \	Komt overeen met cursortoets →
CTRL]	Komt overeen met cursortoets ←
CTRL ^	Komt overeen met cursortoets ↑
CTRL –	Komt overeen met cursortoets ↓
CTRL [	Komt overeen met cursortoets ↑

Controlcodes bij MSX-DOS-editing

↑	Toont de inhoud van het commandogeheugen
↓	Verwijdert de commandoregel die op het beeldscherm werd getoond
→	Toont het eerstvolgende karakter uit het commandogeheugen
←	Verwijdert het laatst weergegeven karakter
CLS<karakter>	Springt over de karakters in het commandogeheugen tot het opgegeven karakter
DEL	Springt steeds over één karakter in het commandogeheugen
HOME	De commandoregel wordt in het commandogeheugen opgeslagen zonder dat deze wordt uitgevoerd
INS	Insert mode; maakt het mogelijk om een karakter tussen te voegen in de reeds bestaande commandoregel
SELECT<karakter>	Toont de commandoregel tot het opgegeven karakter
CTRL –	Toont de inhoud van het commandogeheugen
ESC	Wist de weergegeven commandoregel van het beeldscherm
CTRL ^	idem
CTRL U	idem
CTRL [	idem
CTRL \	Toont het eerstvolgende karakter uit het commandogeheugen
CTRL H	Wist het laatst weergegeven karakter van het beeldscherm
CTRL]	idem
CTRL K	De commandoregel wordt in het commandogeheugen opgeslagen zonder dat deze wordt uitgevoerd
CTRL X	Brengt alles uit het commandogeheugen over naar het beeldscherm tot aan het gespecificeerde karakter
CTRL L	Springt in het commandogeheugen over de karakters heen tot aan het gespecificeerde karakter

CTRL R	Schakelt de insert mode in, het commandogeheugen maakt ruimte voor het tussenvoegen van nieuwe tekst

Overige controlcodes bij MSX-DOS

CTRL C	Onderbreekt commando's en BATCH-bestanden
CTRL J	Brengt de cursor naar het begin van de volgende beeldschermregel
CTRL N	Zet de printer uit
CTRL P	Zet de printer aan
CTRL S	Onderbreekt tijdelijk de weergave op het beeldscherm (pauze)

Kleurcodes

Met de volgende codes kan men de kleuren bepalen.

Code	*Kleur*
0	transparant
1	zwart
2	groen
3	lichtgroen
4	donkerblauw
5	lichtblauw
6	donkerrood
7	cyaan
8	rood
9	lichtrood
10	donkergeel
11	lichtgeel
12	donkergroen
13	magenta
14	grijs
15	wit

Zie ook het MSX2-commando COLOR voor het maken of wijzigen van de kleuren.

Appendix D RS232C-interface

Algemeen

In de computerwereld wordt communicatie een steeds belangrijker begrip. Het betekent dat informatie van de ene naar de andere plaats kan worden verzonden – bijvoorbeeld tussen twee computers die met elkaar in een directe verbinding staan, of via een modem en telefoonlijnen. Om programma's en gegevensbestanden over te zenden, is er voor de MSX-computer een RS232C-interface ontwikkeld. Deze interface zorgt ervoor dat de over te zenden informatie in een speciaal daartoe geschikt formaat wordt omgezet. De RS232C is een serieel interface, omdat de enen en nullen waarmee alle gegevens worden overgezonden, na elkaar (dus sequentieel) worden gezet. Om dit proces naar behoren te laten verlopen, zijn er wél een aantal control-signalen vereist.
We gaan ervan uit dat er een MSX-computer met een andere MSX-computer communiceert, maar in feite zou die andere computer ook een willekeurig ander apparaat kunnen zijn dat is voorzien van een standaard-RS232C-aansluiting (bijvoorbeeld een modem om via de telefoon met een andere computer te communiceren). De interface bestaat uit een cartridge zonder kabel. Op deze cartridge bevindt zich een standaard-aansluiting (een 25-pens D-connector). Er zijn in de handel kabels verkrijgbaar die de interfaces van de twee computers kunnen verbinden.

We voeren op computer 1 bijvoorbeeld het volgende programmaatje in:

```
10 PRINT "Test RS232C-interface"
20 PRINT "Voor de MSX"
```

We geven nu het volgende commando:

```
SAVE "COM:"
```

We schrijven dit programma weg onder de naam "COM", waarbij COM de RS232C-interface aanduidt. Als we nu op computer 2 intoetsen:

```
LOAD "COM:"
```

dan zal na het indrukken van de RETURN-toets van beide computers het programma worden overgestuurd. Als we na het SAVE-commando echter te lang wachten met het invoeren van het LOAD-commando, zal er een foutmelding verschijnen (Device I/O Error). De reden is dat met het intoetsen van het SAVE-commando te kennen werd gegeven dat computer 1 gegevens wil verzenden en computer 1 kijkt dan of computer 2 gereed is (d.m.v. het LOAD-commando) om die gegevens te ontvangen. Als dit te lang duurt, verschijnt de bewuste foutmelding.
In plaats van LOAD "COM:" hadden we ook RUN "COM:" kunnen geven, waardoor het programma direct wordt uitgevoerd.

Het volgende voorbeeld laat zien hoe een file wordt overgestuurd van computer 1 naar computer 2.
Op computer 1 voeren we het volgende programma in:

```
10 OPEN "COM:" FOR OUTPUT AS #1
20 INPUT A$
30 PRINT #1, A$
40 CLOSE #1
```

Op computer 2 voeren het we volgende in:

```
10 OPEN "COM:" FOR INPUT AS #1
20 INPUT #1, A$
30 PRINT A$
40 CLOSE #1
```

Als we nu op computer 1 na het starten van het programma een tekst intoetsen, verschijnt (als we het programma van computer 2 ook het commando RUN hebben gegeven) op het scherm van computer 2 dezelfde tekst op het beeldscherm

Een ander voorbeeld. Als we op beide computers het commando
CALL COMTERM
invoeren, dan wordt alles wat op computer 1 wordt ingevoerd ook zichtbaar bij computer 2 en andersom wat op computer 2 wordt ingetoetst, wordt zichtbaar op computer 1. In dit geval hebben we van beide computers een terminal gemaakt en kan er informatie worden uitgewisseld.

Bij het aansluiten op een ander apparaat dan een MSX-computer (bijvoorbeeld een computer van een ander merk), dienen we allerlei zaken in te stellen zoals:

- de snelheid van het verzenden;
- de pariteitscontrole;
- de lengte van de bytes;
- het aantal stopbits.

Voor een beginnende gebruiker geldt dan ook dat het communiceren via de RS232C niet eenvoudig zal verlopen als men niet over de nodige technische kennis beschikt.

Het gebruik van de RS232C voor gegevensbestanden

Vanuit BASIC kan de RS232-interface worden gebruikt als invoer- of uitvoermedium. Alle commando's die voor gegevensbestanden op cassette kunnen worden gebruikt, zijn ook beschikbaar voor het werken met RS232C. Dit zijn:

OPEN	PRINT# USING
PRINT#	INPUT#
CLOSE	LINE INPUT#

Bij het OPEN-commando dienen we "COM<n>" te gebruiken. Het nummer <n> is alleen nodig indien u meer dan één RS232-interface op uw computer heeft aangesloten.
Een bestandsnaam hoeft u bij het openen van een bestand niet op te geven.

```
OPEN "COM0:" AS #1
OPEN "COM:" FOR OUTPUT AS #2
```

In het eerste voorbeeld ziet u dat er geen mode (FOR INPUT of

FOR OUTPUT) wordt opgegeven. Hierdoor kunt u het aldus geopende bestand zowel voor lezen als voor schrijven gebruiken. Op deze manier kunt u met andere computers communiceren.

Voor programma-opslag heeft u de beschikking over de volgende commando's:

LOAD "COM<n>:"
SAVE "COM<n>:"
MERGE "COM<n>:"
RUN "COM<n>:"

De devicenaam is weer "COM<n>:"; voor het getal <n> geldt wat reeds is besproken. Een bestandsnaam hoeft niet te worden gespecificeerd. Programma's worden altijd in ASCII-code verstuurd.

Instellingen van de RS232C-interface

Voordat twee apparaten via de RS232C-interface met elkaar kunnen communiceren, moeten beide interfaces op dezelfde manier zijn ingesteld. Bij de instellingen gaat het om baudsnelheden, pariteit, handshake, aantal databits en dergelijke.
Alle instellingsgrootheden kunnen worden ingesteld met het commando CALL COMINI. De syntax luidt:

CALL COMINI <string>,<baud Rx>,<baud Tx>,<tijd>

Verklaring

baud Rx:	baudsnelheid (bits/sec) waarmee de RS232C-interface karakters ontvangt. Standaard ingesteld op 1200 baud.
baud Tx:	baudsnelheid (bits/sec) waarmee de RS232C-interface karakters verstuurt. Standaard ingesteld op 1200 baud.
Tijd:	tijd (in sec) die de computer zal wachten op de handshake alvorens een fout te melden. Tijd = 0 betekent dat de computer eeuwig blijft wachten. Standaard ingesteld op 0.

String: het formaat luidt:
"<n>:<db><pa><sb>
<x><hs><lf><slf><s>"

Betekenis

n: kanaalnummer: nummer van de gebruikte interface. Alleen nodig als er met meer dan één interface wordt gewerkt.
Standaard ingesteld op 0.

db: aantal databits waarmee de gegevens worden verstuurd. In te vullen "5", "7" of "8".
Standaard ingesteld op "8".

pa: pariteit.
"E" = even pariteit (Even).
"O" = oneven pariteit (Odd).
"I" = kijk niet naar pariteit (Ignore).
"N" = geen pariteit (No).
Standaard ingesteld op "E".

sb: aantal stopbits, een soort pauze tussen het versturen van bytes.
"1" = 1 bit.
"2" = 1,5 bits.
"3" = 2 bits.
Standaard ingesteld op "1".

x: XON/XOFF-besturing.
"X" = wél besturen.
"N" = niet besturen.
Standaard ingesteld op "X".

hs: CTS/RTS-handshake.
"H" = wel handshake.
"N" = geen handshake.
Standaard ingesteld op "H".

lf: invoegen van een linefeed na een carriage-return.
"A" = wel invoegen.
"N" = niet invoegen.
Standaard ingesteld op "N".

slf: sturen van een linefeed-teken na elke carriage-return.
"A" = stuur geen linefeed.
"N" = stuur wel een linefeed.
Standaard ingesteld op "N".

ss: shift-in/shift-out besturing.
"S" = wel besturen.
"N" = niet besturen.
Standaard ingesteld op "N".

Instellingen die niet worden gespecificeerd, nemen de standaardwaarde aan. De enige uitzondering op deze regel is <baud, Tx>. Als deze niet is gespecificeerd, krijgt hij de waarde van <baud Rx>.

Gebruik van de RS232C voor programmabesturing

Door middel van CALL-commando's kunt u in een BASIC-programma automatisch een subroutine laten uitvoeren zodra er via de RS232C-interface een teken wordt ontvangen. De werking van deze commando's is gelijk aan de werking van ON...GOSUB's. De commando's zijn:

CALL COMON "<n>:"	gelijk aan INTERVAL ON.
CALL COMOFF "<n>:"	gelijk aan KEY OFF.
CALL COMSTOP "<n>:"	gelijk aan STRIG STOP.
CALL COM <n>:,GOSUB <regel>	gelijk aan ON INTERVAL GOSUB.

<n> is weer het nummer van de gewenste interface en kan worden weggelaten als er maar één interface is aangesloten.

Control-commando's

CALL COMBREAK "<n>",<code>
Met dit commando kunnen we 'break-karakters' versturen. De opgegeven <code> wordt nu gebruikt als teken om de communicatie te verbreken (getal tussen 3-32767). Dit commando kan alleen actief zijn als de RS232C al is geopend.

CALL COMDTR "<n>",<0 of niet–0>
Als de uitdrukking <0 of niet–0> gelijk is aan 'niet–0', wordt de interface tijdelijk uitgeschakeld. Dit commando werkt alleen als de RS232C is geopend.

CALL COMSTAT "<n>",<naam variabele>
Als er een fout optreedt, wordt er aan de variabele een waarde gegeven waaruit blijkt wat er eventueel fout is gegaan. De volgende tabel geeft de verschillende mogelijkheden. In een aantal gevallen zijn de Engelse termen overgenomen.

Bit nr.	*Betekenis*
0	Carrier Detect
	0 – niet
	1 – wel
1	Ring Indicator
	0 – niet
	1 – wel
2	Break Detect
	0 – niet
	1 – wel
3	Data Set Ready
	0 – niet
	1 – wel
4	Niet gebruikt
5	Niet gebruikt
6	Timer/counter output-2
	0 – timer/counter wordt uitgezet
	1 – timer/counter wordt toegekend
7	Clear to Send (klaar om te zenden)
	0 – niet
	1 – wel
8	Niet gebruikt
9	Niet gebruikt
10	Control break-toets is ingedrukt (BRONBT)
11	Parity error
	0 – niet
	1 – wel
12	Overrun error
	0 – niet
	1 – wel

13 Framing error
0 – niet
1 – wel
14 Time out error (TMENBT)
0 – niet
1 – wel
15 Buffer overflow error
0 – geen buffer overflow
1 – wel buffer overflow

CALL COMTERM "<n>"
Met <n> wordt eventueel het kanaalnummer aangegeven. Standaard ingesteld op 0. Dit commando zet de computer in de terminalmode, zie ook het begin van deze appendix.
Met de functietoets F6 wordt het mogelijk control-karakters over te sturen.
Met de functietoets F7 schakelt men tussen half en full duplex.
Met de functietoets F8 wordt de printer als een soort echo in- of uitgeschakeld.

Technische gegevens van de aansluiting

De aansluitingen van de connector hebben de volgende betekenis:

Pen Betekenis
1 Chassis ground (GND)
2 Transmit Data (TXD)
3 Receive Data (RXD)
4 Request to Send (RTS)
5 Clear to Send (CTS)
6 Data Set Ready (DSR)
7 Signal Ground (GND)
8 Carrier Detect (CD)
20 Data Terminal Ready (DTR)
22 Ring Indicator (RI)

Aansluitingen van de cartridge-steker

Pen	*Naam*	*Betekenis*
1	CS1	Selectie van ROM-adressen 4000-7FFF
2	CS2	Selectie van ROM-adressen 8000-7BFFF
3	CS12	Selectie van ROM-adressen 4000-BFFF (256K ROM)
4	SLTSL	Slotselectie
6	RFSH	Refresh-cyclus
7	WAIT	WAIT-request-signaal van CPU
8	INT	Interrupt-request van CPU
9	M1	Fetch-cyclus van CPU
10	BUSOIR	Stuurt externe databus-buffer
11	IORQ	I/O request-signaal
12	MERQ	Memory (geheugen) request
13	WR	Write (schrijven)
14	RD	Read (lezen)
15	RESET	Algemene reset
17-32	A0-A15	Adreslijnen
33-40	D0-D7	Datalijnen
41	GND	Aarde
42	CLOCK	3.579545 MHz-klok
43	GND	Aarde
44, 46	SW1, SW2	Protectie in- en uitnemen
45, 47	+ 5 V	Voeding
48	+ 12 V	Voeding
49	SOUNDIN	Input geluid
50	− 12 V	Voeding

Appendix E
Foutmeldingen

Buiten het weergeven van verschillende mededelingen op het beeldscherm zal MSX-DOS ook fouten tijdens het uitvoeren registreren en daar melding van maken. Voordat MSX-DOS die melding op het beeldscherm weergeeft, probeert het besturingssysteem het commando alsnog uit te voeren, en wel drie keer. Mocht de fout dan nog steeds bestaan dan zal MSX-DOS met de volgende melding komen:

```
<1> error <2> drive <3>
Abort, Retry, Ignore?
```

1 Op deze plaats staat een mededeling die de mogelijke fout van de diskette of diskdrive aangeeft. U kunt de volgende meldingen verwachten:

Write protect	De diskette kan niet worden beschreven.
Not ready	Als er geen diskette in de drive is geplaatst.
Disk	Treedt o.a. op als de diskette niet is te lezen of te beschrijven, bijvoorbeeld indien de diskette nog niet is geformatteerd.

2 Deze melding zegt iets over uitvoering van het commando op het moment dat de fout optrad. Er zijn twee mogelijkheden:

reading	Tijdens het lezen van de diskette.
writing	Tijdens het schrijven naar diskette.

3 Ten slotte krijgt u de melding bij welke diskette of diskdrive de fout is opgetreden.

A, B, C of D	één van de diskdrives.

Na deze foutmelding vraagt MSX-DOS wat er met die fout moet gebeuren, het intoetsen van de eerste letter voert het gevraagde commando uit:

A	Abort (stoppen). Het programma wordt onderbroken en de standaard-cursor komt terug op het beeldscherm.
R	Retry (opnieuw proberen). MSX-DOS zal, nadat u de fout zo mogelijk heeft hersteld, proberen het commando nogmaals uit te voeren.
I	Ignore (negeren). MSX-DOS zal een eventuele slechte sector van de diskette waar de fout is ontstaan, negeren en doen alsof de fout niet is opgetreden. Hiermee kan wel informatie verloren gaan.

Over het algemeen zal men eerst de 'R' opgeven om te kijken of het een toevallige éénmalige fout was. Treedt de fout weer op dan toetsen we de 'A' in om te stoppen.

De volgende foutmelding treedt op indien er geen diskette is geplaatst en de computer een file van de diskette wil lezen:

```
Not ready error reading drive A
Abort, Retry, Ignore?
```

Welke letter u ook intypt, de computer komt steeds met deze foutmelding terug tot u een diskette heeft geplaatst. Ook na het plaatsen van de diskette maakt het niet uit welke letter u intypt, de computer maakt eerst het commando af waar hij mee bezig was.

Een foutmelding die ook kan ontstaan tijdens het lezen van of het schrijven naar een diskette is:

```
Bad FAT
```

Deze melding geeft aan dat de gegevens in de 'File Allocation Table' verwijzen naar een niet-bestaande sector. Het kan zijn dat de diskette verkeerd of in het geheel niet is geformatteerd. Treedt deze fout op dan is de informatie op de diskette niet meer te

achterhalen en dus onbruikbaar. De enige manier om de diskette weer te kunnen gebruiken, is deze opnieuw te formatteren. Daarbij gaat alle eventueel op de diskette aanwezige informatie verloren.

Behalve deze twee foutmeldingen kent MSX-DOS nog een aantal foutmeldingen en mededelingen die regelmatig kunnen verschijnen.

Aborted
Deze uitdrukking verschijnt na het indrukken van ^C op momenten dat de computer tijdens een BATCH-programma vraagt om het indrukken van een willekeurige toets, behalve op die plaatsen waar men het BATCH-programma mag en kan onderbreken.

Are you sure (Y/N)?
Indien men alle files van de diskette wil wissen, vraagt de computer u eerst om een bevestiging van het ingevoerde commando.

Bad command or file name
Het opgegeven commando bestaat niet, of het opgegeven woord refereert niet aan een bestaande file.

Boot error
Press any key for retry
Deze fout treedt op als men teruggaat naar MSX-DOS zonder de diskette die is voorzien van de twee systeemfiles.

Disk I/O error
Deze foutmelding treedt alleen op als er tijdens het VERIFY ON-commando een fout wordt geconstateerd.

Drive name? (A,B)
Na het FORMAT-commando vraagt de computer welke diskette moet worden geformatteerd.

.. File copied
Zodra er één of meer files zijn gekopieerd, zal de computer daar melding van maken. Op de plaats van de puntjes komt het aantal gekopieerde files te staan.

File not found
Deze foutmelding zal optreden als we de inhoudsopgave van een lege diskette opvragen of als we na een commando een niet op diskette staande filenaam opgeven.

Format complete
Zodra het formatteren is voltooid, komt deze mededeling op het beeldscherm.

Insert diskette for drive A:
and strike a key when ready
De computer geeft steeds aan wanneer er van diskette moet worden gewisseld. Na het indrukken van een willekeurige toets zal het commando worden afgemaakt. Het gebruik van ^C heeft hier geen invloed.

Insert disk with batch file
and strike any key when ready
Indien u de diskette met het in uitvoering zijnde BATCH-programma verwijdert, zal deze foutmelding optreden. Het intoetsen van ^C zal het BATCH-programma op dit punt onderbreken.

Invalid date
Enter new date:
Deze mededeling verschijnt indien men tijdens het opstarten of na het commando DATE een verkeerde datum opgeeft.

Invalid drive specification
Het opgeven van de benaming van een niet-aangesloten diskdrive resulteert in deze foutmelding.

Invalid parameter
Deze foutmelding treedt op indien men een verkeerd getal opgeeft, bijvoorbeeld het commando MODE 82.

Invalid time
Enter new time:
Deze mededeling verschijnt indien men tijdens het opgeven van de tijd een fout maakt. Werkt niet bij alle MSX-computers.

Insufficient disk space
Indien de diskette onvoldoende ruimte heeft om de file weg te schrijven. De diskette is dan dus vol.

Rename error
Indien men een niet op diskette staande file van naam wil veranderen, treedt deze foutmelding op. Ook als de nieuwe naam reeds als filenaam op de diskette voorkomt, treedt deze melding op.

Strike a key when ready . . .
Na een PAUSE-commando in een BATCH-programma komt deze mededeling. U kunt dan met ^C het BATCH-programma onderbreken.

Terminate batch file (Y/N)?
Deze boodschap verschijnt indien een BATCH-programma wordt onderbroken met <CONTROL>-C. Een 'Y' zal de BATCH-file doen stoppen, terwijl bij 'N' de BATCH-file begint met het uitvoeren van het eerstvolgende commando.

Foutmeldingen en codes bij BASIC en Disk BASIC

Bad allocation table **err=60**
Als de diskette nog niet is geformatteerd en men de inhoudsopgave wil opvragen.

Bad drive name **err=62**
Een niet-aangesloten diskdrive wordt gespecificeerd in één van de commando's.

Bad file mode **err=61**
Bij het gebruik van de commando's PUT, GET of LOF bij een sequentiële file. Openen van een file zonder 'FOR OUTPUT', 'FOR INPUT' of 'FOR APPEND'.

Bad file name **err=56**
Als men een filenaam opgeeft direct achter het MSX-DOS-commando BASIC (dus zonder spatie), verschijnt in Disk BASIC deze foutmelding.

Bad file number **err-52**
Een commando verwijst naar een niet-geopende file of valt buiten het aantal geïnitialiseerde files.

Bad sector number **err=63**
Indien men een verkeerd sectornummer opgeeft bij het rechtstreeks lezen van of schrijven naar de diskette.

Can't continue **err=17**
Men tracht een programma voort te zetten na het beëindigen van dat programma door een opgetreden fout, of na het wijzigen van het programma, of na het einde van het programma.

Device I/O error **err=19**
Bij het optreden van een onherstelbare fout tijdens in- of uitvoer van gegevens naar een extern medium.

Direct statement in file **err=57**
Bij het laden van een ASCII-file die voorzien is van een direct commando. Het laden wordt afgebroken.

Disk full **err=66**
Er is geen ruimte genoeg op de diskette om een programma of een file weg te schrijven.

Disk I/O error **err=69**
Een fout bij het lezen of schrijven van/naar de diskette. De computer kan hierop blijven hangen. Deze foutmelding treedt ook op bij niet-geformatteerde diskettes.

Disk offline **err=70**
Er bevindt zich geen diskette in de gespecificeerde diskdrive.

Disk write protected **err=68**
Indien men wil wegschrijven naar een diskette die beveiligd is tegen schrijven.

Division by zero **err=11**
Indien men een rekenkundige functie uitvoert waarbij men tracht door 0 te delen.

Field overflow **err=50**
Deze melding treedt op als men meer bytes wil gebruiken in een 'FIELD' dan in de lengte van een RECORD zijn gespecificeerd.

File already exist **err=65**
De file die wordt gespecificeerd bestaat reeds op de diskette.

File already open **err=54**
Indien een file al eerder werd geopend, of als men een open file probeert te wissen met het KILL-commando.

File not found **err=53**
De opgegeven file staat niet op de default diskette.

File not OPEN **err=59**
Bij het lezen van of het schrijven naar een niet-geopende file.

File still open **err=64**
De file is nog niet afgesloten met 'CLOSE'.

Illegal direct **err=12**
Bij het rechtstreeks opgeven van een indirect commando.

Illegal function call **err=5**
Er wordt een verkeerd gegeven gebruikt voor een rekenkundige berekening of bij een string-functie.

Input past end **err=55**
Er wordt om een INPUT gevraagd terwijl alle data al is gelezen. Het gebruik van het commando EOF (einde file) voorkomt deze foutmelding.

Internal error **err=51**
Deze melding zal zelden op uw beeldscherm verschijnen, daar ze alleen ontstaat als er een defect aan uw computer is.

Line buffer overflow **err=25**
Een ingevoerde regel bevat te veel karakters.

Missing operand **err=24**
Een bewerking bevat een bewerkingsteken zonder dat er een waarde volgt.

NEXT without FOR **err=1**
Er wordt een NEXT-commando gevonden zonder een FOR-instructie, of er wordt een variabele gebruikt in de FOR-instructie die niet overeenkomt met de variabele bij het NEXT-commando.

No RESUME **err=21**
Een foutafhandelingsroutine bevat geen RESUME-commando terwijl de computer die wel nodig heeft.

Out of DATA **err=4**
Er wordt een READ-commando gegeven terwijl alle DATA al is gelezen, of er is geen DATA aanwezig.

Out of memory **err=7**
Een programma dat deze foutmelding veroorzaakt, is te lang. De fout kunt u zoeken in te veel FOR-loops, te veel GOSUB's, te veel functies of te veel variabelen.

Out of string space **err=14**
Geen ruimte meer in de string-werkruimte van het geheugen.

Overflow **err=6**
Een getal is te groot voor het type variabele of een functie of commando kan er niet mee overweg.

Redimensioned array **err=10**
Men tracht een reeds gedimensioneerde array nogmaals te dimensioneren met het DIM-commando.

Rename across disk **err=71**
Bij het veranderen van een filenaam van de ene diskdrive naar de andere.

RESUME without error **err=22**
Er is een RESUME-commando opgenomen buiten de foutafhandelingsroutine.

Return without GOSUB **err=3**
Het programma komt een RETURN-opdracht tegen zonder eerst een GOSUB-commando gekregen te hebben.

Sequential I/O only **err=58**
Het GET- of PUT-commando wordt gebruikt bij een sequentiële file.

String formula too complex **err=16**
Een stringuitdrukking is te moeilijk voor de computer om er mee te kunnen werken.

String too long **err=15**
Een string wordt groter dan 255 karakters.

Subscript out of range **err=9**
Een index of een array-element valt buiten de gedimensioneerde gegevens van het DIM-commando.

Syntax error **err=2**
Fout in de spelling van een commando of een fout bij het gebruik van bepaalde leestekens.

Too many files **err=67**
Er staan 255 files (het maximum) op diskette en men probeert een programma te SAVEn of een file te openen.

Type mismatch **err=13**
Een stringvariabele wordt aangegeven als een numerieke variabele of omgekeerd.

Undefined line number **err=8**
Er wordt verwezen naar een niet bestaand regelnummer met een GOTO-, IF/THEN/ELSE- of een DELETE-commando.

Undefined user function **err=18**
Een functie (FN) wordt opgeroepen zonder dat deze is gedefinieerd met het DEF FN-commando.

Verify error **err=20**
Bij het controleren van gegevens treedt er een verschil op, o.a. bij gebruik van het CLOAD?-commando.

De volgende foutnummers geven geen foutmelding op het scherm:

Unprintable error **err=23**
Er is een onbekende fout opgetreden.

Unprintable error **err=26 t/m 49**
Foutcodes voor later gebruik.

Unprintable error **err=72 t/m 255**
Geen bestaande foutcodes, kunnen door de gebruiker worden gedefinieerd voor eigen foutmeldingen.

Het is bijna onmogelijk om de vele commando's en functies waarover MSX-computers beschikken uit het hoofd te kennen. Vandaar dit zakboekje. Het bevat een overzichtelijke opsomming van zo goed als alle belangrijke gegevens. De volgende zaken komen onder meer aan de orde:

- Een beschrijving van het BASIC-systeem, de instructies, de functies en alle systeemboodschappen.
- Een overzicht van de MSX-DOS-commando's
- Alle foutmeldingen
- De ASCII-tabel

Al deze informatie maakt dit zakboekje onmisbaar voor zowel beginnende als gevorderde MSX-bezitters.

ISBN 90 201 2002 6