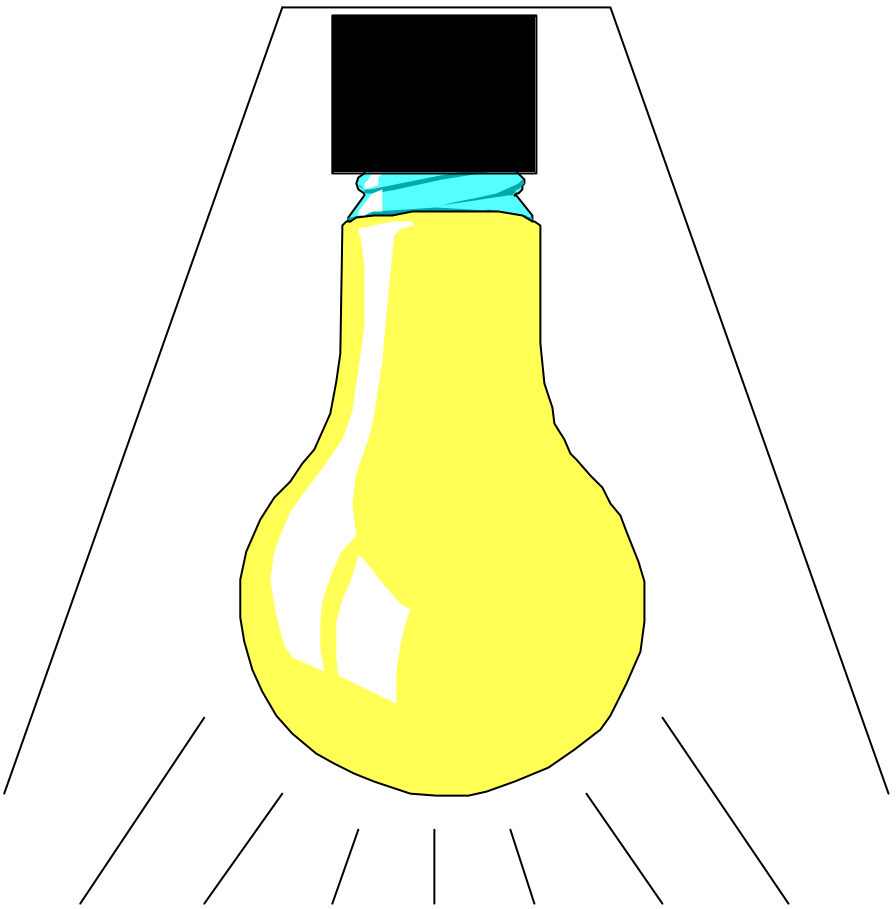
Kijk op (in) MS-DOS



In de kernel zijn ondermeer de volgende functionele componenten te onderscheiden:
File & record management,
memory management,
program & process management,
real-time clock management.

07-11-00

© L H Meuldijk

INHOUDSOPGAVE

1. Inleiding 1
2. Geschiedenis 1
3. Componenten 1
   3.1. BIOS en I/O.SYS 1
   3.2. DOS-kern 1
   3.3. Shell 1
   3.4. Bestands-systeem 1
4. 'Bootstrap' en geheugenindeling 1
5. Architectuur 8086(8088) 1
   5.1. Adressering 1
   5.2. Interrupts 1
6. Executeerbare bestanden 1
   6.1. COM-bestanden 1
   6.2. EXE-bestanden 1
   6.3. Program-Segment-Prefix 1
   6.4. Batch-bestanden 1
7. System calls 1
8. Device-drivers 1
   8.1. Soorten randapparaten 1
   8.2. Standaard device-drivers 1
   8.3. Installeerbare device-drivers 1
   8.4. Opbouw van een device-driver 1
   8.5. Werking van een device-driver 1
9. Bestands-systeem 1
   9.1. Diskette(5.25) 1
   9.2. Harde schijf 1
   9.3. Formatteren 1
   9.4. File Allocation Table 1
   9.5. Directorie 1
10. Literatuur 1

Kijk op (in) MS-DOS blz 1

1. Inleiding

Het Microsoft-Disk-Operating-System MS-DOS is een single-user operating-system voor de IBM-pc en compatibelen.
Grof bezien bestaat de programmatuur uit drie onderdelen:

- een I/O component,
- een component bestands-beheer en
- een commando-interpretator.

Single-user betekent in dit geval ook: geen multiprogrammering dus geen concurrente processen (behalve spool-mogelijkheden).

Dit betekent:

- "geen" concurrente processen,
- "geen" job-scheduling,
- "geen" cpu-scheduling,
- "geen" geheugenbeheer,
- "geen" virtueel geheugen,
- "geen" disk-scheduling,
- "geen" deadlocks,
- "geen" protectie.

2. Geschiedenis

De vroege microcomputers waren voorzien van 8-bits processoren. In de technisch-wetenschappelijke sfeer werd de Apple II met de 6502 microprocessor en het AppleDos operating-system een standaard, in de administratieve sfeer de op de Z80 microprocessor gebaseerde computers met het CPM (Control Program for Microcomputers) operating-system.
De volgende generatie microcomputers zijn gebouwd rond 16-bits processoren, vnl de Motorola 68000 en de Intel 8086(8088).
MS-DOS is ontstaan uit behoefte naar een operating-system voor de IBM-PC waarin de Intel 8088 gebouwd is.
MS-DOS is afgeleid van CPM en is daarna verder ontwikkeld waarbij ook Unix-achtige constructies gebruikt zijn.

1980	Seattle Computer Products bouwt QDOS (86-DOS) met als voorbeeld CPM.	
1981	Microsoft koopt 86-DOS en bouwt dit uit tot	
	PC-DOS 1.0	voor de IBM-PC en
	MS-DOS 1.0	voor compatibelen (grootte 16k)
1982	MS-DOS 1.1:	ondersteuning 2-zijdige diskettes
1983	MS-DOS 2.0:	ondersteuning hiërarchisch bestandssysteem
		ondersteuning harde schijf
		Unix-achtige faciliteiten
		eigen device-drivers mogelijk (grootte 24k)
	MS-DOS 2.11:	internationalisering,
		wijdverspreide versie
1984	MS-DOS 3.0:	ondersteuning 80286 (grootte >36k)
	MS-DOS 3.1:	ondersteuning netwerken.
1986	MS-DOS 3.2:	o.a. ondersteuning 3.5 inch diskettes
1987	MS-DOS 3.3:	standaard-versie
1988	MS-DOS 4:	o.a. ondersteuning grote harde schijven
1991	MS-DOS 5:	beter gebruik van xms, gebruiksvriendelijker

Kijk op (in) MS-DOS blz 2

3. Componenten

MS-DOS is opgebouwd uit drie componenten:

I/O-component,
DOS-kern ('kernel'),
Commando-interpretator (een 'shell').

3.1. BIOS en I/O.SYS

De I/O-component bestaat uit twee delen: een verzameling routines in ROM die de hardware manipuleren (de BIOS). Men zou deze bijna tot de hardware kunnen rekenen omdat ze voor MS-DOS het interface naar de hardware verzorgen. Deze zgn BIOS-routines zijn apparatuur-afhankelijk zodat deze component altijd geleverd wordt door de hardware-fabrikant.
Het andere deel (de IO.SYS) bevat I/O-programmatuur van MS-DOS zoals 'device-drivers'. Dit deel is een echt onderdeel van MS-DOS.
De standaard aanwezige (resident) drivers zijn:

console (beeldscherm en toetsenbord), genaamd CON:,
1e asynchrone communicatiepoort, genaamd AUX: of COM1:,
2e asynchrone communicatiepoort, genaamd COM2:,
1e parallelle printer, genaamd PRN: of LPT1:,
2e parallelle printer, genaamd LPT2:,
dummy-device, genaamd NUL:,
diskette-driver,
harde schijf-driver.

Ook is het mogelijk eigen drivers te installeren.

De IO.SYS staat op de systeem-schijf onder die naam met de aantekening dat het bestand de attributen "system" en "hidden" bezit. De consequentie van het attribuut "hidden" is dat de bestandsnaam niet in een directorie-lijst te zien is.

3.2. DOS-kern

De DOS-kern is, zoals de naam al zegt, de kern ('kernel') van het operating-system.
In de kern zijn ondermeer de volgende functionele componenten te onderscheiden:

bestand- en record-beheer ('file & record management'),
geheugen-beheer ('memory management'),
programma- en proces-beheer ('program & process management'),
klok-beheer ('real-time clock management').

Deze functies zijn op programmeurs-niveau te gebruiken via het 'system-call'-mechanisme en indirect via de systeem-programma's.

De DOS-kern staat op de systeem-schijf onder de naam MSDOS.SYS en heeft, net zoals IO.SYS, de attributen "system" en "hidden".

3.3. Shell

De commando-interpretator is de derde hoofd-component van MS-DOS. Dit is niets meer dan een speciaal applicatie-programma dat het 'user interface' verzorgt van het operating-system. Als er geen andere applicaties te verwerken zijn, wordt dit programma altijd gestart en de beschikbaarheid is te herkennen aan de aanduiding ('prompt') "A>" (of iets van die geest) op het beeldscherm.

De commando-interpretator bestaat uit drie onderdelen:
een initialisatie-deel,
een resident (dus altijd aanwezig) deel,
een transiënt (dus vluchtig) deel.

De interpretator verwerkt de opdrachten van de gebruiker, de commando's. Er zijn drie categorieën commando's:

-- interne, deze resideren in het vluchtige deel en zijn altijd beschikbaar,
-- externe, deze moeten als executeerbare bestanden aanwezig zijn in de werk-directorie of het commando moet voorzien zijn van een pad-naam (zie hierna),
-- 'batch'-bestanden, dit zijn door de gebruiker gemaakte bestanden waarin een verzameling commando's, beschikbaarheid: zie externe.

3.4. Bestands-systeem

Elk bestand heeft een naam.
Elk bestand wordt via de naam geregistreerd in een directorie.
Elke directorie is opgenomen in een directorie-boom.
Elke schijf bevat een (eigen) directorie-boom.

VB: (per schijf)

```
                          ==============
                          |  \         |    de hoofd('root')-directorie
                          ==============
                                |
      -------------------------------------------
      |           |                 |           |
  -----------    ----------    ==========    ----------
  | BESTAND1| ...|BESTAND2| ...| \DIR1  | ...| BESTAND3| ...
  -----------    ----------    ==========    ----------
                                    |
                       --------------------
                       |                  |
                   ===========        -----------
                   | \DIR2   | .....  | BESTAND4| ....
                   ===========        -----------
                       |
            ----------------------
            |                    |
      ------------         ------------
      | BESTAND5 |  .....  | BESTAND6 | .....
      ------------         ------------
```

De werk-directorie is de directorie die als zodanig opgegeven is.
Als een te gebruiken bestand in de werk-directorie opgenomen is, kan volstaan worden met het opgeven van de betrokken bestandsnaam.
VB: als \DIR1\DIR2 de werk-directorie is, dan is de specificatie BESTAND6 voldoende om dit bestand te kunnen gebruiken.
Als een te gebruiken bestand niet in de werk-directorie opgenomen is, dan moet opgegeven worden: zoekpad\bestandsnaam.

VB: stel de werk-directorie is \DIR1 en het te gebruiken bestand is BESTAND6, dan moet opgegeven worden: .\DIR2\BESTAND6

De bestandsnaam is belangrijk voor het gedrag van het systeem.

Kijk op (in) MS-DOS blz 4

Een bestandsnaam is als volgt opgebouwd: naam[.ext] waarin
naam minimaal 1 en maximaal 8 alfanumerieke tekens,
ext maximaal 3 alfanumerieke tekens.
ext is de zgn extensie die in bepaalde gevallen een bijzondere betekenis voor het systeem heeft.
Executeerbare bestanden moeten de extensie COM of EXE hebben (zie later), een commando-bestand de extensie BAT.
Als bij een commando de extensie niet opgegeven wordt, dan is de zoekvolgorde 1e COM, 2e EXE en 3e BAT.

4. 'Bootstrap' en geheugenindeling

Het opstarten van het systeem is een zgn bootstrap-proces, dwz stap voor stap worden de delen van het systeem geladen en geïnitialiseerd.
Dit alles wordt in gang gezet door programmatuur in ROM in de PC.
Grofweg is de gang van zaken als volgt:

1	ROM-boot:	laadt de disk-boot dwz leest kant 0, spoor 0, sector 1 van de aanwezige schijf;
2	Disk-boot:	laadt IO.SYS-boot;
3	IO.SYS-boot:	laadt bestanden IO.SYS en MSDOS.SYS-boot;
4	MSDOS.SYS-boot:	laadt bestand MSDOS.SYS, initialiseert tabellen, drivers, interrupt-vectors, disk-buffer-cache, file-control-blocks, standard input, standard output, standard error, standard list, laadt COMMAND.COM;
5	start de commando-interpretator en prompt.	

De geheugenindeling is dan als volgt:

```
100000h >   ---------------------------------------
            BIOS, o.a. ROM-bootstrap
0FFFF0h >   ---------------------------------------
            ///////////////////////////////////////
            ///////////////////////////////////////
RAM top >   ---------------------------------------
            transiënt COMMAND.COM
            ---------------------------------------
      T     transiënt
      P     program
      A     area
            ---------------------------------------
            resident COMMAND.COM
            ---------------------------------------
            installable device drivers
            ---------------------------------------
            file-control-blocks
            ---------------------------------------
            disk-buffer-cache
            ---------------------------------------
            DOS kernel
             MSDOS.SYS
            ---------------------------------------
            IO.SYS
00400h >    ---------------------------------------
            interrupt-vectors
00000h >    ---------------------------------------
```

5. Architectuur 8086(8088)

Alvorens verder in te gaan op MS-DOS komen nu eerst enige bijzonderheden met betrekking tot de architectuur van de betrokken microprocessor aan de orde.

5.1. Adressering

Instructies kunnen adressen bevatten van 16 bits. Dit zou betekenen dat het direct adresseerbare geheugen 64k is.
Echter de microprocessor bezit ook extra (segment)registers van 16 bits met behulp waarvan een adres opgebouwd wordt van 20 bits zodat in werkelijkheid 1Mb te benaderen is.
(NB: MS-DOS beperkt zelf het adresseerbare geheugen tot 640k)

Adres berekening gaat nu als volgt:

```
adres uit instructie:                  bbbbbbbbbbbbbbbb    16 bits

adres uit segment-register:        bbbbbbbbbbbbbbbb0000    16+4 bits
                                   ---------------------- +
effectief adres:                   bbbbbbbbbbbbbbbbbbbb    20 bits
```

De notatie voor deze optelling is "segment-register:adres".

De betrokken microprocessor bezit vier segment-registers:

CS waarin het adres van het begin van het code-segment,
DS waarin het adres van het begin van het data-segment,
SS waarin het adres van het begin van het stack-segment,
ES waarin het adres van het begin van het extra-segment.

Van belang zijn verder nog:

IP waarin de program counter (relatief tov CS),
SP waarin de stack pointer (relatief tov SS).

De gang van zaken is nu als volgt:

van een instructie is het effectief adres	CS:IP,
van een "gewoon" data-element is het effectief adres	DS:adres,
een stack-element van effectief adres	SS:SP,
een "extra" data-element van effectief adres	ES:adres.

5.2. Interrupts

De architectuur van de 8086(8088) is gebaseerd op het feit dat de eerste 1024 bytes van het werkgeheugen gereserveerd zijn tbv interrupt-vectoren. Elke interrupt-vector omvat vier bytes zodat er 256 interrupt-vectoren zijn.
Een interrupt-vector bevat het adres (segment-adres en offset) van de bijbehorende interrupt-handler.
De interrupts zijn geordend naar aflopende prioriteit.

Interrupts worden ingedeeld in drie categorieën:
- interne hardware interrupts,
- externe hardware interrupts,
- software interrupts (system calls).

Interne hardware interrupts zijn het gevolg van "harde"-fouten gedetecteerd tijdens de executie van een programma, zoals delen door nul e.d..
Externe hardware interrupts worden veroorzaakt door ingrepen van perifere apparaten. Deze interrupts zijn "hardbedraad" door de fabrikant van de PC.

Kijk op (in) MS-DOS blz 6

Software interrupts zijn het gevolg van de executie van de (machine)instructie INT.

Interrupts 00h t/m 1Fh worden gebruikt voor interne hardware interrupts en BIOS-aanroepen, 20h t/m 3Fh worden gebruikt door MS-DOS, de overige worden gebruikt voor externe hardware interrupts of door applicatieprogramma's.

6. Executeerbare bestanden

Zoals eerder gesteld, moeten executeerbare bestanden een bestandsnaam hebben met de extensie COM, EXE of BAT. De BAT-bestanden zijn commando-bestanden en komen daarom hier niet aan de orde.
COM- en EXE-bestanden zijn binaire bestanden maar vertonen belangrijke verschillen die het gevolg zijn van het adresseer-mechanisme van de betrokken microprocessor.

Het belangrijkste verschil is dat COM-bestanden geacht worden een programma te bevatten dat tijdens executie volledig in één segment past. Een programma in een EXE-bestand mag daarentegen groter zijn en dus meerdere segmenten beslaan.

Dit heeft consequenties voor de vorm waarin deze bestanden op schijf opgeslagen worden en voor de manier waarop deze bestanden geladen worden:

6.1. COM-bestanden

Op schijf staan geen speciale gegevens met betrekking tot de inhoud, de extensie COM is voldoende. Bij het laden van een COM-bestand in het werkgeheugen vinden achtereenvolgens de volgende acties plaats: een segment wordt gealloceerd (op een adres mod 16 bytes) en de 16 high-order bits van dat adres worden geplaatst in alle segment-registers. In de eerste 256 bytes van het segment wordt de zgn program-segment-prefix (zie later) opgebouwd en het COM-bestand wordt geladen te beginnen op adres CS:00100h, dit wordt beschouwd als het adres waar executie dient te beginnen. In IP wordt geladen 00100h en in SP 0FFFEh. Executie kan nu beginnen op adres CS:IP.

6.2. EXE-bestanden

Deze bestanden zijn voorzien van een zgn header waarin gegevens staan met betrekking tot het bestand zoals segment-gegevens, startpunt ('entrypoint'), relocatie-tabel, ed. Deze gegevens zijn nodig om het laadproces te laten weten wat er moet gebeuren.
Eerst worden nu de benodigde segmenten gealloceerd. De program-segment-prefix wordt opgebouwd in de eerste 256 bytes van het data-segment en de header wordt geladen te beginnen op adres DS:00100h. DS en ES bevatten hetzelfde adres. De rest van het bestand wordt dan relocaterend geladen en CS, IP en SS worden gevuld met adressen die bepaald worden op grond van de gegevens in de header. Het adres van het laatste woord van het stack-segment (relatief tov het begin van het stack-segment) wordt in SP geplaatst. Executie kan nu beginnen op adres CS:IP.
Let op: DS en ES zijn nog niet gevuld met het adres van het datasegment van het programma.

Een EXE-bestand ontstaat na linken van objectprogramma's (gecompileerde programma's).
Om een COM-bestand te maken moet eerst een EXE-bestand gemaakt worden waarvan dan met behulp van de utility EXE2BIN een COM-bestand gemaakt kan worden.

Kijk op (in) MS-DOS blz 7

6.3. Program-Segment-Prefix

De program-segment-prefix (PSP) is een onderdeel van het "core-image" van een executeerbaar programma en bevat "runtime"-gege-vens voor het proces. Dus is de PSP een (belangrijk) onderdeel van proces-context.
De inhoud van de PSP staat hieronder beknopt beschreven:

offset(hex)	inhoud, betekenis
00-01	CD20h (= int 20h): link naar de proces-termination-handler van MS-DOS, ruimt alles op
02-03	adres van het 1e niet-gealloceerde segment van TPA (normaliter: top of memory)
04	gereserveerd
05-	long call to MS-DOS function dispatcher
0A-0D	oude inhoud v d termination handler interrupt vector
0E-11	oude inhoud v d ctrl-break handler interrupt vector
12-15	oude inhoud v d critical-error handler int. vector
16-2B	gereserveerd
2C-2D	segment-adres van het "environment"-blok: dit blok bevat een aantal ASCII-strings die gebruikt worden door MS-DOS tijdens de programma-executie. o.a. PATH specificatie, PROMPT specificatie, plaats van COMMAND.COM op schijf...
2E-5B	gereserveerd
5C-67	file control block #1: tbv compatibiliteit met CP/M
6C-77	file control block #2: tbv compatibiliteit met CP/M
80-FF	command tail: de rest van de commandoregel die dit programma opriep. disk transfer area: tbv compatibiliteit met CP/M

Vanaf offset 5Ch mogen programma's geen veranderingen in de PSP aanbrengen, dit deel wordt door MS-DOS gebruikt.

6.4. Batch-bestanden

p.m.

7. System calls

Zoals eerder vermeld worden de interrupt-vectoren 20h t/m 3Fh gebruikt door MS-DOS.
De belangrijkste hiervan is interrupt-vector 21h. De bijbehorende interrupt-handler is de zgn MS-DOS function dispatcher. Alle functies die het operating-system beschikbaar heeft zijn toegankelijk via deze interrupt.
Om een systeemfunctie te activeren dient de volgende system call uitgevoerd te worden:

- zet de nodige parameters klaar op de vereiste manier,
- plaats het nummer van de system call in register AH,
- voer de instructie INT 21h uit.

Als programmatuur slechts op deze manier bv I/O enz doet, is de programmatuur hardware onafhankelijk.

Enige voorbeelden van MS-DOS functies zijn hieronder vermeld:

Nr	Functie
00h	program terminate
01h	read keyboard and echo
02h	display output
05h	printer output
2Ah	get date
2Ch	get time
39h	create subdirectory
3Ch	create a file
3Dh	open a file
3Fh	read from a file or device
40h	write to a file or device
3Eh	close a file
41h	delete a file
56h	rename a file
44h	I/O control for devices
48h	allocate memory
49h	free memory
4Bh	load and execute a program
4Ch	terminate a process

Een aantal functies tussen 00h en 30h zijn alleen zinvol in verband met compatibiliteit met CP/M programmatuur. Deze zijn overbodig onder MS-DOS 2.0 en hoger. Ter vervanging van deze zijn nieuwe functies ingebouwd met nummers boven 2Fh. Daarom zijn de functies boven 2Fh alleen te gebruiken onder MS-DOS 2.0 en hoger.

8. Device-drivers

Het hart van een microcomputer bestaat uit twee elementen: de centrale verwerkingseenheid en het werkgeheugen. Alle andere hardware-componenten (schijfeenheden, toetsenbord, beeldscherm, printers, modems, enz.) zijn randapparaten (peripheral devices).
Communicatie tussen de randapparatuur en de centrale machine wordt geregeld door het operating-system. Daartoe bevat MS-DOS voor elk randapparaat een software-component, een zgn device-driver. Deze zijn of standaard opgenomen in MS-DOS of moeten apart geïnstalleerd worden.

8.1. Soorten randapparaten

MS-DOS kent twee soorten randapparaten: apparaten die tekengewijs werken ('character device', vb: toetsenbord, beeldscherm, ed.), en apparaten die gegevens blokgewijs verwerken ('block device', vb: schijfeenheden).
Een karakterapparaat heeft een naam en de betrokken device-driver wordt aangeroepen mbv die naam. Een blokapparaat heeft geen naam maar wordt gerefereerd door een letter te gebruiken die de (schijf)eenheid aanwijst (A:, B:, C:, enz.).
Voorbeelden hiervan zijn in de volgende paragraaf op rij gezet.

8.2. Standaard device-drivers

Ingebouwd in MS-DOS zijn device-drivers voor de console, seriële communicatie poorten, printers en disk-drivers.
Hieronder staan deze standaard drivers opgesomd:

Randapparaat	soort	naam
console (toetsenbord, beeldscherm)	character	CON:
1e asynchrone communicatiepoort	character	COM1: of AUX:
2e asynchrone communicatiepoort	character	COM2:
1e parallelle printer	character	LPT1: of PRN:
2e parallelle printer	character	LPT2:
dummy-device	character	NUL:
diskette-eenheid	block	--
harde schijf	block	--

8.3. Installeerbare device-drivers

Voor andere dan standaard randapparaten is het mogelijk device-drivers toe te voegen aan MS-DOS. Voorwaarde is dat de betrokken software voldoet aan de MS-DOS conventies (zie later).
Bij het opstarten van het systeem wordt altijd gezocht naar een bestand met naam CONFIG.SYS. In dit bestand worden zaken aangegeven die specifiek zijn voor de betrokken configuratie. Tijdens bootstrap wordt het systeem dan aangepast aan de plaatselijke situatie.
Als er extra device-drivers geïnstalleerd moeten worden, kan dat opgegeven worden in CONFIG.SYS dmv de DEVICE= declaratie.
Vb: in CONFIG.SYS staat DEVICE = ANSI.SYS, waarmee aangegeven wordt dat als toetsenbord-driver ANSI.SYS geïnstalleerd moet worden.

8.4. Opbouw van een device-driver

MS-DOS device-drivers zijn programma's die na vertaling en linking in .COM formaat beschikbaar moeten zijn.
Een device-driver omvat drie delen: een 'device-header', een 'strategy-routine' en een 'interrupt-routine'.

De device-header bevat diverse gegevens:
-- link naar volgende driver
(NB: drivers zijn geordend in een keten),
-- attributen van de driver,
-- adres van de stategy-routine,
-- adres van de interrupt-routine,
-- device-naam (character device)
of aantal gelijksoortige randapparaten die door deze driver bediend worden (block device).

De strategy-routine wordt aangeroepen als een dienst van een device-driver gevraagd wordt (hij functioneert dus als een soort start-I/O). De enige actie die deze routine uitvoert, is het bewaren van het adres van de request-header (zie hierna) voor de interrupt-routine.

De interrupt-routine voert de eigenlijke opdracht uit.

De request-header is een control-block dat klaar gemaakt wordt door diegene die een dienst van de betrokken device-driver vraagt. Hierin wordt oa gespecificeerd de opdracht-code, het adres van de gegevens, status-gegevens na uitvoering van de opdracht ed.

8.5. Werking van een device-driver

De scheiding in een strategy-routine en een interrupt-routine is in deze opzet van MS-DOS zinloos. Dit wordt pas zinvol als er sprake is van multiprogrammering. De strategy-routine voert dan de start-I/O uit, dus initieert de I/O-actie. De interrupt-routine wordt dan een echte interrupt routine die geactiveerd wordt als de I/O-opdracht voltooid is.

De verwerking is thans een sequentieel, synchroon verlopend proces. De verwerking van een I/O-opdracht gaat in grote lijnen als volgt:

-- een applicatie-programma maakt de nodige gegevens klaar en doet een syscall 21h,
-- MS-DOS zoekt de bijbehorende driver,
-- MS-DOS maakt een request-header nav de door de gebruiker aangeboden gegevens,
-- MS-DOS roept de strategy-routine aan,
-- de strategy-routine bewaart het adres van de request-header en springt terug naar MS-DOS,
-- MS-DOS roept de interrupt-routine aan,
-- de interrupt-routine redt de inhoud van de registers, haalt het adres van de request-header, bepaalt om welke opdracht het gaat en springt naar de subroutine die de betrokken opdracht afhandelt, als het een transport betreft, wordt dit gestart en daarna wordt gewacht tot het transport voltooid is,
-- als de opdracht gereed is, zet de interrupt-routine de vereiste gegevens (bv: status) in de request-header, herstelt de inhoud van de registers en springt terug naar MS-DOS,
-- MS-DOS bepaalt de return-code nav de driver-status en springt terug naar de aanvrager.

9. Bestands-systeem

De meest gangbare hardware-componenten van het MS-DOS bestands-systeem zijn:
-- harde schijf,
-- 3.5 inch diskette,
-- 5.25 inch diskette,
-- RAM-schijf.

9.1. Diskette(5.25)

Vanaf versie 2.11 is de DSDD (dubble sided, dubble density) diskette standaard.
De beide zijden van de diskette worden gebruikt op de volgende manier:
Elke schijf heeft 2 kanten, kant 0 en kant 1.
Elke kant bevat 40 sporen, 0 (buitenste) t/m 39 (binnenste)
Elk spoor bevat 9 sectoren genummerd 1 t/m 9.
Elke sector is 512 bytes groot.
De totale capaciteit van een DSDD-diskette is dus 368640 bytes (2 * 40 * 9 * 512 = 368640).

9.2. Harde schijf

Harde schijven bestaan uit een of meer platen met elk twee kanten. De oppervlakte van elke kant is verdeeld in sporen en elk spoor is verdeeld in sectoren van 512 bytes groot. Boven elkaar liggende sporen zijn gegroepeerd in cylinders.
Voorbeeld: een 10M schijf heeft 17 sectoren per spoor, 4 sporen per cylinder en 306 cylinders. De totale capaciteit van deze 10M harde schijf is dus (306 * 4 * 17 * 512 =) 10653696 bytes.
Hiervan reserveert MS-DOS een volledige cylinder en vier sectoren van de overige cylinders voor eigen gebruik zodat netto voor de gebruiker ter beschikking zijn 9994240 bytes.

9.3. Formatteren

Voordat een schijf gebruikt kan worden, moet deze geformatteerd worden, dwz de sporen en sectoren moeten klaar voor gebruik gemaakt worden. Hiervoor is beschikbaar het systeemprogramma FORMAT. Tevens wordt hierdoor het bestands-systeem geïnitialiseerd:

- Een kopie van het MS-DOS boot record wordt geplaatst in sector 1 van spoor 0 van kant 0 (dus in de eerste logische sector).
- De File Allocation Table (FAT) wordt gemaakt. (zie later)
- De File Directory (Fdir) wordt gemaakt. (zie later)

De indeling van een 5.25 DSDD-diskette is daarna als volgt:

boot record:	kant 0 / spoor 0 / sector 1,
FAT:	kant 0 / spoor 0 / sector 2 en 3,
FAT (kopie):	kant 0 / spoor 0 / sector 4 en 5,
Fdir:	kant 0 / spoor 0 / sector 6 t/m 9,
	kant 1 / spoor 0 / sector 1 t/m 3.

De wijze van adressering is zoals gebruikelijk:
cylindernr(0 t/m 39), spoornr(0,1), sectornr(1 t/m 9).

9.4. File Allocation Table

De eenheid waarin schijfruimte gealloceerd wordt, is het cluster. Een cluster is een aantal sectoren. De clustergrootte is afhankelijk van het betrokken medium. Zo bestaat een cluster van een 5.25 DSDD diskette uit twee opeenvolgende sectoren.
De cluster indeling van de 5.25 DSDD diskette staat hieronder in schema:

	spoor 0						spoor 1					
sector	1	2	3	4	5	6	7	8	9	1	2	3	4	5	6	7	8	9
cyl 0	bt	f1	f1	f2	f2	d	d	d	d	d	d	d	c-02	c-03	c-04
cyl 1	c-05	c-06	c-07	c-08	c-09	c-10	c-11	c-12	c-13
cyl 2	c-14	c-15	c-16	c-17	c-18	c-19	c-20	c-21	c-22
.									
.									
cyl 39	c-347	c-348	c-349	c-350	c-351	c-352	c-353	c-354	c-355

waarin bt = boot record, f1 = FAT, f2 = FAT(kopie), c_XXX = clusternummer.

De file allocation table (FAT) registreert het cluster gebruik in een "block (cluster) map".
Van het eerste aan een bestand gealloceerd cluster is het nummer vastgelegd in de betrokken directorie-ingang (zie later). Dit nummer verwijst naar de betrokken ingang in de FAT. In deze ingang staat het nummer van het volgende gealloceerde cluster of een code die aangeeft dat dit het laatste cluster van het bestand is, enz.
Dit betekent dat het aantal ingangen in de FAT tenminste 355 moet zijn en dat een FAT-ingang het getal 355 moet kunnen bevatten. Daarom heeft een FAT-ingang een lengte van 1.5 bytes. Gezien de clusternummering van 2 t/m 355 moeten er dus tenminste 354 FAT-ingangen aanwezig zijn. Dus zijn er voor de FAT twee sectoren gereserveerd. Om veiligheidsredenen houdt het systeem een extra kopie van de FAT bij.

Een FAT-ingang bevat dus een clusternummer van het volgende aan het betrokken bestand gealloceerd cluster
of een van de waarden FF8h t/m FFFh als dit het laatste cluster van een bestand is,
of 000h als dit een vrij cluster is,
of een van de waarden FF0h t/m FF6h als dit een speciaal MS-DOS cluster is,
of FF7h als dit een "bad" (niet te gebruiken) cluster is.

Clusternummering begint bij 2 omdat de eerste twee FAT-ingangen speciale gegevens bevatten:
de eerste byte bevat de omschrijving van het betrokken medium, bv FDh voor een 5.25 DSDD-diskette, de rest bevat altijd FFh.

NB: in MS-DOS versie 3 kunnen FAT-ingangen twee bytes groot zijn afhankelijk van het betrokken medium.

9.5. Directorie

De directorie is de "inhoudsopgave" van de betrokken schijf.
Op een DSDD diskette zijn voor deze hoofd-directorie 7 sectoren gereserveerd. De Fdir bevat "file descriptors" van de in de hoofd-directorie geregistreerde bestanden.
De grootte van een file-descriptor is 32 bytes.
Het aantal bestanden dat geregistreerd kan worden in de hoofd-directorie is dus beperkt tot (7 * 512 / 32 =) 112 bestanden.

Een file-descriptor bevat het volgende:

offset	inhoud
000h	Naam van het bestand waarbij de eerste byte een speciale betekenis heeft: 00h als deze ingang nog nooit gebruikt is, E5h als het bestand gewist is, 05h als het eerste teken van de bestandsnaam E5h is,
0008h	bestands-extensie
000Bh	bestands-attribuut bit 0: read only file, bit 1: hidden file, bit 2: system file, bit 3: volume label, bit 4: subdirectory file, bit 5: archive bit (gebruikt door BACKUP),
000Ch	gereserveerd
0016h	tijdstip van creatie of laatste verandering
0018h	datum van creatie of laatste verandering
001Ah	clusternr (nummer van eerste gealloceerde cluster)
001Ch	afmeting van het bestand (low order 2 bytes)
001Eh	afmeting van het bestand (high order 2 bytes)

10. Literatuur

In de handel zijn een massa boeken over MS-DOS. De meeste hiervan behandelen alleen het 'userinterface', dwz de commando's.

De volgende twee boeken gaan wat dieper in op de opbouw en interne werking:

MS-DOS bible, auteur S Simrin, uitgever Sams.
isbn 0-672-22408-9

MS-DOS voor programmeurs, auteur R A King, uitgever Sybex, isbn 3-88745-119-8

Nog dieper gaat:

MS-DOS voor gevorderden, auteur R Duncan, uitgever Kluwer.

Advanced MS-DOS, auteur R Duncan, uitgever Microsoft press.

NB: Het eerste is een vertaling van het tweede.