MSX

A.C.J. Groeneveld

QUICK DISK

handboek voor iedereen

uw **MSX** *computer de baas*

Beste lezer,

Dit MSX boek is afkomstig uit de nalatenschap van Wammes Witkop - hoofdredacteur MSX Computer Magazine 1985 - 1992.

MSX QUICK DISK
handboek voor iedereen

uw MSX computer de baas

De konstruktietekeningen van de Quick Disk zijn beschikbaar gesteld door AVT in Den Haag.

MSX QUICK DISK

handboek voor iedereen

uw **MSX** *computer de baas*

A.C.J. Groeneveld

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

CIP-GEGEVENS KONINKLIJKE BIBLIOTHEEK, DEN HAAG

Groeneveld, A.C.J.

MSX quick disk : handboek voor iedereen : uw MSX computer de baas / A.C.J. Groeneveld. – Oosterend : Stark-Texel
ISBN 90 6398 254 2
SISO 365.3 UDC 681.3.06
Trefw.: MSX (computer) / microcomputers ; programmeren.

. . . . . .

mei 1985
ISBN 90 6398 254 2

INHOUD

1 INLEIDING

In het grote MSX handboek werden de MSX computer en het MSX basic uitgebreid behandeld. De bezitter van een eenvoudige MSX computer vindt in dit grote handboek dan ook alle geheimen van de MSX computer uitgebreid beschreven.

Dit MSX QUICK DISK handboek is bedoeld als een uitbreiding op het eerste, grote handboek en is geschreven voor de bezitter van een MSX computer met een QUICK DISK schijveneenheid, tezamen met het grote MSX handboek of eventueel de verkorte versie van dit handboek die door veel importeurs standaard bij de computer wordt geleverd.

In dit handboek wordt voornamelijk ingegaan op het MSX QUICK DISK BASIC (dat we verder het QD basic zullen noemen). De beginnende programmeur dient dan ook eerst een redelijke ervaring te hebben opgebouwd met behulp van het grote handboek alvorens dit handboek ter hand te nemen. Daarbij is het verstandig om het grote handboek altijd bij de hand te houden om nog eens het een en ander te kunnen nazoeken.

Tezamen met het grote MSX handboek vormt dit QD handboek een zeer duidelijke en complete handleiding voor uw MSX computer met QUICK DISK eenheid.

Indien u nog niet in het bezit bent van het 'grote' handboek, dan kunt u dit alles onthullende, meer dan 400 pagina's tellende, nederlandse handboek bestellen bij uw computerboekhandel. Uw boekhandelaar heeft aan het ISBN nummer 90 6398 100 7 genoeg om het voor u te kunnen bestellen. Eventueel kunt u dit boek natuurlijk ook rechtstreeks bij de uitgever bestellen (telefoon 02223 - 661).

Ik hoop dat dit MSX QD handboek alle eerder door STARK-TEXEL uitgegeven MSX boeken in hun enorme succes mag volgen.

mei 1985,
A.C.J. Groeneveld.

In het grote MSX handboek werd reeds de opbouw van het geheugen van een MSX computer behandeld. We zagen ondermeer dat we het geheugen van een MSX computer kunnen opdelen in vier geheugenbanken. In bank 0 en 1 zetelt het MSX basic in ROM (read only memory, niet uitwisbaar geheugen). In bank 2 en 3 bevindt zich dan maximaal 32 kilobyte RAM (random access memory, vrij toegankelijk geheugen) waarin de MSX computergebruiker bijvoorbeeld zijn programma kan koderen.

Het MSX QD basic brengt in de struktuur van het geheugen in zoverre een wijziging, dat een extra ROM geheugen (van 8 kilobytes) als het ware 'achter geheugenbank 1 is geplaatst. Dit ROM geheugen bevindt zich in de aansluitstekker van uw QD eenheid en wordt bij elk voor de QUICK DISK geldend kommando geraadpleegd.

Wanneer de QD eenheid verder niet wordt gebruikt, is de aanwezigheid van het QD basic dan ook verder niet merkbaar, behalve op twee punten:

- bij het opstarten van het MSX systeem meldt de computer even dat het speciale QD basic aktief is.

- indien in de QD eenheid een floppy aanwezig is met daarop het programma "AUTOEXEC", dan wordt bij het inschakelen van het systeem dit programma automatisch geladen en uitgevoerd.

In het grote MSX handboek werd reeds de opbouw van een MSX computer behandeld. Eén van de onderdelen van de MSX computer kan bestaan uit een schijveneenheid.

De QD eenheid is een speciale vorm van zo'n schijveneenheid. Hij werd ontworpen als een opslagmedium dat zich ongeveer tussen een schijveneenheid en een cassetterecorder in bevindt.

Omdat de QD eenheid niet tot de standaard MSX uitrusting wordt gerekend, werd zijn aansturing als zodanig ook niet in het grote handboek behandeld.

3.1 Permanente gegevensopslag

Het is bij een computer onontbeerlijk dat gegevens voor langere tijd kunnen worden bewaard. Een groot en ingewikkeld programma willen we niet elke avond opnieuw intikken, maar slechts éénmaal invoeren en vervolgens vastleggen. Standaard kan dit vastleggen van gegevens in MSX basic gebeuren op een cassetteband met behulp van een cassetterecorder.

Elke MSX-programmeur stuit al snel tegen twee grote bezwaren van deze opslagmethode, namelijk:

- De opslag duurt vrij lang. 32 kilobytes (32768 tekens) op cassetteband vastleggen duurt ongeveer vijf minuten. Dit lijkt in eerste instantie snel, maar blijkt al vlug een groot bezwaar te zijn.

- De opslag is niet erg betrouwbaar. Vooral wanneer de cassetterecorder niet in topconditie is of verkeerd bandmateriaal wordt gebruikt, blijken gegevens tijdens opslag te kunnen worden verminkt. Het is dan ook noodzakelijk om uitstekend bandmateriaal aan te schaffen, de cassetterecorder regelmatig schoon te maken en af te stellen en de vastgelegde gegevens altijd te controleren.

Ondanks deze grote bezwaren is de cassetteband het meest voor de hand liggende opslagmedium voor hobby-computers; een cassetterecorder is relatief erg goedkoop, terwijl ook het bandmateriaal relatief zeer goedkoop is.

Wanneer snelheid bij het vastleggen en ophalen van gegevens een rol gaat spelen, wordt de cassetterecorder al snel een onbruikbaar randapparaat. Er dient dan naar een sneller opslagmedium te worden uitgekeken: de magneetschijf.

MSX QD basic is een uitbreiding van het standaard MSX basic en bezit een tiental extra instrukties voor het besturen van de QD eenheid.

In de volgende paragrafen wordt eerst het algemene principe van de QD eenheid uitgelegd. Daarna gaan we in op de logische indeling van een QD magneetschijfje. Vervolgens wordt in hoofdstuk 4 beschreven hoe de QD eenheid dient te worden geïnstalleerd. Daarna, in hoofdstuk 5, wordt het MSX QD basic met betrekking tot de besturing van de QD eenheid uitvoerig behandeld

3.2 Het principe van de QD eenheid

De QD eenheid gaat uit van het principe dat gegevens worden gelezen van of geschreven naar een vrij snel ronddraaiende schijf. Deze schijf, van kunststof gemaakt, is bedekt met een magnetisch geprepareerde laag die vergelijkbaar is met de laag magnetisch materiaal die op een cassetteband is aangebracht.

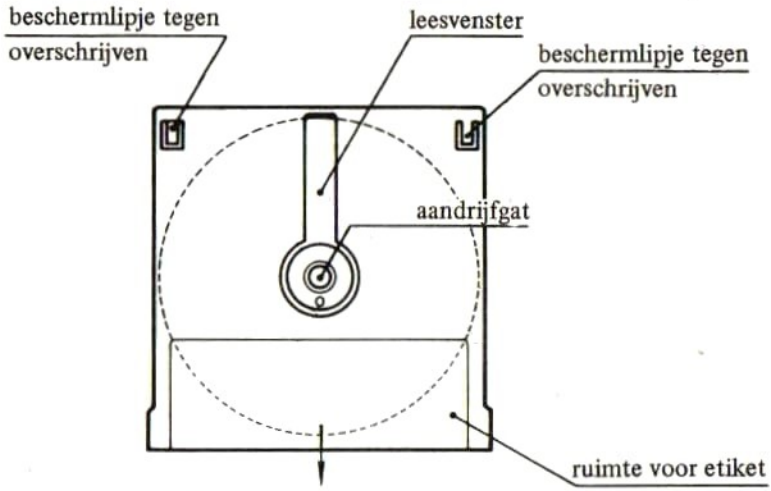
De QD schijf is 7½ cm in doorsnede (2,8 inch). Op een QD schijfje kunnen we de volgende dingen onderscheiden:

- het leesvenster. Door dit venster kijken we recht tegen het daadwerkelijke schijfje aan. De afleeskop van de QD eenheid leest door dit venster de diverse gegevens in. Het spreekt vanzelf dat de voor dit leesvenster blootliggende schijf niet met de vingers mag worden aangeraakt.

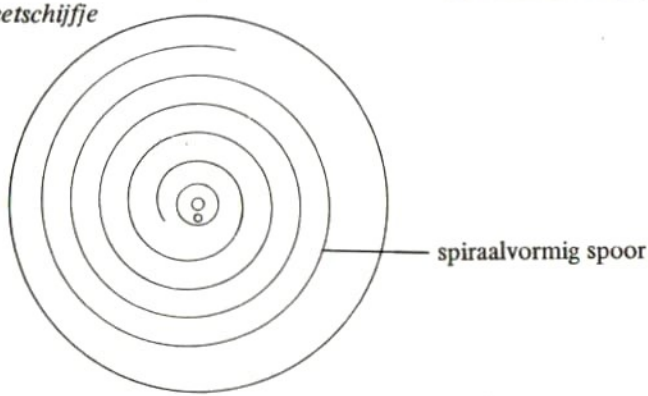
- het aandrijfgat. In dit gat valt bij plaatsing van de schijf in de QD eenheid de aandrijfas die het schijfje ongeveer vijf maal per seconde rond draait.

- write protect lipjes. Door het betreffende lipje voorzichtig uit het vakje te breken, beschermt men de bijbehorende kant tegen overschrijven van de gegevens door nieuwe gegevens. Door een write protect lipje weg te breken kan men belangrijke informatie tegen ondeskundig of abusievelijk gebruik beschermen. Indien later het betreffende schijfje toch weer dient te worden beschreven, kan men de bescherming opheffen door het ontstane gaatje

De QD floppy



Het magneetschijfje



Het spiraalspoor

Uitgerold is het spiraalvormig spoor ongeveer 6 meter lang. Over deze 6 meter worden ruim 60000 tekens opgeslagen. Per mm spoor worden dus ongeveer 10 tekens vastgelegd.

met plakband aan beide zijden af te dekken.

- ruimte voor etiket. In deze ruimte kan precies het bijbehorende etiket worden geplakt. Schrijf op dit etiket de identifikatie van het betreffende schijfje VOORDAT het etiket op de schijf wordt geplakt. Wanneer het etiket toch dient te worden beschreven NADAT het is opgeplakt, doe dat ter voorkoming van schade aan het schijfje dan met een zachte viltstift en zeker NIET met balpen of potlood.

Wanneer we het in het hard plastic omhulsel opgenomen schijfje aan een nader onderzoek onderwerpen, dan zien we dat op dit schijfje slechts één spoor is opgenomen, ongeveer zoals een groef op een grammofoonplaat. Dit spiraalvormige spoor draait zo'n 40 maal rond het aandrijfgat en heeft bij elkaar een lengte van ongeveer 6 meter. Op één zo'n spiraal kan 64 kilobyte aan informatie worden opgenomen (65536 tekens). Per mm spoorlengte worden dus ongeveer 10 tekens opgenomen. Omdat een QD floppy tweezijdig kan worden beschreven, is de totale opslagcapaciteit van zo'n floppy 128 kilobytes.

Het concept van een spiraalspoor is in de wereld van schijveneenheden zeer ongebruikelijk. Meestal past men een hoeveelheid concentrische sporen toe (cirkelvormige sporen die precies om elkaar liggen).

Door een spiraalvormige spoorvorm toe te passen, konden relatief goedkope technieken worden toegepast waardoor de prijs van een QD eenheid ver onder de prijs van een conventionele schijveneenheid kon blijven.

De toepassing van een spiraalvormig spoor heeft ook nadelen. Hierop komen we verderop in dit handboek terug.

Om gegevens van de QD floppy te kunnen schrijven en weer terug te kunnen lezen, is een lees- schrijfkop noodzakelijk, net zoals een opname en een weergavekop in een cassetterecorder noodzakelijk zijn. In het geval van de QUICK DISK zit de gecombineerde lees- schrijfkop op een beweegbare arm gemonteerd. Tijdens het draaien van de schijf beweegt deze arm zich in ongeveer 8 seconden van buiten naar binnen waardoor het spiraalvormig spoor op de schijf wordt gevolgd.

Om gegevens op de QD floppy te kunnen schrijven, zijn de volgende akties door de QD eenheid te ondernemen:

– de schijf dient aan het draaien te worden gebracht.
– de lees- schrijfkop dient het spiraalvormige spoor te volgen.
– op de eerste vrije plaats in het spoor dienen de betreffende gegevens te worden geschreven.

Om gegevens daarna weer te kunnen lezen, zijn de volgende akties door de QD eenheid te ondernemen:

– de schijf dient weer aan het draaien te worden gebracht.
– de lees- schrijfkop dient het spiraal vormige spoor weer te volgen.
– op de juiste plaats dient het inlezen te worden aangevangen.

Indien op de QD floppy gegevens worden geschreven, worden deze automatisch ter controle een keer opnieuw teruggelezen en vergeleken met de zo juist geschreven gegevens (de zogenaamde read after write controle). Hierdoor duurt het schrijven van gegevens ongeveer twee maal zo lang als het lezen van dezelfde gegevens.

Enkele voordelen van de QD eenheid ten opzichte van de cassetterecorder vallen onmiddellijk op:

– er behoeft geen cassetterecorder meer op opnemen of afspelen te worden gezet; de QD eenheid zorgt zelf voor het overschakelen naar opnemen en afspelen.

– de gehele QD schijf wordt in ongeveer 8 seconden afgewerkt. Lange door- of terugspoeltijden komen hierdoor te vervallen.

– de QD eenheid is speciaal voor opslag van gegevens ontworpen. In tegenstelling tot de cassetterecorder behoeft de schijveneenheid niet geschikt te zijn voor het opnemen en afspelen van muziek. Hierdoor kan de kwaliteit van de QD eenheid worden toegespitst op de opslag van gegevens hetgeen een sneller en veel betrouwbaardere opslag van gegevens mogelijk maakt.

3.3 Een snelheidsvergelijking

Voor diegene die hierin is geïnteresseerd, volgt hieronder een globale snelheidsvergelijking tussen een cassetterecorder en een QD eenheid.

De cassetterecorder

De hoogste (en helaas ook meest onbetrouwbare) schrijfsnelheid op cassette is 2400 baud, hetgeen overeenkomt met ongeveer 250 tekens per seconde. Om bijvoorbeeld 16 kilobytes aan gegevens (16384 tekens) van band te lezen, is dus een tijd nodig van ongeveer:

$$\frac{16 \times 1024}{250} = 66 \text{ seconden (ruim een minuut).}$$

De QD eenheid

We zagen reeds dat het 6 meter lange spoor van de QD floppy in ongeveer 8 seconden werd afgelopen. In die tijd kunnen maximaal 64 kilobytes aan gegevens worden gelezen. 16 kilobytes kunnen dus worden gelezen in:

$$\frac{8 \times 16}{64} = 2 \text{ seconden.}$$

Beide tijdsberekeningen zijn slechts globaal. Meestal wordt bij gebruik van een cassetterecorder de 1200 baud snelheid gebruikt waardoor de hier berekende tijd moet worden verdubbeld. Aan de andere kant geldt dat een gemiddelde schrijfopdracht op de QD eenheid ongeveer drie maal zo lang duurt als een leesopdracht.

Over het algemeen kan men stellen dat de QD eenheid toch wel ongeveer 25 tot 50 maal zo snel werkt als een cassetterecorder en in ieder geval veel betrouwbaarder in opslag is.

3.4 De logische indeling van een QD floppy

In de vorige paragrafen zagen we, dat we op één kant van een QD floppy een spiraalvormig spoor van ongeveer 6 meter lengte kunnen onderscheiden. Op dit spoor kunnen 64 kilobytes aan gegevens worden opgeslagen.

Deze door de QD eenheid gehanteerde indeling noemen we de fysische indeling van de schijf. Deze indeling kan als het ware op de schijf zelf worden aangewezen.

Met de logische indeling van de schijf bedoelen we de indeling die het MSX QD basic hanteert, gebruik makende van de fysische indeling.

De gebruiker van de QD drive hoeft zelden of nooit stil te staan bij de fysische indeling van de magneetschijf. Het MSX QD basic houdt bij gebruik van een QD floppy automatisch bij welke delen van het spiraalvormige spoor bezet zijn en welk uiteinde nog vrij is. Met een bepaald kommando behoeft slechts een naam van het MSX QD basic te worden verstrekt; het MSX QD basic zoekt verder uit waar het te SAVEn programma in de spiraal kan worden geplaatst.

Het MSX QD basic houdt op elk QD floppy een inhoudsopgave of INDEX bij. In deze inhoudsopgave staat vermeld:

- wat de naam is van de verzameling van gegevens die op schijf werd geschreven.
- wat de soort is van deze verzameling van gegevens. Zijn het bijvoorbeeld programmaregels of zijn het andere gegevens die op de schijf werden geschreven.
- welk gedeelte van het spiraalspoor door deze gegevens wordt bezet. Dit spiraaldeel mag natuurlijk niet voor opslag van andere gegevens worden gebruikt. MSX QD basic waakt hierover.

Een bij elkaar behorende verzameling van gegevens noemt men in de computerwereld over het algemeen een BESTAND of een FILE. De QD eenheid is een bestandgeörienteerd medium. Een verzameling van gegevens wordt altijd onder één naam op de QD floppy geschreven. Hierop gaan we bij de behandeling van de kommando's verder in.

Het MSX QD basic verzorgt de besturing van de bestanden op de QD floppy (het zogenaamde FILE MANAGEMENT) waardoor we als MSX-programmeur niet te veel met de fysieke opbouw van de QD eenheid worden geconfronteerd.

3.5 Files

Zoals in de vorige paragraaf werd opgemerkt, gaat MSX QD basic uit van een bestandsstruktuur op de QD schijf. De term FILE of BESTAND zal steeds weer terugkeren en is erg belangrijk. Onthoudt:

Een bestand of een file is een verzameling van bij elkaar behorende gegevens.

We onderscheiden verschillende soorten files, te weten:

1. de PROGRAM FILE (programmabestand). Een PROGRAM FILE

bestaat uit allemaal bij elkaar behorende programmaregels die tezamen een programma vormen. Een programmabestand wordt met behulp van het CALL SAVE kommando op de QD schijf geschreven en met het CALL LOAD kommando weer in het computergeheugen geladen. Met een CALL KILL-kommando kan een programmabestand tenslotte weer van de QD schijf worden verwijderd.

2. de DATA FILE (gegevensbestand). Een DATA FILE bestaat uit allemaal bij elkaar behorende gegevens. In een data file kunnen we bijvoorbeeld de namen, adressen en telefoonnummers van onze kennissen opslaan. Ook kunnen we een ander gegevensbestand ontwerpen met daarin de gegevens van onze postzegelverzameling, platenverzameling etcetera. Al deze bestanden mogen eventueel op één QD floppy staan; het MSX QD basic zorgt ervoor dat de verschillende gegevens uit elkaar worden gehouden.

De behandeling van PROGRAM FILES en DATA FILES geschiedt in grote lijnen hetzelfde als op cassette, met dit verschil dat het één en ander op de QD eenheid zeer veel sneller gebeurt.

Bij de behandeling van de MSX QD sleutelwoorden gaan we met enkele voorbeelden dieper in op de verschillende soorten files.

4 DE INSTALLATIE VAN UW QD EENHEID

Nu we het een en ander van de QD eenheid weten, gaat het interessant worden om de eerste pogingen op de QD eenheid te ondernemen.

Voordat we dat gaan doen, dienen we eerst de QD eenheid aan onze MSX computer te installeren. Voor diegenen die nog niet eerder een QD eenheid gebruikten, volgt hieronder stap voor stap het installatievoorschrift van de QD eenheid.

4.1 Wat hebben we allemaal nodig

Voordat we daadwerkelijk de QD eenheid gaan installeren, dienen we na te gaan of we alle benodigde spullen wel in huis hebben. Controleert u eerst even of de volgende onderdelen en toebehoren aanwezig zijn:

- uw MSX computer
- een voedingseenheid voor de QD eenheid
- de QD eenheid zelf
- één of meer QD floppies

wat minder belangrijk:

- een wattenstaafje
- een flesje alcohol 96%

Wanneer al deze dingen aanwezig zijn, kunnen we de QD eenheid gaan installeren.

4.2 De voeding

De QD eenheid wordt met vele verschillende soorten voedingen geleverd. De voeding van een QD eenheid dient aan de volgende eisen te voldoen:

- de voeding dient een spanning van 8.6 volt te verstrekken, gelijkstroom, gestabiliseerd. De voeding dient 400 mA te kunnen bieden.
- de positieve pool dient de binnenste pool van het aansluitstekkertje te zijn.

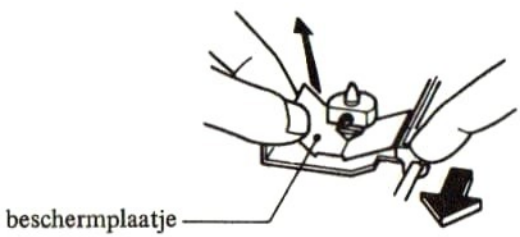
Heeft u een instelbare voeding, stel deze voeding dan op de juiste

spanning (of de meest in de buurt liggende spanning) in. Indien de voeding van polen kan worden gewisseld, controleer dan of de positieve pool op de juiste aansluiting staat.

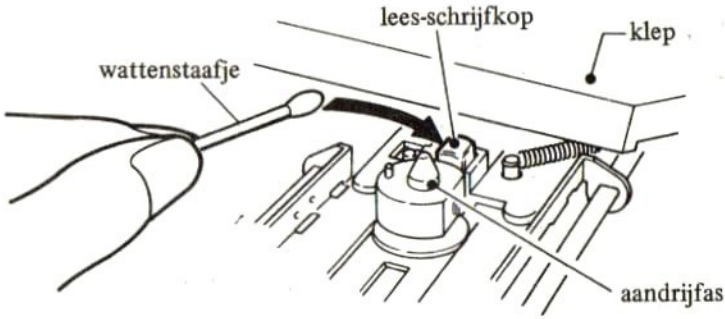
Indien u niet zeker bent van de geschiktheid van de voeding of van de juiste instelling van deze voeding, sluit uw QD eenheid dan niet aan maar raadpleeg eerst uw computerhandelaar.

4.3 De QD eenheid

De QD eenheid heeft achterin een aansluiting voor de voeding. Voordat u een nieuwe QD eenheid voor het eerst aansluit, dient u het beschermingsplaatje te verwijderen dat zich in de QD eenheid bevindt. Zie de bijbehorende afbeelding.



Pas nadat u het beschermingsplaatje heeft verwijderd, kunt u de QD eenheid gaan gebruiken. De ervaring leert dat u de lees- schrijfkop van een nieuwe QD eenheid voor u deze voor de eerste keer gebruikt, het beste eerst een keer kunt schoonmaken. Dit doet u met behulp van het wattenstaafje en de alcohol. Maakt u het wattenstaafje nat met alcohol en wrijft u enige keren licht over de lees- schrijfkop. Laat daarna de QD eenheid enige minuten drogen. Zie voor de lokatie van de lees- schrijfkop de bijbehorende instruktietekening.



Nadat u eventueel de lees- schrijfkop heeft schoongemaakt, kunt u de QD eenheid gaan aansluiten. Doet u dit als volgt:

1. druk de aansluitstekker van de QD eenheid in de sleuf van uw MSX computer. Niet forceren, als de steker niet wil passen, haal hem er dan uit en probeer het nogmaals. Controleert u of de steker niet een halve slag dient te worden gedraaid.

2. sluit de voeding van de QD eenheid aan de eenheid zelf en schakel deze voeding vervolgens in (steek de stekker in het stopkontakt en haal een eventuele schakelaar over.

3. schakel als laatste uw MSX computer in. Na de gebruikelijke melding verschijnt de melding:

Quick Disk System
version 1.00
Copyright 1984 by Mitsumi

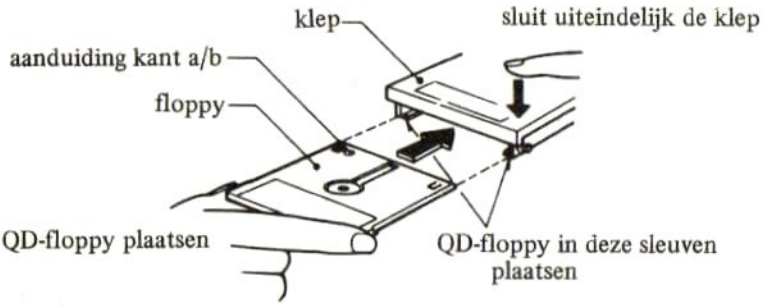
De QD eenheid gaat enige seconden draaien waarna u de gebruikelijke melding (MSX BASIC version...) op het beeldscherm ziet.

Indien u deze melding niet krijgt, is er iets fout gegaan. Schakel alle apparatuur uit en probeer het een en ander nog eens van voor af aan.

Om de eerste keer te controleren of de QD eenheid naar behoren funktioneert, kan men het beste de volgende test uitvoeren:

1. plaats een nog niet eerder gebruikt schijfje in de QD eenheid. Dit doet men door de EJECT-knop in te drukken en vervolgens het schijfje met het leesvenster naar voren gericht in de sleuf van de

plastic klep te schuiven. Uiteindelijk dient de klep weer voorzichtig te worden ingeklapt (zie tekening)



2. Type vervolgens in:

CALL QDFORMAT (return)

Op de vraag "Are you sure?" antwoordt u met de intoetsing van alleen de letter Y.

De floppy wordt vervolgens geformatteerd (zie hoofdstuk 5). Wanneer na enige seconden de melding "Complete" verschijnt, is het formatteren goed gegaan en is het aan te nemen dat de QD eenheid naar behoren werkt.

Wanneer een foutmelding volgt, kunt u het beste nog enkele pogingen, eventueel met andere schijfjes ondernemen. Ook kunt u proberen om de lees- schrijfkop schoon te maken. Wanneer u uiteindelijk toch geen succes heeft, is er iets mis:

- de QD eenheid weigert elke dienst. Hij gaat niet draaien en de foutmelding volgt snel. Waarschijnlijk is de voeding te laag ingesteld, verkeerd-om aangesloten of misschien bent u gewoon vergeten om de stekker van de voeding in het stopkontakt te steken.
- de QD eenheid draait wel maar is vrij lang bezig voordat een foutmelding gegeven wordt. Waarschijnlijk dient de lees- schrijfkop te worden schoongemaakt. Ook kan het zijn dat de floppy niet in orde is. In een enkel geval geeft de voeding te weinig vermogen af.

4.4 Enige gouden regels

Om lang plezier te hebben van uw QD eenheid, dient u de volgende, gouden regels nauwkeurig in acht te nemen:

- gebruik nooit andere ROM cassettes tegelijk met uw QD eenheid. Vaak zal de combinatie niet goed werken.

- sluit de QD eenheid nooit aan of af zonder eerst de MSX computer en de QD eenheid te hebben uitgeschakeld. Het aan- of afsluiten van de QD eenheid aan of van de computer terwijl de spanning niet is uitgeschakeld, kan resulteren in een blijvende beschadiging aan computer of QD eenheid.

- berg de QD floppies altijd rechtstandig op. Plaats ze hiertoe in de bijbehorende kartonnen doos. Houd de floppies altijd ver van magnetische voorwerpen (luidsprekers, beeldbuizen, elektromotoren, enz.). De gegevens zijn in magnetische vorm op de floppy opgeslagen; blootstelling aan magnetische velden kan de gegevens op de floppy vernietigen.

- houd van een belangrijk programma altijd minstens drie verschillende kopieën op verschillende floppies bij. Gaat er iets mis, dan heeft u op een andere floppy toch uw programma nog.

- laat een floppy nooit onbeschermd op tafel liggen. Schuif een floppy altijd in de bijbehorende enveloppe.

- stel de floppies nooit bloot aan direkt zonlicht of grote warmte.

- bescherm zowel de floppies als de QD drive tegen sterke temperatuurswisselingen. Wanneer floppies plotseling van een koude naar een warme omgeving worden overgebracht, laat ze dan minstens een uur IN DE DOOS acclimatiseren alvorens ze te gebruiken.

- schrijf nooit met potlood of balpen op het etiket wanneer dat reeds op de floppy is bevestigd. Gebruik een zachte viltstift.

- verwissel geen floppies terwijl de QD eenheid aan het werk is. Niet alleen de verwijderde maar ook de eventueel nieuw geplaatste floppy kan worden "opgeblazen" (de gegevens kunnen worden verminkt).

4.5 De viltkop

Het kan zijn dat na enige tijd, bij veel gebruik, de QD eenheid steeds meer storing gaat geven. Dit kan een indikatie zijn dat de floppies versleten zijn en dus dienen te worden vervangen. Ook kan het zijn dat de lees- schrijfkop weer eens dient te worden schoongemaakt.

Het kan echter ook zijn dat de viltkop waarmee de schijf op de lees-schrijfkop wordt gedrukt, versleten is. In dat geval dient deze te worden vervangen door een nieuwe. Eén viltkop wordt altijd standaard bij de QD eenheid geleverd. Losse viltkoppen zijn bij uw computerhandelaar te bestellen.

Het is helemaal geen slecht idee om de viltkop door uw computerhandelaar te laten vervangen wanneer u niet zeker bent van uw zaak. Het vervangen van de viltkop is niet erg ingewikkeld maar moet met enige zorg gebeuren.

Het vervangen van de viltkop gaat als volgt:

- draai met een kruiskopschroevendraaier de twee schroefjes los die de klep aan de achterkant bevestigen. Pas op! Het zijn kleine schroefjes, verlies ze niet.

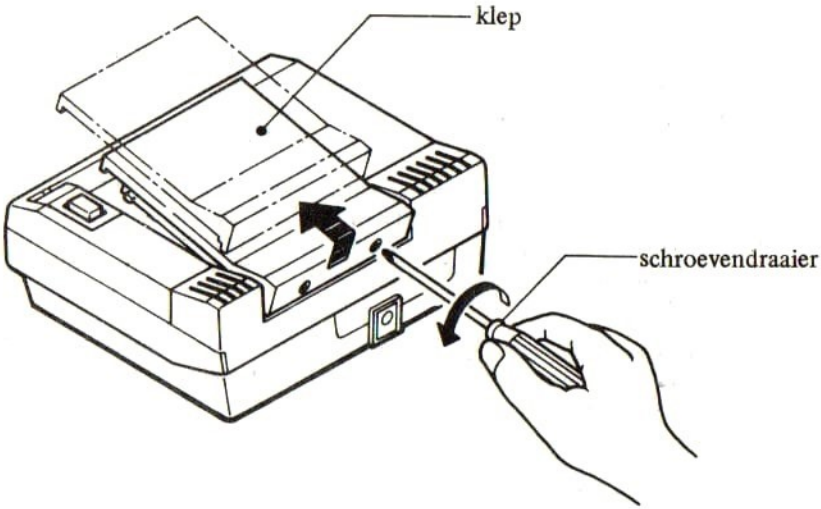
- trek de klep aan de achterkant iets omhoog en schuif hem dan naar voren. De klep is nu los. Leg de klep op een veilige plaats opzij.

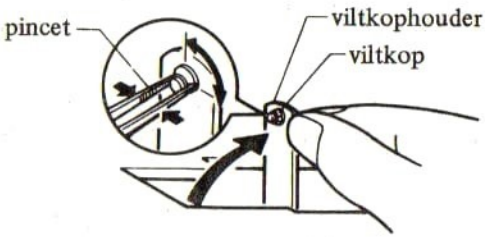
- bovenop de QD eenheid kunt u nu de aandrukarm onderscheiden. Deze laat zich gemakkelijk optillen en veert bij loslaten weer naar beneden. Wanneer u het aandrukarmpje oplicht, kunt u aan de binnenkant de viltkop onderscheiden.

- pak de viltkop (inklusief het omringende kunststof randje) met een pincet vast en draai de viltkop een kwart slag. Hij komt nu los.

- plaats met de pincet de nieuwe viltkop op de plaats van de oude. Draai de viltkop met een kwartslag weer vast.

- bevestig uiteindelijk de klep weer met de twee schroefjes op de QD drive.


klep
schroevendraaier


pincet
viltkophouder
viltkop

5 DE MSX QD BASIC SLEUTELWOORDEN EN HUN BETEKENIS

In dit hoofdstuk worden de MSX QD sleutelwoorden behandeld. Voorafgaand aan deze behandeling worden eerst enkele sleutelwoorden behandeld die in het grote MSX handboek reeds werden behandeld maar in het MSX QD basic een extra betekenis krijgen. Daarna worden de nieuwe sleutelwoorden behandeld.

De schrijfwijzen worden in de BNF notatievorm gepresenteerd. Deze notatievorm werd uitgebreid in het grote MSX handboek behandeld.

Omdat het hier slechts om een beperkt aantal sleutelwoorden gaat, worden deze, in tegenstelling tot in het grote MSX handboek, hier in de aanbevolen leervolgorde gepresenteerd. In de inhoudsopgave vindt u deze sleutelwoorden op alfabetische volgorde opgenomen met de bijbehorende paginanummering.

Per nieuw sleutelwoord worden de volgende gegevens verstrekt:

- de naam van het sleutelwoord.
- de moeilijkheidsgraad van dit sleutelwoord. Deze moeilijkheidsgraad varieert van zeer eenvoudig tot zeer moeilijk.
- de soort. We onderscheiden binnen de sleutelwoorden de A-FUNKTIES (funkties met een alfanumeriek resultaat), de N-FUNKTIES (funkties met een numeriek resultaat), SYSTEEM-VARIABELEN (variabelen met een voorbestemde waarde) en KOMMANDO'S (sleutelwoorden die een aktie aangeven).
- de afkomst. Alle in MSX-basic voorkomende sleutelwoorden zijn afkomstig uit de engelse taal. Het is vaak gemakkelijker om een sleutelwoord en de bijbehorende betekenis te onthouden wanneer men de preciese afkomst kent. Daarom is van elk sleutelwoord de afkomst naar het nederlands herleid.
- de schrijfwijze in BNF-notatievorm.
- de betekenis. De funktie van elk sleutelwoord wordt uitvoerig behandeld. Van de sleutelwoorden die reeds in het grote MSX

handboek werden behandeld, is slechts een aanvullende beschrijving opgenomen.

– een voorbeeld. Waar zinvol is een voorbeeld opgenomen.

Wanneer u de gegeven voorbeelden wilt uitproberen, is het zaak om enige geformatteerde schijfjes bij de hand te houden. In hoofdstuk 4 zagen we reeds hoe een schijfje dient te worden geformatteerd.

Totdat de CALL LOAD en CALL SAVE worden behandeld, is het verstandig om voorbeelden die u wilt bewaren, op de gebruikelijke wijze op cassetteband te zetten. Eventueel kunt u deze voorbeelden dan later (bijvoorbeeld met CALL CASQD) op een schijfje zetten.

De speciale QUICK DISK kommando's zijn allemaal CALL's. Het sleutelwoord CALL mag in alle gevallen worden vervangen door het teken _ (onderstreping). Voor de duidelijkheid zijn alle voorbeelden met het onverkorte sleutelwoord CALL uitgevoerd. Dit sleutelwoord mag ook in de voorbeelden in alle gevallen door het onderstrepingsteken worden vervangen.

5.1 Enkele oude sleutelwoorden wat nader behandeld

OPEN

Het OPEN-kommando wordt gebruikt om een bestand, een groep van bij elkaar behorende gegevens, in te richten cq te kunnen gebruiken.

Het beeldscherm (CRT:), het grafische beeldscherm (GRP:), de printer (LPT:) of de cassetterecorder (CAS:) leerden we in het grote MSX handboek reeds kennen. Als nieuwe randapparaten introduceert MSX QD basic de randapparaten:

QD: de eerste of enige aangekoppelde QUICK DISK DRIVE
QD0: de eerste of enige aangekoppelde QUICK DISK DRIVE
QD1: de tweede aangekoppelde QUICK DISK DRIVE
QD2: de derde aangekoppelde QUICK DISK DRIVE
.
.
.
QD7: de achtste aangekoppelde QUICK DISK DRIVE

MAXFILES

De mogelijkheden van de QUICK DISK worden extra groot doordat we maximaal twee bestanden TEGELIJK op de QD eenheid kunnen openen. Op de cassetterecorder was dat eerder niet mogelijk, maar omdat de QD eenheid zelf bijhoudt waar de verschillende bestanden worden opgeslagen op de schijf, kan dat met de QD eenheid nu wél.

Wanneer meerdere bestanden dienen te worden geopend, dan dient de systeemvariabele MAXFILES gelijkgesteld te worden aan het aantal tegelijk te openen bestanden. Een voorbeeld:

```
NEW
Ok
10 MAXFILES=2
20 OPEN "QD:FILE1" FOR INPUT AS 1
30 OPEN "QD:FILE2" FOR OUTPUT AS 2
40 ...
```

Op regel 10 werd bepaald dat er maximaal twee bestanden tegelijk mogen worden geopend. Op regel 20 werd vervolgens bestand FILE1 op de QD eenheid geopend. Indien dit bestand niet op de schijf wordt gevonden, zal een foutmelding volgen. Op regel 30 wordt het bestand FILE2 op de QD eenheid aangemaakt. Met FOR OUTPUT geeft men aan dat men een nieuw bestand met gegevens wil vullen. Indien FILE2 reeds blijkt te bestaan op de QD-schijf, volgt een foutmelding.
Op regel 40 vervolgt het programma tenslotte.

Maximaal twee bestanden kunnen op de QD eenheid tegelijk worden geopend. Deze mogen dan niet allebei FOR OUTPUT worden geopend.

CLOSE

Voordat we dit sleutelwoord verder behandelen, eerst een waarschuwing:

Een bestand dat nog niet is gesloten, is meestal niet helemaal bijgewerkt. Indien een disk wordt vervangen door een andere terwijl bestanden

nog niet gesloten zijn, kan het gebeuren dat de eerste schijf niet volledig wordt bijgewerkt, terwijl de tweede wordt 'opgeblazen' (de gegevens worden vaak volledig verminkt). Nadat een programma is onderbroken, kunnen bestanden toch nog openstaan; pas bij een poging om het programma te veranderen, kan het plotseling gebeuren dat de disk unit weer even gaat werken; de openstaande bestanden worden dan alsnog gesloten.

Alvorens schijven te verwisselen, moet u zich aanwennen om eerst een close-kommando voor de zekerheid in te toetsen zonder regelnummer. Verwijder pas na de Ok-melding de schijf; er kan dan niets meer fout gaan.

Dus tik eerst in:

```
CLOSE
Ok
```

en verwijder dan pas de schijf.

Het CLOSE-kommando sluit een specifiek bestand; maakt het verder onbenaderbaar voor het programma. Alle kanalen kunnen in één keer worden gesloten door uitvoering van een enkele CLOSE. Door achter het sleutelwoord CLOSE een kanaalnummer te vermelden (eventueel voorafgegaan door een hekje) kan een specifiek bestand worden gesloten, namelijk het bestand dat eerder op het genoemde kanaalnummer werd geopend.

PRINT#

Het PRINT#-kommando heeft PRECIES dezelfde werking in het MSX-disk basic als in het gewone MSX-basic. Het OPEN-kommando bepaalt op welk randapparaat het PRINT#-kommando wordt uitgevoerd; het printkommando zelf verandert niet.

(LINE)INPUT#

Ook het (LINE) INPUT#-kommando behoudt in het MSX-disk basic

precies dezelfde werking. Door het open-kommando wordt bepaald vanuit welk randapparaat gegevens worden ingelezen.

EOF(...)

De N-FUNKTIE EOF behoudt eveneens dezelfde funktie; het OPEN-kommando is bepalend voor het type randapparaat.

Tot slot volgen hier enkele kleine voorbeelden waarin op schijf wordt gewerkt en waarin de oude, hierboven behandelde sleutelwoorden worden gebruikt.

Voorbeeld 1

In dit voorbeeld wordt bestand BES01 op de schijf toegewezen. Vervolgens kunnen regels tekst worden ingegeven die dan in dit bestand worden weggeschreven. Als zodanig kan bijvoorbeeld een brief op schijf worden bewaard.

```
NEW
Ok
10 REM VOORBEELD 1, INGAVE TEKST
20 CLS:OPEN "QD:BES01" FOR OUTPUT AS 1
30 PRINT "GEEF TEKST IN (*=EINDE)"
40 LINE INPUT R$:IF R$="*" THEN 60
50 PRINT#1,R$:GOTO 40
60 CLOSE:STOP
Ok
RUN

(beeldscherm wordt schoongemaakt)

GEEF TEKST IN (*=EINDE)

Beste lezer,

Met deze brief testen we ons eerste
```

```
voorbeeld even uit. Hopelijk komt
deze tekst netjes in bestand BES01
te staan.

Hoogachtend,

Uw auteur.

*
Break in 60
Ok
```

Voorbeeld 2

In dit voorbeeld wordt bestand BES01 geopend en bestand BES02 toegewezen. Vervolgens worden de regels tekst die in BES01 liggen opgeslagen, stuk voor stuk 'overgeheveld' in bestand BES02. Er kan worden bepaald welke regels wel en niet worden overgeheveld. Ook kunnen regels worden tussengevoegd.

```
NEW
Ok
10 REM VOORBEELD 2, OVERHEVELEN
20 MAXFILES=2
30 OPEN "0D:BES01" FOR INPUT AS 1
40 OPEN "0D:BES02" FOR OUTPUT AS 2
50 CLS:PRINT "0=REGEL OVERSLAAN"
60 PRINT "1=REGEL OVERNEMEN"
70 PRINT "2=NA DEZE REGEL TUSSENVOEGEN"
80 PRINT
90 IF EOF(1) THEN 190
100 LINE INPUT#1,R$
110 PRINT R$
120 K$=INKEY$:IF K$="" THEN 120
130 IF K$<>"0" AND K$<>"1" AND K$<>"2" T
HEN BEEP:GOTO 120
140 IF K$="0" THEN PRINT "VERWIJDERD":BE
```

```
EP:GOTO 90
150 PRINT#2,R$:IF K$="1" THEN 90
160 REM TUSSENVOEGEN
170 PRINT "TUSSENVOEGEN (*=EINDE)"
180 LINE INPUT R$:IF R$="*" THEN PRINT "
EINDE TUSSENVOEGEN":GOTO 90 ELSE PRINT #
2,R$:GOTO 180
190 CLOSE:STOP
RUN

(beeldscherm wordt schoongemaakt)

0=REGEL OVERSLAAN
1=REGEL OVERNEMEN
2=NA DEZE REGEL TUSSENVOEGEN


Beste lezer,

TUSSENVOEGEN (*=EINDE)
Allereerst mijn hartelijke dank voor
de aanschaf van dit boekwerkje.

*
EINDE TUSSENVOEGEN
Met deze brief testen we ons eerste
voorbeeld even uit. Hopelijk komt
deze tekst netjes in bestand BES01
TUSSENVOEGEN (*=EINDE)
terecht.
*
EINDE TUSSENVOEGEN
te staan.
VERWIJDERD

Hoogachtend,
```

```
Uw auteur.

Break in 190
Ok
```

Voorbeeld 3

In dit voorbeeld worden de tekstregels die in bestand BES02 liggen opgeslagen, allemaal in het computergeheugen ingelezen. Daarna worden de bestanden BES02 en BES01 van de schijf verwijderd (zie de behandeling van het CALL KILL-kommando verderop) en wordt een nieuw bestand BES01 benoemd. De tekstregels worden vervolgens in BES01 teruggeschreven. Zo kan het programma in voorbeeld 2 steeds worden herhaald totdat het stuk tekst naar wens is uitgevoerd.

```
NEW
Ok
10 REM VOORBEELD 3, BES02->BES01
15 CLEAR 8096
20 OPEN "QD:BES02" FOR INPUT AS 1
30 DIM R$(300)
40 FOR I=0 TO 300
50 IF EOF(1) THEN 70
60 LINE INPUT #1,R$(I):NEXT I
70 CLOSE:CALL QDKILL("BES02"):CALL QDKIL
L("BES01")
80 OPEN "QD:BES01" FOR OUTPUT AS 1
90 FOR J=0 TO I-1:PRINT#1,R$(J):NEXT J
100 CLOSE:STOP
RUN
Break in 100
Ok
```

Voorbeeld 4

In dit voorbeeld wordt bestand BES01 regel voor regel ingelezen en op beeldscherm afgedrukt. Door op regel 40 de PRINT te vervangen door een LPRINT, kan worden bewerkstelligd dat de regels op een eventuele printer worden afgedrukt.

```
NEW
Ok
10 REM VOORBEELD 4, PRINT BES01
20 CLS:OPEN "QD:BES01" FOR INPUT AS 1
30 IF EOF(1) THEN CLOSE:STOP
40 LINE INPUT #1,R$:PRINT R$:GOTO 30
RUN

(beeldscherm wordt schoongemaakt)


Beste lezer,

Allereerst mijn hartelijke dank voor
de aanschaf van dit boekwerkje.

Met deze brief testen we ons eerste
voorbeeld even uit. Hopelijk komt
deze tekst netjes in bestand BES01
terecht.

Hoogachtend,

Uw auteur.

Break in 30
Ok
```

Deze vier programma's bij elkaar vormen een héél eenvoudig tekstverwerkingspakketje. Voor de echte amateur misschien een uitdaging om het één en ander te verfraaien...

5.2 De nieuwe sleutelwoorden

In deze paragraaf worden de sleutelwoorden behandeld die het MSX-QD-basic extra biedt.

Echter, eerst dienen twee veel voorkomende begrippen te worden behandeld en wel:

⟨SCHIJFEENHEID⟩

en

⟨BESTANDSNAAM⟩

Wanneer in dit verdere hoofdstuk de term SCHIJFEENHEID wordt gebruikt, dan wordt daarmee de aanduiding van de schijfeenheid bedoeld waarop een eventuele aktie dient plaats te vinden.

Het MSX-QD-basic kent in verband met de QUICK DISK de volgende aanduidingen voor de schijveneenheid:

"QD:"	de eerste QD-eenheid
"QD0:"	eveneens de eerste schijveneenheid
"QD1:"	de tweede QD-eenheid
"QD2:"	de derde eenheid
.	
.	
.	
"QD7:"	de achtste eenheid

Alhoewel de meeste gebruikers slechts één QD-eenheid aan hun MSX computer zullen schakelen, is het in theorie mogelijk om maximaal 8 QD-eenheden tegelijkertijd aan een MSX-computer te schakelen.

Wanneer in het verdere hoofdstuk wordt gesproken over een BESTANDSNAAM, dan wordt daarmee een alfanumerieke aanduiding bedoeld waarmee een bestand wordt getypeerd.

Een bestandsnaam bestaat in principe uit drie verschillende delen. Het eerste deel wordt gevormd door een aanduiding van het randapparaat waarop het betreffende bestand zich bevindt of moet gaan bevinden. We kennen in het MSX QD basic de volgende randapparaat-aanduidingen:

CAS: cassetterecorder	CRT: beeldscherm
GRP: grafisch scherm	LPT: printer
QD: quick disk	QD0...7: quick disk (zie hierboven)

Dit eerste deel van de bestandsnaam kan eventueel worden weggelaten; het meest voor de hand liggende randapparaat wordt dan geselekteerd. In de specifieke QD-kommando's (de kommando's die met een CALL worden aangeroepen en hierna worden beschreven) zal het weglaten van de randapparaat-aanduiding tot gevolg hebben dat automatisch de eerste aangeschakelde quick disk eenheid wordt geselekteerd. In standaard MSX-kommando's (zoals bijvoorbeeld het OPEN-kommando) zal echter standaard de cassetterecorder worden geselekteerd.

In specifieke Quick Disk kommando's is het op één uitzondering na (zie CASQD) verboden om een andere randapparaat-aanduiding dan QD: of QD0...7: te gebruiken.

Het tweede deel van de bestandsnaam wordt gevormd door de werkelijke naam van het bestand. Meestal dient deze naam te beginnen met een letter (behalve wanneer ook een randapparaat-aanduiding is opgenomen). De naam kan verder uit elk teken (behalve het aanhalingsteken) worden opgebouwd en maximaal 8 tekens lang zijn. Wanneer we een naam langer dan 8 tekens opgeven, dan worden het negende, tiende en elfde teken gebruikt voor de typering (zie hieronder) terwijl de overige tekens worden verwaarloosd.

Het derde en laatste deel van een bestandsnaam bestaat uit de zogenaamde bestandstypering. Deze typering dient maximaal drie letters lang te zijn en moet worden voorafgegaan door een punt. In deze typering wordt, alhoewel het feitelijke gebruik vrij is, meestal een afkorting van het type van bestand opgenomen. Vaak worden standaard de volgende typeringen gehanteerd:

XXXXXXXX.BAS	een programma, gesaved zonder de ,A-optie
XXXXXXXX.ASC	een programma, gesaved met de ,A-optie
XXXXXXXX.TXT	een tekstbestand
XXXXXXXX.SEQ	een sequentieel bestand
XXXXXXXX.$$$	een tijdelijk bestand

Met XXXXXXXX wordt in deze voorbeelden steeds het eerste deel van de bestandsnaam aangeduid.

Door deze typeringen consequent te gebruiken, kunnen we later op een schijf met vele bestanden snel het juiste bestand terugvinden.

Enkele voorbeelden van bestandsnamen:

"QD:VB1.BAS"	een basic programma genaamd VB1 op de QD-eenheid
"CAS:OVL.ASC"	een basic programma, genaamd OVL, met de ‚A-optie op cassette gezet
"BRIEF.TXT"	een tekstbestand genaamd BRIEF op het meest voor de hand liggende randapparaat

Zoals we later bij de behandeling van het betreffende kommando zullen zien, kunnen we bijvoorbeeld een programma met het volgende kommando op een QD-schijfje zetten:

```
CALL SAVE("QD:PRO1.BAS")
```
of
```
CALL SAVE("PRO1.BAS")
```

Later kan het dan weer met:

```
CALL LOAD("PRO1.BAS")
```
of
```
CALL LOAD("QD:PRO1.BAS")
```

in het computergeheugen worden teruggehaald.

SLEUTELWOORD # CALL QDFORMAT

moeilijkheidsgraad . . . . . **normaal**
soort . . . . . **KOMMANDO**
afkomst . . . . . **QDFORMAT is een afkorting van quick disk format: het formatteren van een quick disk schijfje**

schrijfwijze

```
CALL(")QDFORMAT(")[(<SCHIJFEENHEID>[)]]
<SCHIJFEENHEID>::=<A>
<A>::=<ZIE ALGEMENE SPECIFICATIES>
```

betekenis

Een schijf die nog nooit eerder werd gebruikt, moet voor gebruik eerst door de computer worden ingedeeld of geFORMATteerd. Tijdens het formatteren wordt het schijfje geschikt gemaakt voor gebruik in de QD eenheid.

Ook wanneer we in één keer een schijfje helemaal leeg willen maken, kunnen we dit doen door het schijfje te formatteren. Het zal in dit verband duidelijk zijn dat dit kommando met uiterste voorzichtigheid dient te worden gehanteerd: één keer een verkeerd schijfje formatteren en een enorme hoop waardevolle gegevens kan in één klap verloren gaan. Wanneer we het bevel:

```
CALL QDFORMAT
```

ingeven, vraagt de computer ons dan ook onmiddellijk eerst:

```
ARE YOU SURE (Y/N)?
```

Pas wanneer we de letter Y (van YES) intoetsen, neemt het formatteren, dat ongeveer 10 seconden in beslag neemt, een aanvang. Op dat moment gaan alle eventuele gegevens op die schijf compleet verloren; de schijf wordt voor hernieuwd gebruik klaar gemaakt.

Enkele voorbeelden van het QDFORMAT-kommando:

```
CALL QDFORMAT
```

het schijfje in de eerste QD eenheid wordt geformatteerd

```
CALL QDFORMAT("QD:")
```
idem

```
CALL QDFORMAT("QD7:")
```
het schijfje in de achtste QD eenheid wordt geformatteerd

```
CALL QDFORMAT(A$)
```
het schijfje dat zich in de door A$ aangeduide QD eenheid bevindt, wordt geformatteerd. Is A$ bijvoorbeeld gelijk aan "QD2:" dan wordt het schijfje in de derde QD eenheid geformatteerd.

SLEUTELWOORD # CALL QDFILES

moeilijkheidsgraad ..... **eenvoudig**
soort ..... **KOMMANDO**
afkomst ..... **QDFILES IS een afkorting van quick disk files – bestanden op het quick disk schijfje**

schrijfwijze

```
CALL("}QDFILES("}[(<SCHIJFEENHEID>[)]]
<SCHIJFEENHEID>::=<A>
<A>::=<ZIE ALGEMENE SPECIFICATIES>
```

betekenis

Met het sleutelwoord QDFILES kan worden bekeken, welke bestanden zich op een schijf bevinden. Voorbeelden:

```
CALL QDFILES
```
laat zien welke bestanden zich op het schijfje in de eerste (enige) QD eenheid bevinden

```
CALL QDFILES("QD4:")
```
laat zien welke bestanden zich op de vijfde aangesloten QD eenheid bevinden

Het volgende voorbeeld laat de inhoud van zo maar een schijf zien:

```
_QDFILES
      Name              Atr  Size
("QD0:                  "'03 0400
("QD0:QD01              "'03 0400
("QD0:QD02A             "'03 0400
("QD0:QD02B             "'03 0400
("QD0:QD03A             "'03 0400
("QD0:QD03B             "'03 0400
("QD0:QD03C             "'03 0400
("QD0:QD03D             "'03 0400
("QD0:QD04              "'03 0400
("QD0:QD05A             "'03 0400
```

```
("QD0:QD05B              "'03 0400
Ok
```

Met behulp van de pijl omhoog en funktietoets 1 kan een bepaald programma worden geselekteerd en uitgevoerd. Zet hiertoe de cursor eerst op de plaats van het betreffende programma in de zojuist vervaardigde lijst en druk vervolgens funktietoets 1 en de return-toets in. Het betreffende programma wordt van schijf geladen en uitgevoerd. Dergelijke funkties kunnen ook met funktietoets 2 (LOAD) en funktietoets 3 (BLOAD) worden bewerkstelligd. Een voorbeeld:

```
_QDFILES
       Name                Atr Size
("QD0:                    "'03 0400
("QD0:QD01                "'03 0400
("QD0:QD02A               "'03 0400
("QD0:QD02B               "'03 0400
("QD0:QD03A               "'03 0400
_LOAD ("QD0:QD03B                 "'03 0400
("QD0:QD03C               "'03 0400
("QD0:QD03D               "'03 0400
("QD0:QD04                "'03 0400
("QD0:QD05A               "'03 0400
("QD0:QD05B               "'03 0400
Ok
```

Uit de zojuist vervaardigde lijst wordt met behulp van de pijl-naar-boven-toets een programma geselekteerd. Dit programma wordt met funktietoets 2 tenslotte geladen.

Achter de bestandsnamen worden nog twee gegevens per bestand verstrekt door QDFILES. Het eerste gegeven staat onder "Atr" en vormt het zogenaamde attribuut van het betreffende bestand. Dit attribuut kan gelijk zijn aan:

- 01 machinetaalprogramma, binair bestand
- 02 BASIC programma
- 03 sequentiëel bestand
- 0B beeldschermbestand (beeldscherminhoud, met BSAVE op schijf gezet)

Het tweede extra gegeven wordt gevormd door de grootte van het bestand. Deze grootte, onder 'Size' vermeld, wordt hexadecimaal (zestientallig) voorgesteld. Met het volgende programma kan deze hexadecimale grootte naar een decimale grootte worden omgerekend:

```
NEW
Ok
10 INPUT "BESTANDSGROOTTE:";A$
20 IF LEN(A$)<>4 THEN 10 ELSE FOR I=1 TO
 4:Q=ASC(MID$(A$,I)):IF Q<48 OR Q>57 AND
 Q<65 OR Q>70 THEN 10 ELSE NEXT I:R=0
30 FOR I=1 TO 4:Q=ASC(MID$(A$,I)):R=R*16
+Q-48+(Q>64)*7:NEXT I
40 PRINT "BESTANDSGROOTTE:";R;" TEKENS."
50 GOTO 10
RUN
BESTANDSGROOTTE:? 0400
BESTANDSGROOTTE: 1024  TEKENS.
BESTANDSGROOTTE:? 001C
BESTANDSGROOTTE: 28  TEKENS.
BESTANDSGROOTTE:? 14FE
BESTANDSGROOTTE: 5374  TEKENS.
BESTANDSGROOTTE:?
```

SLEUTELWOORD # CALL QDKILL

moeilijkheidsgraad . . . . . **normaal**
soort . . . . . **KOMMANDO**
afkomst . . . . . **QDKILL is een afkorting van quick disk kill – verwijder bestand van quick disk**

schrijfwijze

```
CALL("}QDKILL("}(<BESTANDSNAAM>[)]
<BESTANDSNAAM>::=<A>
<A>::=<ZIE ALGEMENE SPECIFICATIES>
```

betekenis

Met het sleutelwoord QDKILL kunnen we een bestand van de QD schijf verwijderen. Voorwaarde hiervoor is dat het te verwijderen bestand bij CALL QDFILES als laatste wordt geprojekteerd (en dus ook als laatste op de schijf is gezet). Indien het bestand niet als laatste op de schijf staat, is verwijderen niet mogelijk zonder eerst de bestanden te verwijderen die ná het betreffende bestand op de schijf zijn geplaatst.

Een voorbeeld:

```
_QDFILES
      Name            Atr Size
("QD0:BES01           "'03 0400
("QD0:TEST            "'02 0092
("QD0:PROG     .BAS"'02 0191
Ok
CALL QDKILL("PROG.BAS")
Ok
CALL QDKILL("BES01")
Device I/O error
Ok
```

In het eerste geval kon het bestand worden verwijderd; in het tweede geval niet omdat het betreffende bestand niet als laatste op de schijf stond.

moeilijkheidsgraad . . . . . **vrij eenvoudig**
soort . . . . . **KOMMANDO**
afkomst . . . . . **CASQD is een afkorting van cassette to quick disk – van cassette naar quick disk**

schrijfwijze

```
CALL{"}CASQD{"}[([<BESTANDSNAAM1>][,
  <BESTANDSNAAM2>][)]\<...>()
<BESTANDSNAAM1>::=<A>
<BESTANDSNAAM2>::=<A>
<A>::=<ZIE ALGEMENE SPECIFICATIES>
```

betekenis

Met het sleutelwoord CASQD kan een bestand direkt van cassette naar een QD-schijfje worden gekopiëerd. Dit is vooral handig wanneer bijvoorbeeld machinetaal-spelletjes op schijf moeten worden gezet. In zeer veel gevallen is dit zonder het bevel CASQD niet mogelijk.

Wanneer alleen het bevel CALL CASQD wordt gebruikt, wordt het eerst gevonden bestand op cassette onder dezelfde naam op het schijfje in de eerste QD eenheid gezet. Wanneer bestandsnamen worden gespecificeerd, dient men zich te realiseren dat als randapparaat-aanduiding in de eerste bestandsnaam geen andere aanduiding dan CAS: is toegestaan. In de tweede bestandsnaam is alleen de aanduiding van een QD-eenheid toegestaan (QD0...7). In de eerste bestandsnaam MAG, in de tweede bestandsnaam MOET een randapparaat-aanduiding worden opgenomen.

Enkele voorbeelden:

```
CALL CASQD
```
het eerste op cassette aangetroffen bestand wordt onder de aangetroffen naam op schijf gezet

```
CALL CASQD("PROG.BAS")
```
de cassetteband wordt afgezocht totdat het programma PROG.BAS wordt gevonden. Dit programma wordt onder dezelfde naam op schijf gezet

```
CALL CASQD(,"QD:PR")
```
het eerste op de band aangetroffen bestand wordt onder de naam PR op de eerste QD eenheid vastgelegd

```
CALL CASQD("TEST","QD1:TEST1")
```
de cassetteband wordt afgezocht totdat het bestand TEST wordt gevonden. Dit bestand wordt onder de naam TEST1 op de tweede aangesloten QD-eenheid veilig gesteld.

NB: zet ter voorkoming van leesfouten op de band het volume van de cassetterecorder altijd op ongeveer 80%. Indien mogelijk dient u de naar schijf gekopiëerde gegevens op korrektheid te toetsen.

SLEUTELWOORD # CALL SAVE

moeilijkheidsgraad . . . . . **normaal**
soort . . . . . **KOMMANDO**
afkomst . . . . . **SAVE – veilig stellen, bewaren**

schrijfwijze

```
CALL{"}SAVE{"}(<BESTANDSNAAM>[,A][)]
<BESTANDSNAAM>::=<A>
<A>::=<ZIE ALGEMENE SPECIFICATIES>
```

betekenis

Met het sleutelwoord SAVE kan een programma op schijf worden geplaatst. Dit kan op twee manieren:

1. zonder de ,A-optie. Het programma wordt op schijf gezet zoals het zich ook in het computergeheugen bevindt. Hierdoor wordt ruimte op de schijf gewonnen maar kan het programma later niet met CALL MERGE bij een ander programma worden gevoegd. Ook kan het programma later niet als DATA-bestand worden gebruikt door een ander programma.

2. met de ,A-optie. Het programma wordt in ASCII formaat (volledig in leesbare lettertekens) op schijf geplaatst. Het nadeel is, dat een op dergelijke wijze vastgelegd programma wat meer ruimte inneemt. Voordeel is, dat het programma later bij een CALL MERGE kan worden gebruikt en eventueel als DATA-bestand (tekstbestand) in een ander programma kan worden geopend.

Indien de ,A-optie wordt gebruikt, behoeft geen programmanaam te worden gespecificeerd. In dat geval dient echter wél een randapparatuur-specifikatie te worden opgenomen. Het kommando

```
CALL SAVE("QD:",A)
```

resulteert hierin, dat een programma in ASCII-formaat ZONDER NAAM op schijf wordt vastgelegd. Dit programma kan niet met CALL LOAD of CALL RUN worden benaderd. Echter, met CALL MERGE (zie verder) kan dit programma wel worden geladen.

Wanneer een programma, in BASIC of in machinetaal geschreven, op schijf gezet wordt onder de naam

```
AUTOEXEC
```

dan wordt indien de computer wordt aangezet met het schijfje met dit programma erop in de QD eenheid, dit programma automatisch opgestart.

Met het CALL LOAD-kommando kan een met CALL SAVE vastgelegd programma weer naar het computergeheugen worden teruggehaald.

Een voorbeeld:

```
CALL SAVE("TEST")
Ok
CALL SAVE("TEST")
Device I/O error
Ok
CALL SAVE("TEST.ASC",A)
Ok
CALL LOAD("TEST.ASC")
Ok
```

In het eerste geval werd een programma op schijf gezet in de vorm zoals het zich ook in het computergeheugen bevindt. In het tweede geval werd nogmaals hetzelfde gedaan. Dit keer mislukte dit echter omdat er reeds een programma met de gebruikte naam op de schijf stond. In het derde geval werd het programma onder een andere naam met de ,A-optie gesaved; het programma staat hierdoor in leesbare lettertekens als DATA-file op schijf. In het laatste geval werd dit programma weer van schijf in het geheugen geladen.

NB: wanneer een CALL SAVE-kommando in een programmaregel is opgenomen, zal bij uitvoering van deze regel na het veilig stellen een Ok-melding verschijnen; het programma gaat niet verder.

SLEUTELWOORD # CALL LOAD

moeilijkheidsgraad . . . . . **normaal**
soort . . . . . **KOMMANDO**
afkomst . . . . . **LOAD is laden**

schrijfwijze

```
CALL{"}LOAD{"}(<BESTANDSNAAM>[,R][)]
<BESTANDSNAAM>::=<A>
<A>::=<ZIE ALGEMENE SPECIFICATIES>
```

betekenis

Met het sleutelwoord LOAD kan een eerder met CALL SAVE op schijf geplaatst programma weer in het computergeheugen worden geladen. Wanneer de ,R-optie wordt toegepast, wordt het programma onmiddellijk uitgevoerd.

Enkele voorbeelden:

```
CALL LOAD("PR01.BAS")
```
het programma PR01.BAS wordt in het geheugen geladen

```
CALL LOAD("QD2:PR",R)
```
het programma PR wordt van de derde aangesloten QD eenheid geladen en onmiddellijk uitgevoerd.

SLEUTELWOORD # CALL RUN

moeilijkheidsgraad . . . . . **normaal**
soort . . . . . **KOMMANDO**
afkomst . . . . . **RUN is werken/lopen/rennen**

schrijfwijze

```
CALL{"}RUN{"}(<BESTANDSNAAM>[)]
<BESTANDSNAAM>::=<A>
<A>::=<ZIE ALGEMENE SPECIFICATIES>
```

betekenis

Met het sleutelwoord RUN kan een BASIC-programma of een MACHINETAAL-programma in het geheugen van de computer worden geladen en onmiddellijk worden uitgevoerd.

Enkele voorbeelden:

```
CALL RUN("PR01")
```
het programma PR01 wordt van de eerste aangesloten QD eenheid geladen en uitgevoerd

```
CALL RUN("QD3:PR.BIN")
```
vanaf QD eenheid nummer vier wordt het programma PR.BIN geladen en onmiddellijk uitgevoerd.

SLEUTELWOORD # CALL MERGE

moeilijkheidsgraad ..... **normaal**
soort ..... **KOMMANDO**
afkomst ..... **MERGE is samenvoegen**

schrijfwijze

```
CALL{"}MERGE{"}[(<BESTANDSNAAM>[)]]
<BESTANDSNAAM>::=<A>
<A>::=<ZIE ALGEMENE SPECIFICATIES>
```

betekenis

Met het sleutelwoord MERGE kan een programma dat op schijf staat BIJ een in het computergeheugen aanwezig programma worden geladen. Voorwaarde is, dat het bij te laden programma met de ,A-optie eerder werd gesaved.

Een voorbeeld:

```
NEW
Ok
10 PRINT "MERGE";
20 STOP
CALL SAVE("P1",A)
Ok
NEW
Ok
15 PRINT "TESTPROGRAMMA"
20 END
CALL MERGE("P1")
Ok
LIST
10 PRINT "MERGE";
15 PRINT "TESTPROGRAMMA"
20 STOP
Ok
RUN
```

```
MERGETESTPROGRAMMA
Break in 20
Ok
```

Eerst werd een klein programma geschreven en met de ‚A-optie op schijf gezet. Vervolgens werd het computergeheugen schoongemaakt en een tweede programma geschreven. Uiteindelijk werd het eerder geschreven programma bijgevoegd.

Merk op dat wanneer een programmaregel in het bij te voegen programma hetzelfde regelnummer heeft als een regel die zich reeds in het geheugen bevindt, de regel uit het bij te voegen programma de eerder in het computergeheugen opgenomen regel vervangt.

Met het kommando:

```
CALL MERGE
```

of

```
CALL MERGE("QD:")
```

kan een programma dat zonder naam werd vastgelegd, worden bijgevoegd in een bestaand programma (zie de behandeling van CALL SAVE). Bij veel gebruikte programma-onderdelen die vaak moeten worden bijgevoegd, kan dit verschijnsel, eigenlijk een klein foutje in de computertaal, misschien handig zijn.

SLEUTELWOORD # CALL BSAVE

moeilijkheidsgraad . . **zeer moeilijk, kennis van machinetaal en opbouw van computergeheugen is vaak een vereiste**

soort . . . . . **KOMMANDO**

afkomst . . . . . **BSAVE is een afkorting van binary save – binair bewaren**

schrijfwijze

```
CALL{"}BSAVE{"}(<BESTANDSNAAM>,<EERSTE
  BYTE>,<LAATSTE BYTE>[,<STARTADRES>
  !S][)]
<BESTANDSNAAM>:::=<A>
<EERSTE BYTE>:::=<N>
<LAATSTE BYTE>:::=<N>
<STARTADRES>:::=<N>
<A>:::=<ZIE ALGEMENE SPECIFICATIES>
<N>:::=<ZIE ALGEMENE SPECIFICATIES>
```

betekenis

Met CALL BSAVE kan een stuk computergeheugen rechtstreeks op schijf worden gezet. Dit stuk computergeheugen kan een machinetaalprogramma bevatten, maar kan ook andere gegevens bevatten.

Door het adres van het eerste byte en het adres van het laatste byte aan te geven, wordt het geheugengedeelte bepaald. Bij een machinetaalprogramma kan eventueel een startadres worden meegegeven. Wanneer dit programma dan later met CALL RUN of CALL BLOAD wordt geladen, weet de computer onmiddellijk waar het allereerst uit te voeren machinetaalbevel zich binnen het machinetaalprogramma bevindt.

Met de ‚S-optie bewerkstelligt men dat niet het 'gewone' computergeheugen maar het beeldschermgeheugen wordt veilig gesteld. Zo kunnen grafische voorstellingen op beeldscherm bijvoorbeeld op schijf worden gezet en later weer geladen.

Enkele voorbeelden:

```
CALL BSAVE("QD:FILE.BIN",32768,65535)
```
het gehele RAM-geheugen wordt op schijf gezet

```
CALL BSAVE("SCREEN",0,16383,S)
```
het gehele beeldschermgeheugen wordt op schijf gezet

```
CALL BSAVE("SPEL.BIN",40000,40256,40250)
```
een machinetaalprogramma wordt op schijf gezet. Ook het startadres wordt bepaald.

Een heel interessant kommando is het volgende:

```
CALL BSAVE("QDROM",16384,24480)
```

Doordat tijdens het veilig stellen het ROM-geheugen van de QD eenheid aktief moet zijn, wordt door dit kommando de 8 kilobyte ROM die voor de besturing van de QD eenheid zorgt, op schijf gezet. De machinetaal-expert kan dan later dit stuk machinetaalprogramma uitpluizen en zo precies te weten komen hoe de QD eenheid werkt.

De diverse adresgegevens mogen niet kleiner zijn dan –32768 en niet groter zijn dan 65535. Indien een adresgegeven negatief blijkt te zijn, wordt voorafgaand aan de uitvoering van het betreffende kommando eerst 65536 bij dit adresgegeven geteld. Decimale frakties worden altijd verwaarloosd.

moeilijkheidsgraad .. **zeer moeilijk, kennis van machinetaal en opbouw van computergeheugen is vaak een vereiste**

soort ..... **KOMMANDO**

afkomst ..... **BLOAD is een afkorting van binary load – binair laden**

schrijfwijze

```
CALL(")BLOAD(")(<BESTANDSNAAM>
   [ ,R[,<VERSCHUIVING1>]
   [ ,S[,<VERSCHUIVING2>]  ][)]
<VERSCHUIVING1>::=<N>
<VERSCHUIVING2>::=<N>\R<...>\S<...>
<BESTANDSNAAM>::=<A>
<N>::=<ZIE ALGEMENE SPECIFICATIES>
<A>::=<ZIE ALGEMENE SPECIFICATIES>
<...>::=<ZIE ALGEMENE SPECIFICATIES>
```

betekenis

Een eerder met CALL BSAVE op schijf geplaatste geheugeninhoud kan met CALL BLOAD weer worden geladen. Wanneer verder geen opties of verschuivingen worden opgegeven, wordt de geheugeninhoud gelezen zoals hij eerder werd geschreven.

Wanneer de ,R-optie werd gespecificeerd, dan wordt direkt na het laden vervolgd met de uitvoering van het machinetaalkommando op het startadres dat eerder met CALL SAVE werd gespecificeerd.

Indien een verschuiving werd gespecificeerd, worden de eerder met CALL BSAVE opgegeven adressen (eerste byte, laatste byte, startadres) verhoogd met de opgegeven verschuiving voordat de geheugeninhoud wordt geladen. Indien door de opgegeven verschuiving één van de adressen de waarde 65535 overschrijdt, dan wordt dit adres met de waarde 65536 verlaagd. Indien door de opgegeven verschuiving een

gedeelte van de geheugeninhoud over de grens van 65535 zou worden ingelezen, dan wordt dit gedeelte verder vanaf geheugenlokatie 0 ingelezen.

Met de ,S-optie kan een geheugeninhoud van het video-geheugen worden ingelezen.

Een verschuiving mag niet kleiner dan –32768 en niet groter dan 65535 worden opgegeven. Indien een verschuiving kleiner dan 0 wordt opgegeven, dan wordt voorafgaand aan het laden automatisch 65536 bij deze waarde bijgeteld.

In het volgende voorbeeld wordt een beeldschermgeheugen op schijf gezet en later weer gelezen.

```
NEW
Ok
10 REM ZET SCHERM OP SCHIJF
20 SCREEN 2:FOR I=1 TO 20
30 CIRCLE (RND(1)*200,RND(1)*200),RND(1)
*100
40 NEXT I
50 CALL BSAVE("SCR",0,16383,S)
60 STOP
RUN

(cirkels worden getekend en het
 scherm wordt op schijf gezet)

Break in 60
Ok


NEW
Ok
10 REM HAAL SCHERM VAN SCHIJF
20 SCREEN 2:CALL BLOAD("SCR",S)
30 GOTO 30
RUN
```

(het beeldscherm wordt weer gela-
den; heel plotseling ontstaat
weer het eerder ontworpen beeld)

SLEUTELWOORD # CALL QDKEY

moeilijkheidsgraad . . . . . **eenvoudig**
soort . . . . . **KOMMANDO**
afkomst . . . . . **QDKEY is een afkorting van quick disk keys – quick disk toetsen**

schrijfwijze

```
CALL{"}QDKEY{"}[(<N>[)]]
<N>::=<ZIE ALGEMENE SPECIFICATIES>
```

betekenis

De funktietoetsen van de MSX-computer hebben bij het inschakelen van de computer een bepaalde betekenis. Wanneer de QD eenheid is aangeschakeld, worden ook de standaard betekenissen van deze funktietoetsen gewijzigd. Met CALL QDKEY kunnen de volgende situaties worden bewerkstelligd:

CALL QDKEY(0)	de standaardfunkties voor de funktietoetsen worden aktief
CALL QDKEY(1)	de QD-funkties worden geaktiveerd voor de funktietoetsen
CALL QDKEY	de set van funkties wordt gewisseld. Waren de standaardfunkties aktief, dan worden de QD funkties aktief en vice versa.

De waarde tussen haakjes dient groter te zijn dan –32769 en kleiner dan 65536. Een decimale fraktie wordt verwaarloosd. Indien de waarde tussen haakjes ongelijk is aan nul, wordt de QD-funktieset geaktiveerd; indien de waarde tussen haakjes gelijk is aan nul, wordt de standaardset geaktiveerd.

De funktietoetsen hebben standaard de volgende betekenissen:

standaard funktieset	QD funktieset
1 color	_RUN
2 auto	_LOAD
3 goto	_BLOAD
4 list	list

```
5  run                 run
6  color 15,4,4        color 15,4,7
7  _QDKEY              _QDKEY
8  cont                _SAVE("QD0:
9  list.               _BSAVE("QD0:
10 run                 _QDFILES
```

Merk op dat niet de gehele standaardset wordt geaktiveerd. Funktietoets 7 blijft afwijken van de standaard funktietoetsindeling die MSX hanteert wanneer geen QD eenheid is aangesloten.

6 FOUTMELDINGEN OP VOLGORDE VAN NUMMER

Het MSX QD BASIC kent standaard de volgende foutmeldingen, hier op nummer opgenomen:

01 NEXT without FOR	gepoogd werd om een NEXT uit te voeren zonder dat een bijbehorende FOR werd uitgevoerd.
02 Syntax error	er werd een fout in de schrijfwijze ontdekt.
03 RETURN without GOSUB	gepoogd werd om een RETURN uit te voeren terwijl er niet eerder een GOSUB werd uitgevoerd.
04 Out of DATA	gepoogd werd om een READ uit te voeren terwijl er geen DATA meer kon worden gevonden.
05 Illegal function call	er werd gepoogd om een kommando of funktie uit te voeren met niet toegestane waarden.
06 Overflow	het resultaat van een berekening ligt boven het toegestane maximum of onder het toegestane minimum.
07 Out of memory	er werd gepoogd om een kommando uit te voeren dat meer geheugen nodig heeft dan er beschikbaar is.
08 Undefined line number	een niet bestaand programmaregelnummer werd benaderd.
09 Subscript out of range	een array-variabele werd buiten zijn dimensies aangesproken of er werd een verkeerd aantal dimensies genoemd.
10 Redimensioned array	er werd gepoogd om een array-variabele voor een tweede keer te DIMensioneren.
11 Division by zero	gepoogd werd om een deling door nul te doen of een negatieve macht van nul te berekenen.
12 Illegal direct	er werd gepoogd om een kommando

	direkt in te tikken terwijl dit kommando slechts in een programmaregel mag voorkomen.
13 Type mismatch	een numerieke en een alfanumerieke uitdrukking werden met elkaar verwisseld.
14 Out of string space	gepoogd werd om een kommando uit te voeren waardoor het stringgeheugen te klein werd. Vergroot het string-geheugen met CLEAR.
15 String too long	er werd gepoogd om een string samen te stellen die langer werd dan 255 posities.
16 String formula too complex	de alfanumerieke uitdrukking die werd uitgewerkt is te ingewikkeld voor MSX-basic. Splits deze uitdrukking in enkele kleinere.
17 Can't continue	gepoogd werd on een programma met CONT te vervolgen terwijl dat niet (meer) gaat.
18 Undefined user function	gepoogd werd om met FN een niet gedefinieerde funktie aan te roepen.
19 Device I/O error	een fout werd ontdekt tijdens het lezen van/schrijven naar een randapparaat (cassetterecorder of printer e.d.).
20 Verify error	tijdens de controle met CLOAD? werd een verschil ontdekt tussen het programma op cassetteband en het programma in het geheugen.
21 No RESUME	de ERROR-routine werd aangeroepen en een RESUME kon niet worden gevonden.
22 RESUME without error	er werd gepoogd om een RESUME uit te voeren terwijl er geen fout via de ON ERROR GOTO werd gedetecteerd.
24 Missing operand	een noodzakelijke uitdrukking wordt in een kommando niet gevonden.
25 Line buffer overflow	een ingetikte programmaregel

	is te lang voor MSX-basic (langer dan 255 karakters).
50 FIELD overflow	deze foutmelding is wel aanwezig maar kan in het standaard MSX-basic niet voorkomen tenzij met ERROR 50 gegenereerd.
51 Internal error	deze melding wijst op een fout die niet zou mogen kunnen voorkomen. Licht uw computerleverancier of MICROSOFT in.
52 Bad file number	een niet toegestaan kanaalnummer werd gebruikt.
53 File not found	deze foutmelding is wel aanwezig maar kan in het standaard MSX-basic niet voorkomen tenzij met error 53 gegenereerd.
54 File already open	deze foutmelding is wel aanwezig maar kan in het standaard MSX-basic niet voorkomen tenzij met error 54 gegenereerd.
55 Input past end	er werd gepoogd om nog gegevens uit een bestand te lezen terwijl dit bestand reeds volledig was doorgewerkt.
56 Bad file name	een bestandsnaam werd niet volgens de voorschriften samengesteld.
57 Direct statement in file.	tijdens het LOADen van een programma werd een direkt kommando (zonder regelnummer) ontdekt: LOAD werd gestopt.
58 Sequential I/O only	deze foutmelding is wel aanwezig maar kan in het standaard MSX-basic niet voorkomen tenzij met ERROR 58 gegenereerd.
59 File not open	een kanaal werd aangesproken zonder dat hierop een bestand geOPENd is.

Indien een niet bestaand foutnummer met ERROR wordt genegeerd, zal de foutmelding 'Unprintable error' verschijnen. Deze foutmeldingen kunnen door de programmeur een bepaalde betekenis worden

7 FOUTMELDINGEN OP ALFABETISCHE VOLGORDE

Het MSX QD BASIC kent standaard de volgende foutmeldingen, hier op alfabetische volgorde opgenomen:

56 Bad file name	een bestandsnaam werd niet volgens de voorschriften samengesteld.
52 Bad file number	een niet toegestaan kanaalnummer werd gebruikt.
17 Can't continue	gepoogd werd on een programma met CONT te vervolgen terwijl dat niet (meer) gaat.
19 Device I/O error	een fout werd ontdekt tijdens het lezen van/schrijven naar een randapparaat (cassetterecorder of printer e.d.).
57 Direct statement in file	tijdens het LOADen van een programma werd een direkt kommando (zonder regelnummer) ontdekt; LOAD werd gestopt.
11 Division by zero	gepoogd werd om een deling door nul te doen of een negatieve macht van nul te berekenen.
50 FIELD overflow	deze foutmelding is wel aanwezig maar kan in het standaard MSX-basic niet voorkomen tenzij met ERROR 50 gegenereerd.
54 File already open	deze foutmelding is wel aanwezig maar kan in het standaard MSX-basic niet voorkomen tenzij met error 54 gegenereerd.
53 File not found	deze foutmelding is wel aanwezig maar kan in het standaard MSX-basic niet voorkomen tenzij met error 53 gegenereerd.
59 File not open	een kanaal werd aangesproken zonder dat hierop een bestand geOPENd is.
12 Illegal direct	er werd gepoogd om een kommando direkt in te tikken terwijl dit

	kommando slechts in een programmaregel mag voorkomen.
05 Illegal function call	er werd gepoogd om een kommando of funktie uit te voeren met niet toegestane waarden.
55 Input past end	er werd gepoogd om nog gegevens uit een bestand te lezen terwijl dit bestand reeds volledig was doorgewerkt.
51 Internal error	deze melding wijst op een fout die niet zou mogen kunnen voorkomen. Licht uw computerleverancier of MICROSOFT in.
25 Line buffer overflow	een ingetikte programmaregel is te lang voor MSX-basic (langer dan 255 karakters).
24 Missing operand	een noodzakelijke uitdrukking wordt in een kommando niet gevonden.
01 NEXT without FOR	gepoogd werd om een NEXT uit te voeren zonder dat een bijbehorende FOR werd uitgevoerd.
21 No RESUME	de ERROR-routine werd aangeroepen en een RESUME kon niet worden gevonden.
04 Out of DATA	gepoogd werd om een READ uit te voeren terwijl er geen DATA meer kon worden gevonden.
07 Out of memory	er werd gepoogd om een kommando uit te voeren dat meer geheugen nodig heeft dan er beschikbaar is.
14 Out of string space	gepoogd werd om een kommando uit te voeren waardoor het stringgeheugen te klein werd. Vergroot het string-geheugen met CLEAR.
06 Overflow	het resultaat van een berekening ligt boven het toegestane maximum of onder het toegestane minimum.
22 RESUME without error	er werd gepoogd om een RESUME uit te voeren terwijl er geen

	fout via de ON ERROR GOTO werd gedetecteerd.
03 RETURN without GOSUB	geprobeerd werd om een RETURN uit te voeren terwijl er niet eerder een GOSUB werd uitgevoerd.
10 Redimensioned array	er werd geprobeerd om een array-variabele voor een tweede keer te DIMensioneren.
58 Sequential I/O only	deze foutmelding is wel aanwezig maar kan in het standaard MSX-basic niet voorkomen tenzij met ERROR 58 gegenereerd.
16 String formula too complex	de alfanumerieke uitdrukking die werd uitgewerkt is te ingewikkeld voor MSX-basic. Splits deze uitdrukking in enkele kleinere.
15 String too long	er werd geprobeerd om een string samen te stellen die langer werd dan 255 posities.
09 Subscript out of range	een array-variabele werd buiten zijn dimensies aangesproken of er werd een verkeerd aantal dimensies genoemd.
02 Syntax error	er werd een fout in de schrijfwijze ontdekt.
13 Type mismatch	een numerieke en een alfanumerieke uitdrukking werden met elkaar verwisseld.
08 Undefined line number	een niet bestaand programmaregelnummer werd benaderd.
18 Undefined user function	geprobeerd werd om met FN een niet gedefinieerde funktie aan te roepen.
20 Verify error	tijdens de controle met CLOAD? werd een verschil ontdekt tussen het programma op cassetteband en het programma in het geheugen.

Indien een niet bestaand foutnummer met ERROR wordt genegeerd, zal de foutmelding 'Unprintable error' verschijnen. Deze foutmeldingen kunnen door de programmeur een bepaalde betekenis worden toegedacht.

8 FRAME, EEN BASISPROGRAMMA VOOR BESTANDSONDERHOUD

In dit hoofdstuk is de lijst opgenomen van het programma FRAME.

Het programma FRAME is een basisprogramma waarvan iedereen binnen enkele minuten zijn eigen onderhoudsprogramma's kan maken.

Om een eigen bestandsonderhoudsprogramma samen te stellen, behoeft men slechts de volgende handeling te verrichten:

1. Laad het programma FRAME (of tik het de eerste keer in).

2. Breng op regel 140 in een DATA-kommando de naam van het te onderhouden bestand in.

3. Neem op regel 190 in een DATA-kommando het aantal records in dat het bestand dient te bevatten.

4. Breng op regel 250-490 de gegevens per veld in. Per veld steeds de veldnaam in een DATA-kommando inbrengen, gevolgd door de maximale lengte van het veld.

5. Leg het programma tenslotte eerst op schijf vast en test daarna of het goed werkt. Het kan zijn dat de recordlengte groter is dan 255 bytes of dat een veldomschrijving te lang is; kortom, er kunnen kleine schoonheidsfoutjes optreden.

In de lijst zijn wat voorbeeldgegevens op de in te vullen regelnummers opgenomen; deze mogen natuurlijk worden vervangen of verwijderd.

Vanaf regel 60000 is een ingaveroutine opgenomen die fool-proof is. Ook in vele andere programma's kan deze routine zijn nut bewijzen.

De handige programmeur kan het programma misschien later aanpassen, zodat er ook een sorteer- en printmogelijkheid is. Terwille van de eenvoud zijn deze mogelijkheden achterwege gelaten.

Het programma FRAME is in een TOP-DOWN struktuur opgebouwd en doorspekt met dokumentatie. De iets meer geoefende amateur moet de werking van dit programma toch wel grotendeels kunnen doorgronden.

De werking van FRAME is vrij vanzelfsprekend. Steeds dient éérst een recordnummer te worden ingegeven, gevolgd door de betreffende gegevens.
Onthoud echter het volgende:

INGAVE van	RESULTAAT/FUNKTIE
RETURN	de inhoud van een veld blijft onveranderd.
– (min-toets, gevolgd door return)	ga terug naar de vorige ingave.
–– (twee maal min)	maak het veld schoon.
––– (drie maal min)	verwijder het record.

FRAME is in staat om bestanden tot een grootte van 16K te verzorgen. Wanneer het programma wordt opgestart, wordt het bestand eerst ingelezen. Dit kan enkele ogenblikken duren.

Pas helemaal aan het einde van het programma (keuze 3) wordt het oude bestand van de schijf verwijderd en een nieuw bestand met de laatste gegevens aangelegd. Dit betekent dat tussentijdse onderbreking van het programma altijd tot gevolg heeft dat de laatste veranderingen en toevoegingen niet zijn verwerkt.

Nadat FRAME op een QD floppy is geïnstalleerd, mogen geen andere bestanden meer op deze floppy worden gezet of andere programma's meer worden gesaved; het te onderhouden bestand moet het laatste op schijf geplaatste bestand zijn.


```
10 REM ************************
20 REM * ALGEMEEN BESTANDSVER- *
30 REM * ZORGINGSPROGRAMMA     *
40 REM * VOOR EEN MSX-COMPUTER *
```

```
50 REM * MET QUICK DISK.        *
60 REM * (C)1985 STARK TEXEL    *
70 REM ***************************
80 REM
90 CLEAR 17000'MAX. ONG. 16KB FILES
100 REM ------------------------
110 REM NEEM OP REGEL 140 OP:
120 REM 140 DATA "<file>"
130 REM ------------------------
140 DATA "KENNIS"
150 REM ------------------------
160 REM NEEM OP REGEL 190 op:
170 REM 190 DATA <aantal records>
175 REM (NOOIT MEER DAN 1500!)
180 REM ------------------------
190 DATA 1500
200 REM ------------------------
210 REM NEEM OP REGEL 250 EN VER-
220 REM DER OP (MAX. 20 KEER):
230 REM XXX DATA "<veld>",<lang>
240 REM ------------------------
250 DATA "NAAM",24
260 DATA "STRAAT",24
270 DATA "POSTKODE",6
280 DATA "WOONPLAATS",24
290 DATA "TELEFOON",15
300 DATA "GEBOORTEDATUM",8
310 DATA "HOBBIES",24
320 DATA "LEEFTIJD",2
330 DATA "LIEVELINGSKLEUR",8
340 DATA "LIEVELINGSDRANK",12
350 DATA "LAATSTE VISITE",8
360 DATA "OPMERKINGEN",24
500 REM ------------------------
510 DATA ""
520 REM ***************************
```

```
530 REM * HIER BEGINT HET FEI-  *
540 REM * TELIJKE PROGRAMMA PAS *
550 REM ************************
560 REM
570 REM ************************
580 REM *     HOOFDPROGRAMA      *
590 REM *     -------------      *
600 REM * VANUIT DIT HOOFDPRO-  *
610 REM * GRAMMA WORDEN ALLE    *
620 REM * FUNKTIES AANGEROEPEN  *
625 REM ************************
630 REM
640 GOSUB 1000'INITIALISATIE
650 GOSUB 1500'PROGRAMMAKEUZE
660 REM NAAR DE DRIE PROGRAMMA'S
670 ON K GOSUB 2000,3000,4000
680 GOTO 650' EN WEER PROGRAMMAKEUZE
1000 REM ************************
1010 REM *     INITIALISATIE      *
1020 REM *     -------------      *
1030 REM * HIER WORDEN DE VASTE  *
1040 REM * WAARDEN BEPAALD EN    *
1050 REM * HET BESTAND TOEGEWE-  *
1060 REM * ZEN OF GEOPEND.       *
1070 REM ************************
1080 REM
1090 REM BESTAND TOEWIJZEN
1100 RESTORE:READ F$,LR:F$="QD:"+F$
1110 ON ERROR GOTO 1220:OPEN F$ FOR INPU
T AS #1:ON ERROR GOTO
1120 REM VELDOMSCHRIJVINGEN BEPALEN
1130 DIM V$(21),L(20),VI$(20)
1140 RL=0:FOR I=1 TO 20:READ V$(I)
1150 IF V$(I)<>"" THEN READ L(I):RL=RL+L
(I):NEXT I
1160 REM RECORDS INLEZEN
```

```
1170 DIM R$(LR):FOR I=1 TO LR:IF EOF(1)=
0 THEN LINE INPUT #1,R$(I):NEXT I
1210 CLOSE:RETURN
1220 OPEN F$ FOR OUTPUT AS #1:CLOSE:RESU
ME
1500 REM ************************
1510 REM *    PROGRAMMAKEUZE     *
1520 REM *    -------------      *
1530 REM * HIER WORDT DE PRO-    *
1540 REM * RAMMAKEUZE GEREGELD   *
1550 REM ************************
1560 REM
1570 K$="PROGRAMMAKEUZE"
1580 GOSUB 55000'CLS/KOP
1590 LOCATE 0,3
1610 PRINT "1=AANMAKEN/WIJZIGEN RECORDS"
1620 PRINT "2=OPVRAGEN RECORDS"
1630 PRINT "3=EINDE PROGRAMMA"
1640 I$="000901UW KEUZE"
1650 GOSUB 60000'INPUT
1660 IF II$<>"1" AND II$<>"2" AND II$<>"
3" THEN 1650'FOUT
1670 K=VAL(II$):RETURN
2000 REM ************************
2010 REM *   AANMAKEN/WIJZIGEN   *
2020 REM *   -----------------   *
2030 REM * HIER WORDT DE AANMAAK *
2040 REM * EN WIJZIGING OF VER-  *
2050 REM * WIJDERING VAN DE RE-  *
2060 REM * CORDS GEREGELD.       *
2070 REM ************************
2080 REM
2090 K$="AANMAKEN/WIJZIGEN RECORDS"
2100 GOSUB 55000'CLS/KOP
2110 GOSUB 50000'PRINT VELDEN
2120 GOSUB 45000'INGAVE/LEES RECORD
```

```
2125 IF RN=0 THEN RETURN'0=EINDE
2130 REM NU INGAVE VELDEN
2140 FOR I=1 TO 20:IF V$(I)<>"" THEN I$=
RIGHT$(STR$(MV+101),2)+RIGHT$(STR$(I+103
),2)+RIGHT$(STR$(L(I)+100),2):GOSUB 6000
0 ELSE 2170'INPUT
2150 IF II$=STRING$(L(I)," ") THEN II$=V
I$(I)
2155 IF LEFT$(II$,3)="---" THEN R$(RN)="
":GOTO 2100
2156 IF LEFT$(II$,2)="--" THEN II$=SPACE
$(L(I)):GOTO 2160
2157 IF LEFT$(II$,1)="-" THEN LOCATE MV+
1,I+3:PRINT VI$(I);:I=I-2:GOTO 2165
2160 VI$(I)=II$:LOCATE MV+1,I+3:PRINT II
$;
2165 IFI=-1 THEN I=0
2166 NEXT I
2170 GOSUB 40000'SCHRIJF RECORD
2180 GOTO 2120' VOLGENDE RECORD
3000 REM ***************************
3010 REM *    OPVRAGEN RECORDS   *
3020 REM *    ----------------   *
3030 REM * HIER WORDEN DE RE-    *
3040 REM * CORDS OPGEVRAAGD.     *
3050 REM ***************************
3060 REM
3070 K$="OPVRAGEN RECORDS"
3080 GOSUB 55000'CLS/KOP
3090 GOSUB 50000'PRINT VELDEN
3100 GOSUB 45000'INGAVE/LEES RECORD
3110 IF RN=0 THEN RETURN ELSE 3100
4000 REM ***************************
4010 REM *    EINDE PROGRAMMA    *
4020 REM ***************************
4030 REM
```

```
4031 CALL QDKILL(F$):OPEN F$ FOR OUTPUT
AS #1
4032 FOR I=1 TO LR:PRINT#1,R$(I):NEXT I
4040 CLOSE:CLS:STOP
40000 REM ***************************
40010 REM *      SCHRIJF RECORD     *
40020 REM *      ---------------    *
40030 REM * HIER WORDT HET EERDER *
40040 REM * INGELEZEN RECORD WEER *
40050 REM * GESCHREVEN IN RAM     *
40060 REM ***************************
40070 REM
40080 RR$="*":FOR I=1 TO 20:IF V$(I)<>""
 THEN RR$=RR$+VI$(I):NEXT I
40090 R$(RN)=RR$:RETURN
45000 REM ***************************
45010 REM *  INGAVE RECORDNUMMER   *
45020 REM *  ---------------------  *
45030 REM * HIER WORDT HET RE-     *
45040 REM * CORDNUMMER INGEGEVEN   *
45050 REM * EN HET RECORD INGELE- *
45060 REM * ZEN VANUIT RAM EN AF- *
45065 REM * GEDRUKT.               *
45070 REM ***************************
45080 REM
45090 I$="000304RECORD"+STRING$(MV-6," "
)
45100 GOSUB 60000'INPUT
45110 FOR I=1 TO 4:IF (MID$(II$,I,1)<"0"
 OR MID$(II$,I,1)>"9") AND MID$(II$,I,1)
<>" " THEN 45100 ELSE NEXT I
45120 RN=VAL(II$):IF RN=0 THEN 45190
45125 IF RN>LR THEN 45090
45126 LOCATE MV+1,3:PRINT USING"     ###
#";RN
45130 R$=R$(RN):IF LEFT$(R$,1)<>"*" THEN
```

```
 R$="*"+SPACE$(RL)
45140 RP=2:FOR I=1 TO 20:IF V$(I)="" THE
N 45160
45150 VI$(I)=MID$(R$,RP,L(I)):RP=RP+L(I)
:NEXT I
45160 FOR I=1 TO 20:IF V$(I)="" THEN 451
90
45170 LOCATE MV+1,I+3:PRINT VI$(I);:NEXT
 I
45190 RETURN
50000 REM ***************************
50010 REM *       PRINT VELDEN      *
50020 REM *       ------------      *
50030 REM * HIER WORDEN DE VELD-   *
50040 REM * NAMEN AFGEDRUKT.       *
50050 REM ***************************
50060 REM
50070 LOCATE 0,3:MV=6:PRINT "RECORD"
50080 FOR I=1 TO 20
50090 IF V$(I)="" THEN 50120
50100 LOCATE 0,I+3:PRINT V$(I);:IF MV<LE
N(V$(I)) THEN MV=LEN(V$(I))
50110 NEXT I
50120 FOR I=1 TO 21:LOCATE MV,I+2:PRINT"
:";:IF V$(I)<>"" THEN NEXT I
50130 RETURN
55000 REM ***************************
55010 REM *  BEELD SCHOON EN KOP   *
55020 REM *  --------------------  *
55030 REM * HIER WORDT HET BEELD   *
55040 REM * SCHOONGEMAAKT EN DE    *
55050 REM * KOP AFGEDRUKT          *
55060 REM ***************************
55070 REM
55080 KEY OFF:COLOR 15,4,4:WIDTH 40:SCRE
EN 0:CLS
```

```
55090 PRINT STRING$(40,"■")
55100 LOCATE (40-LEN(K$))/2-1,0:PRINT "
";K$;" "
55110 RETURN
60000 REM ************************
60010 REM *    INGAVEROUTINE     *
60020 REM *    ------------      *
60030 REM * I$="HHVVTTA---A"     *
60040 REM * HH=HORIZONTALE POS   *
60050 REM * VV=VERTIKALE POS.    *
60060 REM * TT=AATAL TEKENS.     *
60070 REM * A---A=TEKST INPUT    *
60080 REM * RESULTAAT, AANGEVULD *
60090 REM * MET SPATIES IN II$   *
60100 REM ************************
60110 REM
60120 II$="": XX=VAL(MID$(I$,1,2)):YY=VA
L(MID$(I$,3,2)):TT=VAL(MID$(I$,5,2)):SS$
=MID$(I$,7):IF LEN(SS$) THEN SS$=SS$+":"
60130 LOCATE XX,YY,0:PRINT SS$;STRING$(T
T,"▁");:LOCATE XX+LEN(SS$),YY
60140 KK$=INKEY$:IF KK$="" THEN 60140
60150 IF KK$=CHR$(8) OR ASC(KK$)=127 OR
ASC(KK$)=29 THEN 60210
60160 IF KK$=CHR$(13) THEN 60240
60170 IF ASC(KK$)<32 OR ASC(KK$)>122 THE
N BEEP:GOTO 60140
60180 IF ASC(KK$)>96 AND ASC(KK$)<123 TH
EN LET KK$=CHR$(ASC(KK$)-32)
60190 IF LEN(II$)=TT THEN BEEP:GOTO 6014
0
60200 II$=II$+KK$:PRINT KK$;:GOTO 60140
60210 IF II$="" THEN BEEP:GOTO 60140
60220 II$=LEFT$(II$,LEN(II$)-1):PRINT CH
R$(8);"▁";CHR$(8);:GOTO 60140
60240 II$=II$+STRING$(TT-LEN(II$),32):LO
```

```
CATE XX+LEN(SS$),YY:PRINT II$;:RETURN
60250 RETURN
```

9 TEKEN, EEN MSX-TEKENTAFELPROGRAMMA

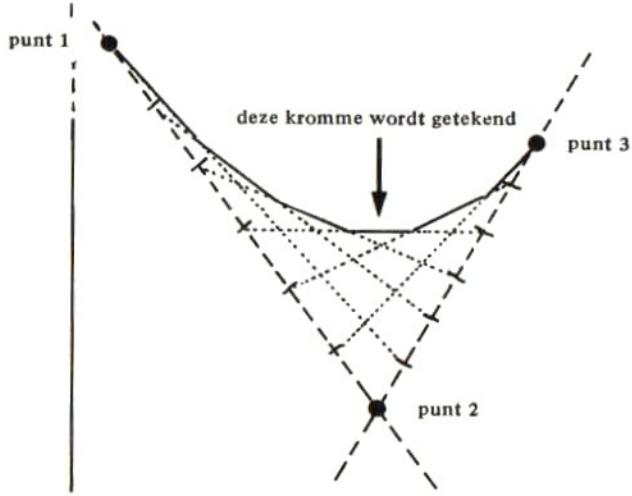
In dit hoofdstuk is de lijst opgenomen van het programma TEKEN.

Het programma TEKEN is een vrij uitgebreid en geavanceerd tekentafelprogramma; technische tekeningen kunnen worden ingebracht, vergroot, verkleind, verschoven en kunnen op schijf worden opgeslagen.

Nadat het programma is opgestart, wordt een leeg beeldscherm gepresenteerd. De volgende kommando's zijn mogelijk;

TOETS	FUNKTIE (zorg dat CAOS LOCK aan staat)
pijl omhoog	verplaatst de tekenpijl naar boven.
pijl naar rechts	verplaatst de tekenpijl naar rechts.
pijl naar beneden	verplaatst de tekenpijl naar beneden.
pijl naar links	verplaatst de tekenpijl naar links.
–	(mintoets) verbetert de laatste aktie.
S	vergroot/verkleint de 'stappen' die de tekenpijl tegelijk over het beeldscherm neemt van 1 naar 5 puntjes of van 5 naar 1 puntje.
R	maakt de tekening opnieuw maar nu met een tekenraster er overheen. Met de R kan dit raster ook weer worden uitgeschakeld.
spatie	met de spatie kan een punt worden geplaatst ter hoogte van de tekenpijl. Deze punt dient om de lokatie voor een latere aktie vast te leggen.
L	trekt een lijn tussen de twee laatste (met spatie) geplaatste punten. Na L blijft alleen de laatste punt aktief; de andere verdwijnen.
C	tekent een cirkel. De laatste twee geplaatste

	punten vormen twee middellijnpunten op de cirkel. Het laatst geplaatste punt blijft aktief, het andere andere wordt verwijderd.
B	tekent een rechthoek. Het eerste geplaatste puntje en het tweede geplaatste puntje vormen de hoeken van een diagonaal van de rechthoek. Alleen het laatste puntje blijft aktief.
P	start veelhoek. Nadat met P een punt geplaatst is, kunnen diverse verbindingslijnen worden getekend om bijvoorbeeld een veelhoek te tekenen. De figuur kan als laatste aktie dan worden gesloten met de D-funktie.
D	maak veelhoek dicht. Het laatste aktieve punt wordt met het eerste via P geplaatste punt verbonden.
Q	teken kromme. Voor deze funktie dienen *drie* punten met spatie op het beeldscherm te zijn geplaatst. De wijze waarop de kromme wordt getekend, laat zich het best in een tekening verduidelijken.



	Op deze wijze kunnen allerlei krommen, ook ellipsen, cirkeldelen etcetera, worden geconstrueerd.
W...	na de letter W dient een cijfer (0 ... 9) of een letter (A ... Z) te worden ingetoetst. Indien deze intoetsing langer dan een seconde op zich laat wachten, wordt deze funktie niet uitgevoerd. De tekening wordt in bestand "TEK....DAT" (op ... staat het ingetoetste teken) opgeslagen.
O...	na de letter O dient net zoals bij het kommando -W- een letter of cijfer te worden ingetoetst. De eerder in "TEK....DAT" opgeslagen tekening wordt weer opgehaald.
A...	na de letter A dient net zoals bij het kommando -W- een letter of cijfer te worden ingetoetst. De eerder in "TEK....DAT" opgeslagen tekening wordt vanaf het laatst geplaatste punt opgehaald en *bij* de reeds aanwezige tekening gevoegd. Zo kunt u een éénmaal getekend detail steeds weer opnieuw, desnoods meerdere malen, automatisch in een tekening bijplaatsen.
X	maakt de gehele aktieve tekening schoon.
G	vergroot de tekening met een faktor twee. Deze funktie kan maximaal drie maal achter elkaar worden gebruikt (8 maal vergroten). Het laatst geplaatste punt blijft op zijn plaats, de rest van de tekening wordt hier omheen uitvergroot. In de vergrote toestand kunnen veranderingen in de tekening worden aangebracht die later bij het verkleinen ook weer worden meeverkleind. Zo kunnen detailtekeningen eerst groot worden ingevoerd en pas later naar de werkelijke grootte worden verkleind.
K	verkleint de tekening met een faktor twee. Deze funktie kan alleen worden gebruikt na een vergroting en brengt de tekening weer naar een eerder, kleiner formaat. Het laatst geplaatste punt blijft op zijn plaats; de rest van de tekening wordt hier om-

	heen verkleind.
V	verschuift de tekening. De gehele tekening wordt verschoven zover als het eerst en het tweede puntje uit elkaar liggen. Zo kan een afgewerkt deel van de tekening (ook in vergrote toestand) uit beeld worden geschoven en kan daarnaast worden verder getekend; het beeldscherm maakt steeds slechts een gedeelte zichtbaar. De V-funktie maakt het mogelijk om tekeningen te maken die véél groter dan het beeldscherm zijn. Het beeldscherm vormt slechts een soort 'venster' waardoor een gedeelte van een tekening zichtbaar is. Zo kan een tekening met een bereik van meer dan 30000 x 30000 beeldschermpuntjes worden gemaakt.
F	kleurt een gedeelte wit in. Vanuit het laatst getekende puntje wordt een met lijnen omsloten oppervlak wit ingekleurd. De maximale ingewikkeldheid van een tekening is beperkt tot 500 akties (lijnen, krommen, cirkels, inkleuringen of blokken). Binnen dit aantal akties kunnen over het algemeen vrij gedetailleerde tekeningen ruim worden gekodeerd. *Let op: SHIFT LOCK moet bij dit programma altijd AAN staan.*

Het programma TEKEN is zo ontworpen dat de handige programmeur zonder veel moeite een aanpassing kan maken voor het aansluiten van een plotter.

Het programma TEKEN is in een TOP-DOWN struktuur geschreven. De berekeningen en gedachtengangen die achter het programma zitten, maken het ondanks alle kommentaar en de duidelijke opbouw toch een moeilijk geheel. Een hele slimme amateur doorgrondt het programma op den duur misschien helemaal.

Het werken met dit programma is een erg leuke ervaring. Al snel bent u in staat om mooie tekeningen te fabriceren.

Wanneer u met TEKEN een fout maakt (u haalt bijvoorbeeld een niet

bestaande tekening op of u probeert een tekening op schijf te zetten terwijl deze al op schijf aanwezig is), klinkt een drietonig waarschuwingssignaal ten teken dat een fout werd geconstateerd. Het is dus zaak om het volume van uw TV-toestel wat open te draaien.

Het programma TEKEN mag op non-commerciële basis worden gekopiëerd ten behoeve van vrienden, kennissen enzovoorts. Het is echter verboden om dit programma op commerciële basis te verhandelen of zonder uitdrukkelijke en schriftelijke toestemming van de uitgever te publiceren.

```
10 REM ***************************
20 REM *     TEKENPROGRAMMA      *
30 REM *     --------------      *
40 REM *  (C)1985 STARK TEXEL    *
50 REM ***************************
60 REM
70 REM ***************************
80 REM *     HOOFDPROGRAMMA      *
90 REM *     --------------      *
100 REM ***************************
110 REM
120 GOSUB 150'INITIALISATIE
130 GOSUB 260'KOMMANDO INTEPRETATOR
140 STOP
150 REM ***************************
160 REM *      INITIALISATIE      *
170 REM *      -------------      *
180 REM ***************************
190 REM
200 COLOR 15,4,4:SCREEN 2,0
210 ON ERROR GOTO 3210
220 DIM P%(500,6),X%(2),Y%(2)
230 SPRITE$(1)=CHR$(128):SPRITE$(2)=CHR$
(128):SPRITE$(3)=CHR$(128):SPRITE$(0)=CH
R$(240)+CHR$(192)+CHR$(160)+CHR$(144):GO
SUB 2980'PIJL
```

```
240 GOSUB 1530'SCHOONMAKEN TEKENING
250 RETURN
260 REM ***************************
270 REM * KOMMANDO INTEPRETATOR *
280 REM * --------------------  *
290 REM ***************************
300 REM
310 C$=CHR$(30)+CHR$(28)+CHR$(31)+CHR$(2
9)+"-SR LCBOWXGKVFDPAQ"
320 K$=INKEY$:IF K$="" THEN 320
330 FOR I%=1 TO LEN(C$):IF MID$(C$,I%,1)
=K$ THEN 340 ELSE NEXT I%:GOTO 320
340 F%=0:ON I% GOSUB 500,570,640,710,780
,860,920,990,1080,1150,1220,1290,1420,15
30,1600,1660,1720,1800,1870,1940,2000,21
40
350 GOSUB 2980'PIJL
360 GOTO 320
370 REM ***************************
380 REM *       CLS/KOP/GRID      *
390 REM *       ------------      *
400 REM ***************************
410 REM
420 CLS
430 LINE (10,10)-(240,10):LINE -(240,180
):LINE -(10,180):LINE -(10,10)
440 PAINT(0,0):GOSUB 2910'KOPTEKST
450 IF GR%=0 THEN 490
460 FOR I%=20 TO 240 STEP 10*VF%
470 LINE (I%,10)-(I%,180),3:IF I%<180 TH
EN LINE (10,I%)-(240,I%),3
480 NEXT I%
490 CLOSE:RETURN
500 REM ***************************
510 REM *       PIJL OMHOOG       *
520 REM *       -----------       *
```

```
530 REM ************************
540 REM
550 Y%=Y%-ST%:IF Y%<-VY% THEN Y%=Y%+ST%
560 RETURN
570 REM ************************
580 REM *   PIJL NAAR RECHTS    *
590 REM *   ----------------    *
600 REM ************************
610 REM
620 X%=X%+ST%:IF X%>230/VF%-VX% THEN X%=
X%-ST%
630 RETURN
640 REM ************************
650 REM *   PIJL NAAR BENEDEN   *
660 REM *   -----------------   *
670 REM ************************
680 REM
690 Y%=Y%+ST%:IF Y%>170/VF%-VY% THEN Y%=
Y%-ST%
700 RETURN
710 REM ************************
720 REM *   PIJL NAAR LINKS     *
730 REM *   ---------------     *
740 REM ************************
750 REM
760 X%=X%-ST%:IF X%<-VX% THEN X%=X%+ST%
770 RETURN
780 REM ************************
790 REM *  MINTOETS=CORRECTIE   *
800 REM *  ------------------   *
810 REM ************************
820 REM
830 IF PP%=0 THEN RETURN ELSE PP%=PP%-1
840 GOSUB 2210'TEKENING
850 RETURN
860 REM ************************
```

```
870 REM * S-TOETS SNEL/LANGZAAM *
880 REM * ---------------------- *
890 REM ***************************
900 REM
910 ST%=1-4*(ST%=1):RETURN
920 REM ***************************
930 REM * R-TOETS RASTER AAN/UIT *
940 REM * ---------------------- *
950 REM ***************************
960 REM
970 GR%=NOT(GR%):GOSUB 2210'HERTEKENEN
980 RETURN
990 REM ***************************
1000 REM * SPATIE=PUNT PLAATSEN  *
1010 REM * --------------------  *
1020 REM ***************************
1030 REM
1040 X%(0)=X%(1):Y%(0)=Y%(1):X%(1)=X%(2)
:Y%(1)=Y%(2):X%(2)=X%:Y%(2)=Y%
1050 PN%=PN%+1+3*(PN%=3):PUTSPRITE PN%,(
(X%+VX%)*VF%+10,(Y%+VY%)*VF%+9),1
1060 IF SP%=0 THEN SP%=1:SX%=X%:SY%=Y%
1070 P%=P%-(P%<3):RETURN
1080 REM ***************************
1090 REM *    L=LIJN TREKKEN     *
1100 REM *    --------------     *
1110 REM ***************************
1120 REM
1130 E%=2:I%=0:GOSUB 3040'PLAATS IN P%
1140 RETURN
1150 REM ***************************
1160 REM *   C=CIRCEL TEKENEN    *
1170 REM *   ----------------    *
1180 REM ***************************
1190 REM
1200 E%=2:I%=1:GOSUB 3040'PLAATS IN P%
```

```
1210 RETURN
1220 REM ************************
1230 REM *    B=BLOK TEKENEN    *
1240 REM *    --------------    *
1250 REM ************************
1260 REM
1270 E%=2:I%=2:GOSUB 3040'PLAATS IN P%
1280 RETURN
1290 REM ************************
1300 REM *  O=OPHALEN TEKENING   *
1310 REM *  -----------------   *
1320 REM ************************
1330 REM
1340 GOSUB 3130'BESTANDSNAAM
1350 IF F$="" THEN RETURN
1360 GOSUB 1530'SCHOONMAKEN
1370 OPEN F$ FOR INPUT AS 1:IF F% THEN R
ETURN ELSE INPUT #1,R$:J%=VAL(R$)
1380 FOR PP%=0 TO J%-1:FOR I%=0 TO 6:INP
UT#1,R$:P%(PP%,I%)=VAL(R$):NEXT I%:GOSUB
 2320'TEKEN EEN DETAIL
1390 NEXT PP%:CLOSE
1400 GOSUB 2910'KOPTEKST
1410 RETURN
1420 REM ************************
1430 REM *  W=WEGSCHRIJVEN TEK.  *
1440 REM *  -------------------  *
1450 REM ************************
1460 REM
1470 GOSUB 3130'BESTANDSNAAM
1480 IF F$="" THEN RETURN
1490 OPEN F$ FOR OUTPUT AS 1:IF F% THEN
RETURN ELSE R$=STR$(PP%):PRINT #1,R$
1500 FOR J%=0 TO PP%-1:FOR I%=0 TO 6
1510 R$=STR$(P%(J%,I%)):PRINT #1,R$
1520 NEXT I%:NEXT J%:CLOSE:RETURN
```

```
1530 REM ************************
1540 REM *  X=SCHOONMAKEN TEK.   *
1550 REM *  -----------------   *
1560 REM ************************
1570 REM
1580 SP%=0:PP%=0:VX%=0:VY%=0:VF%=1:X%=0:
Y%=0:P%=0:PN%=0:ST%=1:PUTSPRITE 1,(-1,-1
):PUTSPRITE 2,(-1,-1):PUTSPRITE 3,(-1,-1
):GOSUB 2210:RETURN
1590 PUTSPRITE 0,(10,10),12:RETURN
1600 REM ************************
1610 REM *  G=VERGROOT MAAL TWEE *
1620 REM *  -------------------- *
1630 REM ************************
1640 REM
1650 IF VF%=8 OR P%=0 THEN RETURN ELSE V
F%=VF%*2:VX%=(VX%-X%(2))/2:VY%=(VY%-Y%(2
))/2:P%=0:SP%=0:GOSUB 2210:PUTSPRITE 1,(
-1,-1):PUTSPRITE2,(-1,-1):PUTSPRITE 3,(-
1,-1):RETURN
1660 REM ************************
1670 REM * V=VERKLEIN MAAL TWEE  *
1680 REM * --------------------  *
1690 REM ************************
1700 REM
1710 IF VF%=1 OR P%=0 THEN RETURN ELSE V
F%=VF%/2:VX%=X%(2)+2*VX%:VY%=Y%(2)+2*VY%
:SP%=0:P%=0:GOSUB 2210:PUTSPRITE 1,(-1,-
1):PUTSPRITE 2,(-1,-1):PUTSPRITE 3,(-1,-
1):RETURN
1720 REM ************************
1730 REM *      V=VERSCHUIF      *
1740 REM *      -----------      *
1750 REM ************************
1760 REM
1770 IF P%<2 THEN RETURN ELSE VX%=VX%+X%
```

```
(2)-X%(1):VY%=VY%+Y%(2)-Y%(1):GOSUB 2210
:PUTSPRITE 1,(-1,-1):PUTSPRITE 2,(-1,-1)
:PUTSPRITE 3,(-1,-1)
1780 X%=X%-X%(2)+X%(1):Y%=Y%-Y%(2)+Y%(1)
:P%=0
1790 RETURN
1800 REM ************************
1810 REM *      F=INKLEUREN       *
1820 REM *      -----------       *
1830 REM ************************
1840 REM
1850 E%=1:I%=3:GOSUB 3040'PLAATS IN P%
1860 RETURN
1870 REM ************************
1880 REM *    D=SLUIT POLYGOON    *
1890 REM *    ----------------    *
1900 REM ************************
1910 REM
1920 E%=1:I%=4:GOSUB 3040' PLAATS IN P%
1930 RETURN
1940 REM ************************
1950 REM *    P=START POLYGOON    *
1960 REM *    ----------------    *
1970 REM ************************
1980 REM
1990 SP%=0:GOSUB 990:RETURN
2000 REM ************************
2010 REM * A=BIJVOEGEN TEKENING  *
2020 REM * -------------------  *
2030 REM ************************
2040 REM
2050 IF P%=0 THEN RETURN
2060 GOSUB 3130'BESTANDSNAAM
2070 IF F$="" THEN RETURN
2080 OPEN F$ FOR INPUT AS 1:IF F% THEN R
ETURN ELSE INPUT #1,R$:J%=VAL(R$)
```

```
2090 FOR PP%=PP% TO PP%+J%-1:FOR I%=0 TO
 6:INPUT#1,R$:P%(PP%,I%)=VAL(R$):NEXT I%
:P%(PP%,0)=P%(PP%,0)+X%(2):P%(PP%,1)=P%(
PP%,1)+Y%(2):P%(PP%,2)=P%(PP%,2)+X%(2):P
%(PP%,3)=P%(PP%,3)+Y%(2)
2100 P%(PP%,4)=P%(PP%,4)+X%(2):P%(PP%,5)
=P%(PP%,5)+Y%(2):GOSUB 2320'TEKEN EEN DE
TAIL
2110 NEXT PP%:CLOSE
2120 GOSUB 2910'KOPTEKST
2130 RETURN
2140 REM ***************************
2150 REM *  K=TEKEN EEN KROMME    *
2160 REM *  ------------------    *
2170 REM ***************************
2180 REM
2190 E%=3:I%=5:GOSUB 3040'PLAATS IN P%
2200 RETURN
2210 REM ***************************
2220 REM *  TEKEN HELE TEKENING   *
2230 REM *  -------------------   *
2240 REM ***************************
2250 REM
2260 GOSUB 370'KOP EN RASTER
2270 IF PP%=0 THEN RETURN
2280 QQ%=PP%:FOR PP%=0 TO QQ%-1
2290 GOSUB 2320'TEKEN 1 FIGUUR
2300 NEXT PP%:PP%=QQ%:GOSUB 2910'KOPTEKS
T
2310 RETURN
2320 REM ***************************
2330 REM *     TEKEN EEN DETAIL    *
2340 REM *     ---------------     *
2350 REM ***************************
2360 REM
2370 ON P%(PP%,6)+1 GOSUB 2390,2450,2510
```

```
,2570,2660,2730
2380 RETURN
2390 REM ***************************
2400 REM *     TEKEN EEN LIJN      *
2410 REM *     --------------      *
2420 REM ***************************
2430 REM
2440 LINE ((P%(PP%,2)+VX%)*VF%+10,(P%(PP
%,3)+VY%)*VF%+10)-((P%(PP%,4)+VX%)*VF%+1
0,(P%(PP%,5)+VY%)*VF%+10):RETURN
2450 REM ***************************
2460 REM *    TEKEN EEN CIRCEL     *
2470 REM *    ----------------     *
2480 REM ***************************
2490 REM
2500 CIRCLE (((P%(PP%,4)+P%(PP%,2))/2+VX
%)*VF%+10,((P%(PP%,5)+P%(PP%,3))/2+VY%)*
VF%+10),VF%*SQR((P%(PP%,2)-P%(PP%,4))^2+
(P%(PP%,3)-P%(PP%,5))^2)/2:RETURN
2510 REM ***************************
2520 REM *     TEKEN EEN BLOK      *
2530 REM *     --------------      *
2540 REM ***************************
2550 REM
2560 LINE ((P%(PP%,2)+VX%)*VF%+10,(P%(PP
%,3)+VY%)*VF%+10)-((P%(PP%,4)+VX%)*VF%+1
0,(P%(PP%,5)+VY%)*VF%+10),,B:RETURN
2570 REM ***************************
2580 REM *      VUL EEN VLAK       *
2590 REM *      ------------       *
2600 REM ***************************
2610 REM
2620 XF%=(P%(PP%,4)+VX%)*VF%+10:YF%=(P%(
PP%,5)+VY%)*VF%+10
2630 IF XF%<10 OR XF%>240 OR YF%<10 OR Y
F%>180 THEN 2650
```

```
2640 PAINT (XF%,YF%)
2650 RETURN
2660 REM ***************************
2670 REM *     SLUIT POLYGOON      *
2680 REM *     --------------      *
2690 REM ***************************
2700 REM
2710 SP%=0:P%(PP%,2)=SX%:P%(PP%,3)=SY%:P
%(PP%,6)=0:GOSUB 2320'TEKEN EEN DETAIL
2720 RETURN
2730 REM ***************************
2740 REM *     KROMME TEKENEN      *
2750 REM *     --------------      *
2760 REM ***************************
2770 REM
2780 AX=P%(PP%,0):AY=P%(PP%,1):BX=P%(PP%
,2):BY=P%(PP%,3):CX=P%(PP%,4):CY=P%(PP%,
5):D=(BX-AX)^2+(BY-AY)^2:D1=(CX-BX)^2+(C
Y-BY)^2:IF D>D1 THEN D=D1
2790 IF D=0 THEN 2900 ELSE D=SQR(D)/2
2800 DX=(BX-AX)/D:DY=(BY-AY)/D:EX=(CX-BX
)/D:EY=(CY-BY)/D:BX=BX+EX:BY=BY+EY:PSET
((AX+VX%)*VF%+10,(AY+VY%)*VF%+10):GOTO 2
880
2810 Q1=BX-AX:IF Q1=0 THEN 2870
2820 Q1=(BY-AY)/Q1:Q2=AY-Q1*AX
2830 Q3=BX+EX-AX-DX:IF Q3=0 THEN 2870
2840 Q3=(BY+EY-AY-DY)/Q3:Q4=AY+DY-Q3*(AX
+DX)
2850 IF Q1-Q3=0 THEN 2870 ELSE Q3=(Q4-Q2
)/(Q1-Q3)
2860 LINE -((Q3+VX%+.5)*VF%+10,(Q1*Q3+Q2
+VY%+.5)*VF%+10)
2870 BX=BX+EX:BY=BY+EY:AX=AX+DX:AY=AY+DY
2880 IF ABS(BX-CX)<=ABS(EX) AND ABS(BY-C
Y)<=ABS(EY) THEN LINE -((CX+VX%)*VF%+10,
```

```
(CY+VY%)*VF%+10):GOTO 2900
2890 GOTO 2810
2900 RETURN
2910 REM ***************************
2920 REM *        KOPTEKST         *
2930 REM *        --------         *
2940 REM ***************************
2950 REM
2960 OPEN "GRP:" AS 1:PRESET(8,2):COLOR
0:PRINT #1,"MSX-TEKENPROGRAMMA STARK TEX
EL":COLOR 15:CLOSE
2970 RETURN
2980 REM ***************************
2990 REM *     TEKEN DE PIJL       *
3000 REM *     -------------       *
3010 REM ***************************
3020 PUTSPRITE 0,((X%+VX%)*VF%+10,(Y%+VY
%)*VF%+9),14:RETURN
3030 RETURN
3040 REM ***************************
3050 REM * PLAATS FIGUUR IN P%    *
3060 REM * --------------------    *
3070 REM ***************************
3080 REM
3090 IF P%<E% THEN 3120
3100 P%=1:P%(PP%,0)=X%(0):P%(PP%,1)=Y%(0
):P%(PP%,2)=X%(1):P%(PP%,3)=Y%(1):P%(PP%
,4)=X%(2):P%(PP%,5)=Y%(2):P%(PP%,6)=I%:G
OSUB 2320'TEKEN 1 FIGUUR
3110 PP%=PP%+1:PUTSPRITE PN%+1+3*(PN%=3)
,(-1,-1):PUTSPRITE PN%+2+3*(PN%>1),(-1,-
1):GOSUB 2910'KOPTEKST
3120 RETURN
3130 REM ***************************
3140 REM * SPECIFICEREN TEK.NAAM *
3150 REM * ---------------------- *
```

```
3160 REM ***************************
3170 REM
3180 FOR I=1 TO 100:K$=INKEY$:IF K$="" T
HEN NEXT I:F$="":RETURN
3190 IF (K$<"A" OR K$>"Z") AND (K$<"0" O
R K$>"9") THEN RETURN
3200 F$="QD:TEK"+K$+".DAT":RETURN
3210 REM ***************************
3220 REM *       FOUTAFVANG        *
3230 REM *       ----------        *
3240 REM ***************************
3250 REM
3260 F%=-1:PLAY "T64L4M7000S1C","T64L4S1
R32E","T64S1L4R16G"
3270 RESUME NEXT
```

Aantekeningen

Aantekeningen

Aantekeningen

Aantekeningen

Aantekeningen

ENKELE MSX-uitgaven

serie: UW MSX-COMPUTER DE BAAS

MSX BASIC HANDBOEK voor iedereen, door A.C.J. Groeneveld
Een kompleet nederlandstalig handboek voor iedere MSX computergebruiker
ISBN 90 6398 100 7

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

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

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

SOFTWARE PLUS IN MSX

INTROTAPE MSX door A.C.J. Groeneveld.
Heeft u nog maar net een MSX computer gekocht en wilt u graag weten wat de computer kan en hoe u hem kunt leren programmeren? Deze cassette introduceert MSX op een uiterst vriendelijke en onderwijzende manier. U krijgt instructies hoe u de computer aan moet sluiten en de tape laden. Daarna volgt een demonstratie van de mogelijkheden in MSX, zoals het tekenen van sprites en het werken met de driestemmige toongenerator. Het geheel wordt afgesloten met twee 'les' gedeeltes. In anderhalf à drie uur weet u wat de MSX computer is, wat hij kan, en heeft u haast ongemerkt al wat regels geprogrammeerd.
ISBN 90 6398 148 1

MSX SCRIPT door Ton Weijters
Een menugestuurde nederlandstalige tekstverwerker. Het programma is geschikt om efficiënt grotere of kleinere teksten te bewerken. Paginaindeling (regellengte, paginalengte, marge, inspringen, centreren, enz.) wordt door het programma verzorgd. Dit geldt ook voor de paginatelling, toptitel en het eventueel uitvullen van de regels. Ook corrigeren, zoeken, string-substitutie, blokken tekst verplaatsen, copiëren of verwijderen, onderstrepen en vet zetten, is mogelijk met dit programma.
ISBN 90 6398 189 9

MSX DRAWS door A.C.J. Groeneveld
Een tekenprogramma in MSX basic, waarmee u al binnen 10 minuten uw eerste tekening kunt maken. Draws werkt erg vriendelijk en maakt gebruik van alle grafische mogelijkheden van de MSX computer. U kunt met Draws zowel technisch als creatieve tekeningen maken. Het programma heeft een effectief bereik van ruim 30.000 bij 30.000 puntjes met mogelijkheden als lijnen, cirkels, krommen, inkleuren, vergroten, verkleinen, verschuiven, verdraaien en andere tekeningen invoegen
ISBN 90 6398 754 4

MSX QUICK DISK

Met het MSX-basic, dat door de systeemsoftware-expert MICROSOFT is ontwikkeld, is er eindelijk een einde gekomen aan het probleem van de uitwisselbaarheid van software op microcomputers. Tussen de nu meer dan DRIE HONDERD verschillende BASIC-dialecten die het levenslicht al hebben gezien, is er eindelijk een standaard opgestaan. En dat werd tijd!
De letters MSX staan voor MicroSoft eXtended basic. Met zijn ongeveer 150 verschillende sleutelwoorden is MSX een basic zonder weerga en wordt het reeds door vele onafhakelijke microcomputerfabrikanten als standaard taal gevoerd.

De Quick-disk vormt binnen MSX een alternatief tussen de cassette-recorder en de floppy disk. Een disk-drive blijft voor veel mensen een financieel niet haalbare droom, terwijl de cassetterecorder als traag en onbetrouwbaar opslagmedium al snel begint te vervelen.
De Quick-disk werd speciaal voor MSX-computers ontworpen als een snel sequentieel opslagmedium en is door zijn lage prijs al vrij sterk ingeburgerd bij de vele MSX-hobbyïsten.
Alhoewel de Quick-disk geen gebruik maakt van standaard MSX sleutelwoorden, is zijn succes zo groot dat men spoedig kan verwachten dat de Quick-disk als standaard low-cost MSX-medium zal worden geaccepteerd.
Het MSX Quick-disk handboek vormt een aanvulling op het MSX BASIC handboek voor iedereen (ISBN 90 6398 100 7) en behandelt specifiek de Quick-disk kommando's. Voorafgaand aan deze behandeling wordt eerst het principe van de Quick-disk uitvoerig besproken.
De verschillende kommando's zijn voorzien van heldere, goed bruikbare voorbeelden die de laatste onduidelijkheden wegnemen.

ISBN 90 6398 254 2