5.

DOS-maskinen - operativsystemkernen

Indhold:

DOS servicefunktioner

I DOS manualerne findes en beskrivelse af servicefunktionerne ordnet efter deres nummer; her skal funktionskaldene tabelleres efter deres anvendelsesområder.

De allerfleste ikke-reserverede funktionskald er underfunktioner til interrupt 21H, men INT 20H, 25H, 26H og 27H kan også være nyttige. INT 20H og 27H afslutter en programudførelse, men INT 27H på en sådan måde, at programmet forbliver i RAM lageret efter afbrydelsen. INT 25H og 26H hhv. læser og skriver sektorer på diskette ved hjælp af absolut adressering.

Tastatur I/O. INT 21H.

AH=01H	Vent til næste tegn og læs dette med ekko til skærm. Resultat i AL. Hvis Ctrl-Break, udføres INT 23H.
AH=06H	DL=FFH. Hvis tegn er parat, hentes dette ind i AL og ZF=0. Hvis ikke, sættes ZF=1. Ctrl-Break ignoreres.
AH=07H	Som AH=01H, men uden ekko og uden Ctrl-Break detektering.
AH=08H	Som AH=01H, men uden ekko.
AH=0AH	Læs til buffer ved DS:DX+2. DS:DX peger på bufferlængden.
AH=0BH	AH sættes til FFH, hvis tegn er parat; ellers 00H. Hvis Ctrl-Break, udføres INT 23H.
AH=0CH	Tøm tastaturbuffer og udfør en af funktionerne 01H, 06H, 07H, 08H eller 0AH ovf. afhængigt af AL's indhold.
AH=33H	Undersøg eller aktiver Ctrl-Break. Kun DOS 2.

Monitor I/O. INT 21H.

AH=02H	Vis tegnet i DL. Ctrl-Break medfører INT 23H.
AH=06H	DL(<FFH) vises. Ctrl-Break ignoreres.
AH=09H	Vis tekst i buffer DS:DX frem til $ tegn. Iøvr. som AH=02H

Skriver I/O. INT 21H.

AH=05H	Skriv tegnet i DL. Kun 1 skriver understøttes.

Seriel port. INT 21H.

AH=03H	Vent til næste tegn og anbring det i AL. Ingen status information; jfr. ROM BIOS INT 14H i tabel 3.5.3.
AH=04H	Send tegn i AL til seriel port. Iøvr. som AH=03H.

RAM lager. INT 21H. Kun DOS 2.00 og følgende versioner.

AH=48H	Skaf om muligt BX paragraffer. Hvis muligt, returneres 1. paragrafnummer i AX; ellers returneres største frie blok i BX.
AH=4AH	Giv ES segmentet den størrelse, som BX angiver. Iøvr. som for AH=48H.
AH=49H	Aflever lager fra ES og frem til den frie pulje.

Program håndtering. INT 21H.

AH=4BH	Eksekver og/eller load program, hvis filnavn er i DS:DX og hvis parameter blok (se DOS manual) er i ES:BX. Hvis AL=0, konstrueres et PSP og programmet loades og eksekveres; hvis AL=3, udføres kun load. Kun under DOS 2.
AH=4DH	Hent returkode i AL. Bør altid udføres efter 4BH. Kun DOS 2.
AH=26H	Opret nyt programsegment.
AH=00H	Afslut programudførelsen uden returkode. Samme som INT 20H.
AH=4CH	Afslut programudførelsen med returkode. Kun under DOS 2.
AH=31H	Afslut programudførelsen og forbliv resident. Kun under DOS 2. Samme som INT 27H, dog kan returkode sættes (i AL).

DOS SERVICEFUNKTIONER FORTSAT

Filhåndtering. INT 21H. DOS 1.10 og følgende versioner

AH=0EH	Gør disk DL (0=A,1=B osv.) til den aktuelle. AL sættes til antal drev ialt.
AH=19H	Hent nummer på aktuelle drev i AL.
AH=0DH	Initialiser disk og nedlæg alle buffere.
AH=16H	Opret ny fil med FCB i DS:DX. AL=0, hvis det lykkedes; ellers AL=FFH.
AH=0FH	Åbning af fil. FCB fyldes ud. Iøvr. som AH=16H.
AH=10H	Lukning af fil. Iøvr. som ovf.
AH=13H	Nedlæg en lukket fil. Iøvr. som AH=16H.
AH=17H	Giv fil et nyt navn. Iøvr. som AH=16H.
AH=29H	Undersøg filnavn for bl.a. filtype, ?- og *-tegn og opret en FCB på adresse ES:DI for filen uden at åbne denne.
AH=11H	Opsøg 1. fil i en uåbnet FCB med "?"-tegn i filnavn.
AH=12H	Opsøg næste fil i en uåbnet FCB med "?"-tegn i filnavn.
AH=1BH	Hent FAT information for det aktuelle drev.
AH=1CH	Hent FAT information for et udvalgt drev.
AH=1AH	Opret buffer med adresse (DTA) angivet i DS:DX.
AH=14H	Sekventiel læsning af post. DS:DX peger på åben FCB. Unormal afslutning: AL=01-ingen data; AL=02-ikke plads nok i DTA; AL=03-kun del af post fundet.
INT 25	Læs CX sektorer ind i DS:BX. (Absolut adressering).
AH=15H	Sekventiel skrivning af post. Iøvr. som AH=14H; dog betyder AL=01: fuld diskette og AL=02: ikke plads i DTA.
INT 26	Skriv CX sektorer fra DS:BX. (Absolut adressering).
AH=24H	Sæt pegepind til næste ikke-sekventielle post.
AH=21H	Ikke-sekventiel læsning af en post. Iøvr. som AH=14H.
AH=27H	Ikke-sekventiel læsning af en blok af poster. Iøvr. som ovf.
AH=22H	Ikke-sekventiel skrivning af en post. Iøvr. som AH=14H.
AH=28H	Ikke-sekventiel skrivning af blok m. CX poster.Iøvr.som ovf.
AH=23H	Hent antal poster i filen.
AH=28H	CX=0. Giv filen en ny størrelse.
AH=2EH	Slå "læs-efter-skriv verifikation" til(AL=1) eller fra(AL=0)

Fil håndtering. 21H. Kun DOS 2.00 og følgende versioner.

AH=3CH	Opret ny fil som specificeret i DS:DX og med attribut i CX. Fil "handle" returneres i AX.
AH=3DH	Åbning af fil. AL=0,1,2: read only, write only, read/write.
AH=3EH	Lukning af fil med handle i BX.
AH=41H	Nedlæg en lukket fil specificeret i DS:DX.
AH=56H	Giv fil (DS:DX) et nyt navn (ES:DI).
AH=4EH	Opsøg første fil med "?"-tegn i filnavn.
AH=4FH	Opsøg næste fil med "?"-tegn i filnavn.
AH=45H	Giv fil en ny handle.
AH=46H	Lad handle pege på en anden fil.
AH=2FH	Hent buffer adresse (DTA) i ES:BX.
AH=3FH	Læs CX byte ind i buffer ved DS:DX fra fil med handle i BX.
AH=40H	Som AH=3F, men skrivning.
AH=42H	Sæt pegepind til næste datum, der læses eller skrives.
AH=44H	Hent eller sæt status.
AH=39H	Opret underkatalog (MKDIR) med navn ved DS:DX.
AH=3AH	Nedlæg underkatalog (RMDIR) med navn ved DS:DX.
AH=3BH	Skift til nyt katalog (CHDIR) med navn ved DS:DX.
AH=47H	Hent aktuelle katalog ind i DS:SI for drev angivet i DL.
AH=36H	Hent oplysninger om ledig plads på disketten i drev DL.
AH=43H	Hent (AL=0) eller sæt (AL=1) filattributter i CX.
AH=57H	Hent (AL=0) eller sæt (AL=1) dato:klokkeslet i DX:CX.
AH=54H	Hent "læs-efter-skriv verifikation" status i AL.

Specielle funktioner. INT 21H.

AH=2AH	Hent dato i CX:DX.
AH=2CH	Hent klokkeslet i CX:DX.
AH=2BH	Sæt dato i CX:DX.
AH=2DH	Sæt klokkeslet i CX:DX.
AH=35H	Hent interrupt vektor nr. AL i ES:BX. Kun DOS 2.
AH=25H	Sæt interrupt vektor nr. AL til værdien i DS:DX.
AH=38H	Hent nationale oplysninger (mønt, dato format mv.) anbragt på adresse DS:DX. Kun DOS 2.
AH=30H	Hent DOS versionens nummer i AL:AH. Kun DOS 2.

BIOS INTERRUPTS.

BIOS interrupts går fra 00H til 1FH og DOS's fra 20H.
I denne oversigt er = anvendt som inddata til BIOS og
:= anvendt som uddata fra BIOS.

INT nr. Hex.	Dec.	Kommentarer og parametre.
0		Genereres af CPU'en ved division med 0.
1		Single step - anvendes af DEBUG el. SYMDEB.
2 NMI		NON MASKABLE INT. Normalt memory paritetsfejl.
3		Breakpoints. Anvendes af debuggere.
4		Overflow - (anvendes normalt ikke - retur).
5		Print Screen.
6		Reserveret.
7		- - " - - .
8 (IRQ 0)		Timer of Day. Brug 1CH i stedet. 18,2 * pr sek
9 (IRQ 1)		Tastatur interrupt.
A til F er		maskinafhængig - pointer til BIOS rutiner.
A (IRQ 2)		Vertikal retrace for EGA- og VGA- kort.
B (IRQ 3)		Seriel COM2 printer controller.
C (IRQ 4)		- " - COM1 - - - - " - - - - .
D (IRQ 5)		Parallel LPT2 - - - - " - - - - .
E (IRQ 6)		Hard disk controller.
F (IRQ 7)		Parallel LPT1 printer controller.

Skærmen.	AH.	
10H	0	Sæt skærm mode. (Se side 228). AL skal indeholde en af følgende værdier : 0 = 40 * 25 S/H. 1 = 40 * 25 farve (8/16). 2 = 80 * 25 S/H. 3 = 80 * 25 farve (8/16). 4 = 320 * 200 farve (4). 5 = 320 * 200 S/H. 6 = 640 * 200 S/H. 7 = 80 * 25 for monokrom skærm. 0DH = 320 * 200 farve (16). 0EH = 640 * 200 farve (16). 0FH = 640 * 350 S/H. 10H = 640 * 350 farve (16). 11H = 640 * 480 farve (2). 12H = 640 * 480 farve (16). 13H = 320 * 200 farve (256).
10H	1	Sæt cursor mode/type. (Se side 229) Cursorens blink kan ikke standses, da det er sat hardware mæssigt, men kan redefineres da den på en monokrom skærm er 14 linier høj og 8 linier høj på en grafisk skærm. F.eks. CH og CL begge = 0DH vil kun sidste linie blive anvendt. CH = 0DH og CL = 00H vil kun første og sidste linie blive anvendt. CH = Cursor start linie. CL = Cursor stop linie.
10H	2	Sæt cursor til bestemt sted på skærmen. BH = Sidenummer. (Se side 230) DH = Rækkenr. DL = Kolonnenr.

BIOS interrupts.

10H	3	Læs cursorens aktuelle position på skærmen. BH = Sidenr. (Se side 231) CH/CL := Nuværende cursor mode (som funk.1). DH := Rækkenr. DL := Kolonnenr.
10H	4	Læs lyspens aktuelle position på skærmen.
10H	5	Vælg aktive skærmside. AL = 0-3 for mode 2 og 3 se funk. 0.(4 sider) AL = 0-7 - - - - 0 og 1 - - - - - .(8 sider)
10H	6	Aktiv skærmside scrolles op. (Se side 233) AL = 0 så blankes det hele. AL = X så blankes X linier i bunden af siden. BH = Attributen til blankning. CH/CL = Øvre venstre række, kolonne til scroll DH/DL = Nedre højre - - - - - " - - - - - - -
10H	7	Aktiv skærmside scrolles ned. (Se side 233) Som for funktion 6.
10H	8	Læs tegnet/attribut fra aktuel cursorposition. BH = Sidenr. (Se side 234) AL := ASCII tegn. AH := Attributen.
10H	9	Skriver tegn/attribut til - - - - " - - - - -. BH = Sidenr. (Se side 235) BL = Attribute. CX = Antal gange tegnet skal gentages. AL = ASCII tegnet.
10H	10	Skriv tegn til aktuel cursorposition. AL = ASCII tegn. BH = Sidenr. CX = Antal gange tegnet skal gentages.
10H	11	Sæt farve palette (kun i grafik mode). BH = Farve palette nr. (Se side 237) BL = 0 Vælges baggrundsfarven ifølge BH. BL = 1 Vælges palette ifølge BH.
10H	12	Skriv til pixel. (Se side 238) AL = Farve. DX = Rækkenr. CX = Kolonnenr.
10H	13	Læs pixel værdi (som for funktion 12). (Se side 239)
10H	14	Skriv tegn som TTY (teletype). (Se side 240) AL = Tegnet. BL = Forgrundsfarven (kun i grafik mode). BL = Sidenr.
10H	15	Returner nuværende skærmmode. (Se side 241) AH := Antal skærmkolonner (40 el. 80). AL := Nuværende mode. (Se INT 10H funk. 0). BH := Aktiv sidenr.

BIOS interrupts.

RS 232C (Seriel printer port - normalt 2 - COM1 og COM2)

```
14H        0   Initiering af porten.
               AL = Bit 0,1 Ordlængde (01=7 bits,11=8 bits).
                         2  Stopbits. ( 0=1 bit , 1=2 bits).
                       3,4 Paritet.  (00=ingen , 01 = ulige
                                                 10 = lige).
                     5,6,7 Baud Rate  :
                                        000 = 110
                                        001 = 150
                                        010 = 300
                                        011 = 600
                                        100 = 1200
                                        101 = 2400
                                        110 = 4800
                                        111 = 9600

14H        1   Send et tegn.
               AL  = Tegnet.
               AH := bit 7 = 1 så fejl, ved bit 7 = 0 så ok.
                     bit 6...0 se under funktion 3.

14H        2   Modtag et tegn.
               AL := Tegnet.
               AH := 0 så ok, ellers fejl se funktion 3.

14H        3   Hent portens status.
               AH := Bit 7 = Time out.
                         6 = Shift register tom.
                         5 = Holding register tom.
                         4 = Break detected.
                         3 = Framing fejl.
                         2 = Paritets fejl.
                         1 = Overflow fejl.
                         0 = Data Ready.

               AL := Bit 7 = Received Line Signal Detect.
                         6 = Ring indicator.
                         5 = Data Set Ready.
                         4 = Clear to Send.
                         3 = Delta Received Line Signal Det.
                         2 = Trailing Edge Ring Detector.
                         1 = Delta Data Set Ready.
                         0 = Delta Clear to Send.
```

Tastatur. AH

```
16H        0   Læs næste ASCII tegn til AL og Scan kode i AH.
               (Se side 261)
16H        1   Afgør om der er tegn i bufferen. (Se side 262)
               Z bit = 1 hvis bufferen er tom.
               Z bit = 0 AH = Scan kode.   (Anvend funk. 0 da
                         AL = ASCII tegn.   det forbliver i
                                            bufferen.)
```

BIOS interrupts.

```
16H        2   AL := Tastaturets statusbyte. (Se side 263)
               bit 0 = Højre shift taste tryk.
                   1 = Venstre - - - " - - - .
                   2 = Ctrl/shift  - " - - - .
                   3 = Alt shift   - " - - - .
                   4 = Scroll lock.
                   5 = Numerisk lock.
                   6 = Caps lock.
                   7 = Ins aktiv.
```

Printer.(Parallel printer port - normalt 3 - LPT1,LPT2,LPT3)

```
17H        0   Send et tegn til printer.
               AL = Tegn.
               DX = Printernr. (0,1 el.2)
               AH := Statusbits se funktion 1.

17H        1   Initier printer porten.
               DX = Printernr.
               AH := Printerens status.
                     Bit 7 = Printer klar.
                         6 = ACK.
                         5 = Intet papir i printer.
                         4 = Printer selected.
                         3 = I/O fejl.
                         2 = Ubrugt.
                         1 = Ubrugt.
                         0 = Time out. (Valg af ej existeren-
                                        de printer).

17H        2   Hent printerportens status.
               DX = Printernr. (0,1 el. 2).
               AH := Printerens status - se under funktion 1.
```

Timer. AH

```
1AH        0   Læs nuværende klokværdi. (Time of Day).
               AL := 0 hvis midnat er passeret siden sidst.
               CX+DX := Timerens værdi.
               Timeren tæller 65.536 gange i timen eller om-
               kring 18,204 gange pr. sekund.

1AH        1   Sæt klokværdier.
               CX+DS = Time of Day med CX som high byte.
```

Diverse.

```
1BH            Tastaturets BREAK adresse d.v.s. hvortil der
               fortsættes ved BREAK.

1CH            Timer Tick Interrupt d.v.s. man får adressen
               på IRET ordren i interruptrutinen, som kal-
               des ca. 18,2 gange pr. sekund. Se INT 8.
1DH            Pointer til skærm parameter tabel.
1EH            Pointer til diskette parameter tabel.
1FH            Pointer til grafiktabel. Se side 226.
               Hvert tegn er 8 bytes lang.
```

Int 33h Function 00h V2
Initialize the Mouse

Determines whether a mouse is installed, resets the driver, and returns the number of buttons on the mouse

Calling Registers: AX 0000h

Return Registers: AX 0, mouse not installed
-1, mouse installed
BX Number of buttons (2 for Microsoft, 3 for some other brands)

Comments: When a program in which you want to use the mouse starts, that program must verify the mouse's presence. The usual way to do this is to call Function 00h. This function resets the mouse to the center of the screen, makes sure that the mouse is off, and sets the default mouse cursor and default movement ratios.

In DOS V2.x, it is best to verify that the vector for Interrupt 33h points to code before using this function. If the four bytes of the vector are not all 00, it should be safe to use Function 00h to determine whether the driver is present.

The initial conditions established for the mouse driver by this function are

Display page:	Page 0
Cursor range:	Entire screen (x=0 to 639, y=0 to 199)
Exclusion area:	None
Cursor position:	At screen center (x=320, y=100)
Cursor state:	Hidden
Cursor shape:	Arrow for graphics modes Reverse block for text modes
User Interrupts:	Disabled
Light pen emulator:	Enabled
Mickey/Pixel ratio:	Horizontal = 8 to 8 Vertical = 16 to 8
Speed threshold:	64 mickeys per second (Mickey is the unit of mouse motion. One mickey is approximately 1/200 inch.)

Int 33h Function 01h V2
Show Mouse Cursor

Causes the mouse cursor to appear on the display

Calling Registers: AX 0001h

Return Registers: None

Comments: Turns on the mouse cursor, allowing display on the screen. This is not an absolute function; rather, it increments an internal mouse-cursor flag. Initially, this flag is set to a −1. Whenever the flag is zero, the mouse cursor will be displayed. Function 02h decrements the cursor flag, causing the cursor to disappear if the original value was zero. Thus, multiple calls to Function 02h require multiple calls to this function. The software prevents the flag's value from becoming greater than zero, however.

Int 33h Function 02h V2
Hide Mouse Cursor

Turns off display of the mouse cursor

Calling Registers: AX 0002h

Return Registers: None

Comments: This function turns off the display function but does not disable the driver. As noted in the comment for Function 01h, Function 02h decrements a cursor flag. If the value is not zero, the cursor is turned off. Because the flag can never be greater than zero, a single call to this function is guaranteed to hide the cursor.

Int 33h Function 03h V2
Get Mouse Position

Returns current mouse position and button status

Calling Registers: AX 0003h

Return Registers: BX Button status
CX X-coordinate (horizontal)
DX Y-coordinate (vertical)

Comments: This function tells you where the mouse is located. No matter what mode the screen is in, Function 03h always returns an x-coordinate (column) between 0 and 639 and a y-coordinate (row) between 0 and 199. Table M.1 shows the mouse cursor's allowable positions for each display mode, in terms of screen (pixel) coordinates.

Table M.1. *Mouse Cursor's Position*

Screen Mode	*Mouse Coordinates*
00h, 01h	x = 16 × column y = 8 × row
02h, 03h	x = 8 × column y = 8 × row
04h, 05h	x = 2 × screen X y = Screen Y
06h	x = Screen X y = Screen Y
07h	x = 8 × screen column y = 8 × screen row
0Eh–10h	x = Screen X y = Screen Y

The status of the mouse buttons is returned in BX; only the low-order bits are significant. Table M.2 illustrates the meaning of the bits. Because each button acts independently, the value can be anything from 0 to 3 for a two-button mouse or 0 to 7 for the three-button version.

Table M.2. *Mouse Button Status Bits*

Bit 76543210	*Meaning*
.....0	Left button up
.....1	Left button down
.....0.	Right button up
.....1.	Right button down
.....0..	Center button (if present) up
.....1..	Center button (if present) down
xxxxx...	Undefined

Int 33h — Function 04h — V2

Set Mouse Position

Sets the mouse's position on the screen

Calling Registers:
- AX — 0004h
- CX — New x-coordinate (horizontal)
- DX — New y-coordinate (vertical)

Return Registers: None

Comments: You can use this function to place the mouse cursor anywhere on the screen. (The mouse driver will resume operating from that location.) This function is useful, for example, when you want to start the mouse at the first item on a menu that is brought up on the screen.

If either coordinate has a value inappropriate for the current screen mode as defined for Function 03h, it will be adjusted to the nearest appropriate value. If the specified position lies outside the display range established by calls to Functions 07h and 08h, the cursor will be placed as close to the specified position as possible while remaining within the range limits. If the position lies within an exclusion area defined by Function 10h, it will be hidden.

Int 33h — Function 05h — V2

Get Button-Press Information

Returns information about button presses

Calling Registers:
- AX — 0005h
- BX — Button
  - 0, left
  - 1, right
  - 2, center (if present)

Return Registers:
- AX — Button status (see table M.2)
- BX — Count of button presses (reset to zero at each call)
- CX — Cursor's horizontal position at last button press
- DX — Cursor's vertical position at last button press

5.1.d

Bjørk Busch

DOS-interrupt funktioner for skrivning på skærm/std.output

Funktion 02: Tegn output med Ctrl/break check

```
     Mov        Ah,02h
     Mov        Dl,Tegn          ; Tegn der skal udskrives
     Int        21h
```

Funktion 05: Tegn print - printerudgang

```
     Mov        Ah,06h
     Mov        Dl,Tegn          ; Tegn der skal udskrives
     Int        21h
```

Funktion 06: Tegn - direct Console I/O.

```
     Mov        Ah,06h
     Mov        Dl,Tegn          ; Tegn som udskr. (ikke
                FFh)
     Int        21h
```

Funktion 09: Streng udskrift

```
     Mov        Ah,06h
     Lea        Dx,Tekststr      ; streng afsluttes med $
     Int        21h
```

eks. på tekststr

```
     Tekststr  DB   'Tekstlinie for udskrift',13,10,'$'
```

13 er ascii-tegn carrige-return
10 er ascii-tegn linefeed

Funktion 40: Write blok (ikke linier - også til alm. filer)

```
     Mov        Ah,40h
     Mov        Bx,1             ; handle 1 = std. out.
                                 ;        2 = std. error
                                 ;        4 = print (prn:)
     Lea        Dx,Outputdata    ; adr. på buffer der
                                 ; skal skrives
     Mov        Cx,Outlen        ; recordlængde
     Int        21h
     JC         error            ; C-flag sat hvis fejl
     Cmp        Ax,Cx            ; Ax - faktisk skrevet
     JNE        error2           ; ikke alt skrevet
```

DOS-interrupt funktioner for læsning fra tastatur / std.input

Funktion 01: Tegn input med ECHO og med Ctrl/Break check

```
     Mov        Ah,01h
     Int        21h
     Mov        Tegn,Al          ; Hvis 00h så extended;
```

Funktion 07: Tegn input **UDEN** ECHO og **UDEN** Ctrl/Break check

Som funktion 01.

Funktion 08: Tegn input **UDEN** ECHO og **MED** Ctrl/Break check

Som funktion 01.

Funktion 06: Tegn - direct Console I/O.

```
     Mov        Ah,06h
     Mov        Dl,0FFh          ; for læsning
     Int        21h
     JZ         ingeninp         ; Z-flag sat hvis tom
                                 buffer
     Mov        Tegn,Al          ; Hvis 00h så extended
```

Funktion 0A: Buffered Input (linie - til carrige return):

```
     Mov        Ah,0Ah
     Lea        Dx,Inpbuf
     Int        21h
```

Før kald: 1.byte i Inpbuf skal indeholde længde af buffer excl. 2 byte (1 byte med max. længde og 2.byte med indlæst længde)

Efter kald: 2.byte indeholder faktiske antal indlæste tegn excl. carrige-return (ascii-tegn 13). 3-?? byte indeholder de indlæste tegn samt en afsluttende carrige-return.

Inpbuf kan defineres på følgende måde:

```
     Inpbuf     DB   20              ; max. længde incl. CR
     Inplen     DB   ?               ; til faktisk længde
     Inpdata    DB   20 DUP (?)      ; egentlig input
```

eller med brug af ekstra adresse-symbol og label

```
     Inpbuf     LABEL BYTE           ; startadr.
     Inpmax     DB   20              ; max. længde incl. CR
     Inplen     DB   ?               ; til faktisk længde
     Inpdata    DB   20 DUP (?)      ; egentlig input
```

når LABEL anvendes som her afsættes ikke plads, men der knyttes en adresse og type til symbol. Adresse bliver her den samme som for "Inpmax".

Funktion 3F: Read blok (ikke linier - også til alm. filer)

```
     Mov        Ah,3Fh
     Mov        Bx,0             ; handle 0 = std. inp.
     Lea        Dx,Inputdata     ; adr. på buffer til input
     Mov        Cx,Maxinp        ; recordlængde
     Int        21h
     JC         error            ; C-flag sat hvis fejl
     Cmp        Ax,0             ; Ax - faktisk indlæst
     Jz         EndOfFile        ;    der var EOF
```

5.1.e

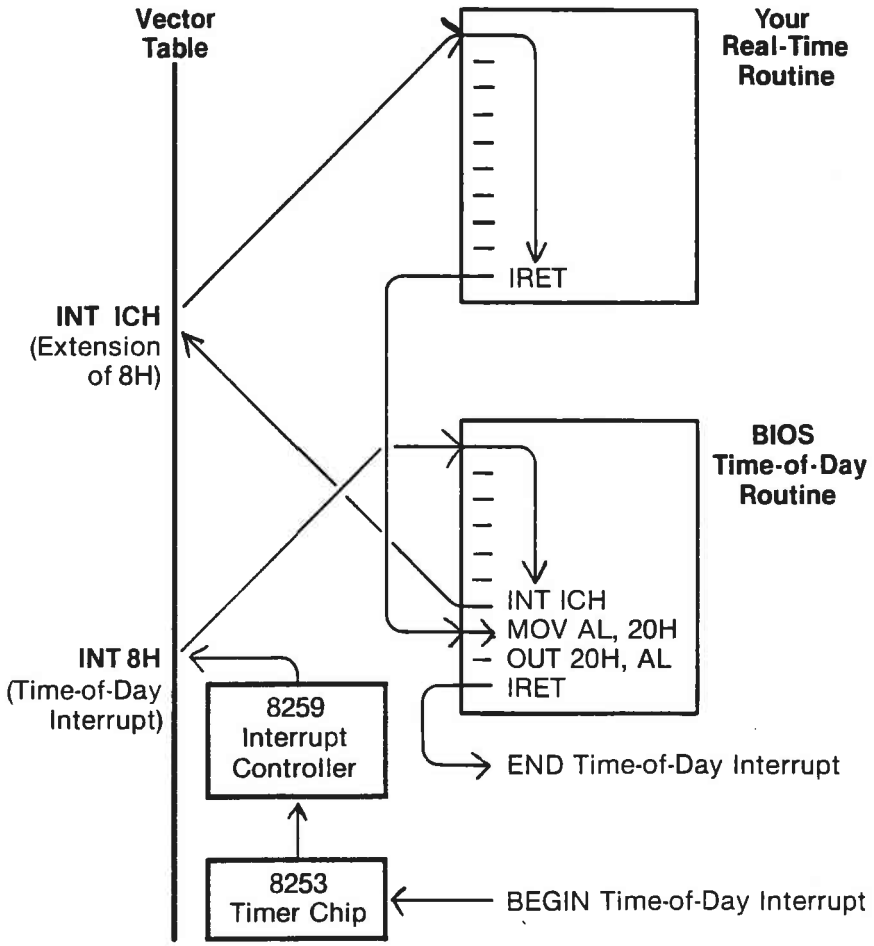
Selvdefinerede interruptrutiner

Bjørk Busch

Der er muligt for programmøren, selv at skrive nye interruptrutiner, såvel hardware- som software.

Man kan desuden overtage de oprindelige, idet man evt. inddrage det "gamle" ved, at flytte det til et andet interrutnr og kalde det fra sin egen interruprutine.

Timerinterruptet er på forhånd etableret, så det er let for programmøren, at udvide rutinen, idet man "blot" instalerer sig på interruptnr 1Ch.

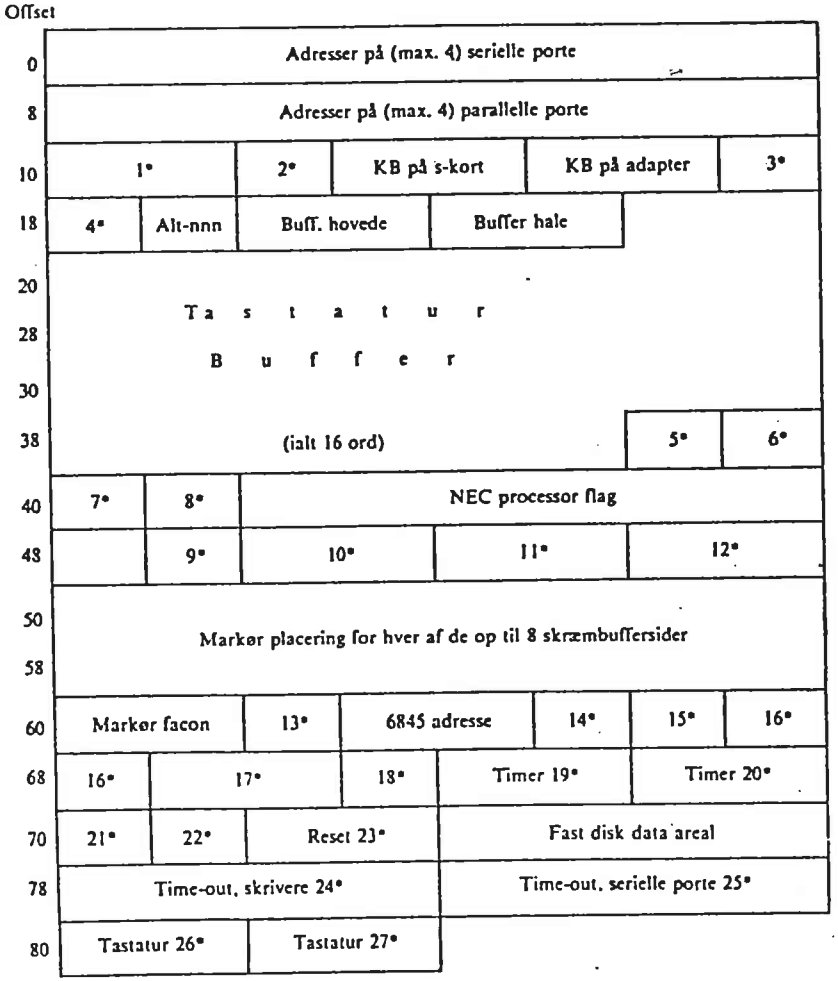


Extending the timer interrupt.

ROM BIOS data arealer

ROM BIOS data befinder sig i segment 40H fra offset 0 til 83H, altså på adresse 0000:0400 - 0000:0483.

Neden for er vist data områdets lay-out. Dernæst følger detailler for felter markeret i lay-outet med et tal og en stjerne.



1* Bit i "equipment" flag
Offset: 10H. Længde: 2 byte.

....1	Mindst et diskette drev
.... nn..	(nn+1)x16 KB RAM. nn=3 betyder mindst 64 KE
..nn	nn=01: 40x25 farve monitor nn=10: 80x25 farve monitor. nn=11: 80x25 monokrom monitor
nn..	nn+1 diskettedrev
.... nnn.	nnn serielle porte
....1	spil adapter installeret
.... nn..	nn parallelle porte (skrivere)

2* Offset adresse på data areal til fejlsøgning under fabrikation
Offset: 12H. Længde: 1 byte.

3* 1. tastatur flag
Offset: 17H. Længde: 1 byte.

....1	Højre skifte taste holdes nede
.... ..1.	Venstre skifte taste holdes nede
.... .1..	Ctrl taste holdes nede
.... 1...	Alt taste holdes nede
...1	Scroll Lock er aktiv
..1.	Num Lock er aktiv
.1..	Caps Lock er aktiv
1...	Ins er aktiv

4* 2. tastatur flag
Offset: 18H. Længde: 1 byte.

.... 1...	Ctrl-NumLock (pause) aktiv
...1	Scroll Lock taste holdes nede
..1.	Num Lock taste holdes nede
.1..	Caps Lock taste holdes nede
1...	Ins taste holdes nede

5* Diskette søge status
Offset: 3EH. Længde: 1 byte.

....0	Drev A behøver rekalibrering før næste søgning
.... ..0.	Drev B behøver rekalibrering før næste søgning
.... .0..	Drev C behøver rekalibrering før næste søgning
.... 0...	Drev D behøver rekalibrering før næste søgning

6* Diskette motor status
Offset: 3FH. Længde: 1 byte.

....	...1	Drev A's motor kører
....	..1.	Drev B's motor kører
....	.1..	Drev C's motor kører
....	1...	Drev D's motor kører
1...	....	Igangværende operation er skrivning

7* Time-out værdi til slukning af diskette motor
Offset: 40H. Længde: 1 byte.

8* Diskette status byte
Offset: 41H. Længde: 1 byte.

....	...1	Ugyldig kommando sendt til diskette adapter
....	..1.	Formateringsfejl
....	..11	Skrivning forsøgt på skrivebeskyttet diskette
....	.1..	Sektor ikke fundet
....	11..	DMA overløb
...1	....	64 KB grænse overskredet ved DMA
..1.	....	Bitfejl ved læsning
..1.	....	Fejl i NEC processor
.1..	....	Forgæves søgning
1...	....	Time-out

9* Aktuel video status
Offset: 49H. Længde: 1 byte.
0 = 40x25 sort/hvid tekst
1 = 40x25 farve tekst
2 = 80x25 sort/hvid tekst
3 = 80x25 farve tekst
4 = 320x200 farve grafik
5 = 320x200 sort/hvid grafik
6 = 640x200 sort/hvid grafik
7 = monokrom tekst

10* Antal søjler på skærm
Offset: 4AH. Længde: 2 byte. Værdi: 40 eller 80.

11* Skærmbuffer længde i bytes
Offset: 4CH. Længde: 2 byte.

12* Skærmbuffers relative start adresse
Offset: 4EH. Længde: 2 byte.

13* Sidenummer for aktuel side
Offset: 62H. Længde: 1 byte. Værdi: 0-7.

14* Register status
Offset: 65H. Længde: 1 byte.

15* Aktuel palet
Offset: 66H. Længde: 1 byte.

16* Kassetteports synkroniserings ord
Offset: 67H. Længde: 2 byte.

17* Bitfejl kontrol (CRC) for kassetteport
Offset: 69H. Længde: 2 byte.

18* Sidst læste/skrevne data byte fra kassetteport
Offset: 6BH. Længde: 1 byte.

19* Tælles 1 op hver 5/91 sekund
Offset: 6CH. Længde: 2 byte.

20* Tælles 1 op hver time. Nulstilles efter 1 døgn.
Offset: 6EH. Længde: 2 byte.

21* Overløb af tæller (ur) siden sidste aflæsning
Offset: 70H. Længde: 1 byte.

22* Bit 7 = 1, hvis Ctrl-Break har været aktiveret; ellers 0.
Offset: 71H. Længde: 1 byte.

23* 1234H, hvis Ctrl-Alt-Del genstart er igang; ellers FF00H.
Offset: 72H. Længde: 2 byte.

24* Hver byte angiver en skrivers time-out
Offset: 78H. Længde: 4x1 byte. Værdi 14H = 20 sek.

25* Hver byte angiver en seriel ports time-out
Offset: 7CH. Længde: 4x1 byte. Værdi: 1 sek.

26* Offset adresse for tastatur buffers start
Offset: 80H. Længde: 2 byte. Værdi: 001E.

27* Offset adresse for tastatur buffers slutning
Offset: 80H. Længde: 2 byte. Værdi: 003E.

Program Segment Prefix.

PSP'et er et område på 256 bytes (100 hex), som MS-DOS initierer ved opstart af programmet. Området er fælles for alle programmer startet under MS-DOS, og findes altså også i programmer i EXE-format.

De sidste 128 bytes af området indeholder eventuelle parametre, som i forbindelse med opstarten blev indtastet i kommandolinien efter programnavnet. Disse oplysninger kan nu blive læst af programmet, og danne grundlag for variationer i programmets eksakte virkemåde.

Den følgende skitse viser opbygningen af PSP'et:

Offset (hex)	Længde bytes	Beskrivelse
00	2	INT 20 maskininstruktion (stop program)
02	2	Segmentadressen for den sidste del af det lager, som er allokeret (tildelt) til programmet. Ved hjælp af denne information kan programmet beregne, hvor meget lager, der aktuelt er tilknyttet til det.
04	1	Reserveret.
05	5	Maskininstruktion, som kan anvendes til kald af MS-DOS funktionerne.
0A	4	INT 22 adresse.
0E	4	INT 23 adresse (Ctrl-C afbrydelse).
12	4	INT 24 adresse (afbrydelse på grund af kritisk fejl).
16	22	Reserveret.
2C	2	Environment segment adresse. I den såkaldte environment blok kan man indsætte forskellige former for information, som kan benyttes af operativsystemet og programmet.
2E	34	Reserveret.
50	3	INT 21 og RETF maskininstruktioner.
53	9	Reserveret.
5C	16	FCB#1
6C	16	FCB#2. De to FCB'er er et levn fra en tid, hvor man brugte en anden form for filbehandling i MS-DOS. Siden er denne erstattet af en langt mere moderne teknik.
80	1	Længden på kommandoliniens parametre.
81	127	Kommandoliniens parametre. Parametrene er indeholdt i en tekststreng, som begynder med tegnet umiddelbart efter kommandoens navn (et blanktegn). Herefter følger kommandoliniens parametre i uformatteret form, som de blev indtastet, med blanktegn, kommaer og andre skilletegn.

5.2.c

Environment-blok

Bjørk Busch

Environment-blokken anvendes som bindeled mellem et program og styresystemet. Ved opstart af et program vil der blive skabt en kopi af den aktuelle environment-blok til programmet. Programmet kan direkte få adgang til denne kopi gennem PSP'et, der ligeledes skabes i forbindelse med opstart af et program. Når programmet afslutter frigives hukommelsen til såvel den tilknyttede environment-blok som PSP og programmet selv. Dette gælder dog ikke for residente programmer.

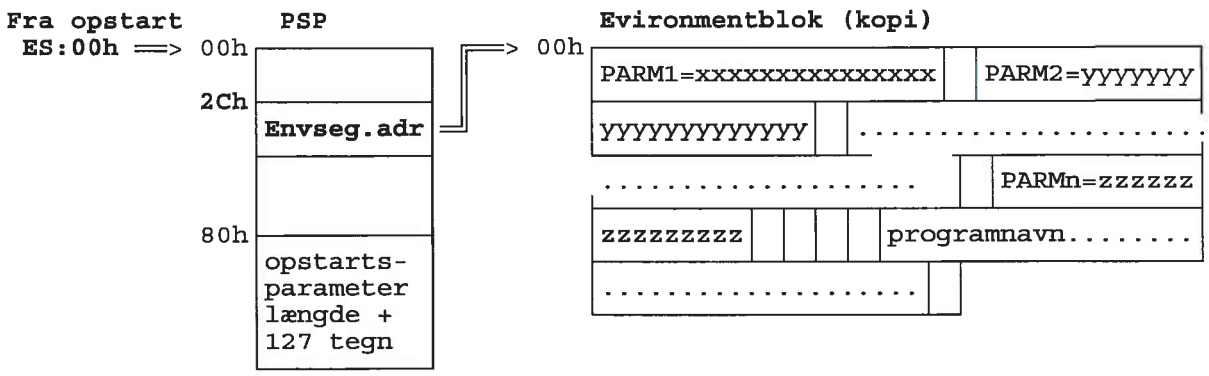
Systemvarable.

Environment-blokken består af en tabel med NUL-terminerede tekststrenge, der indeholder systemvariable.
Strengene indeholder ingen længdebyte, men de afsluttes hver især med en byte med binært-nul. Der er således ikke faste størelser på elementerne og tabellen må søges igennem sekventielt.
Tabellen med systemvariable afsluttes med en "ekstra" byte med binært nul, idet der således er 2 bytes med binært nul efter sidste systemvariabel.
Hver af element i tabellen indeholder først navnet på systemvariablen (med store bogstaver), herefter følger et lighedstegn og efter dette indholdet af systemvariablen.

Programnavn.

Efter tabellen med systemvariable følger en byte med 01h, herefter en byte med 00h og herefter en NUL-termineret streng med navnet på det opstartede program med fuld stinavn.
Den sidste beskrivelse er ikke i de officielle beskrivelser fra Microsoft, men kan være nyttig, idet man her kan skaffe sig navnet på det katalog, hvor programmet befinder sig.

Sammenkædning til program.



5.2.d

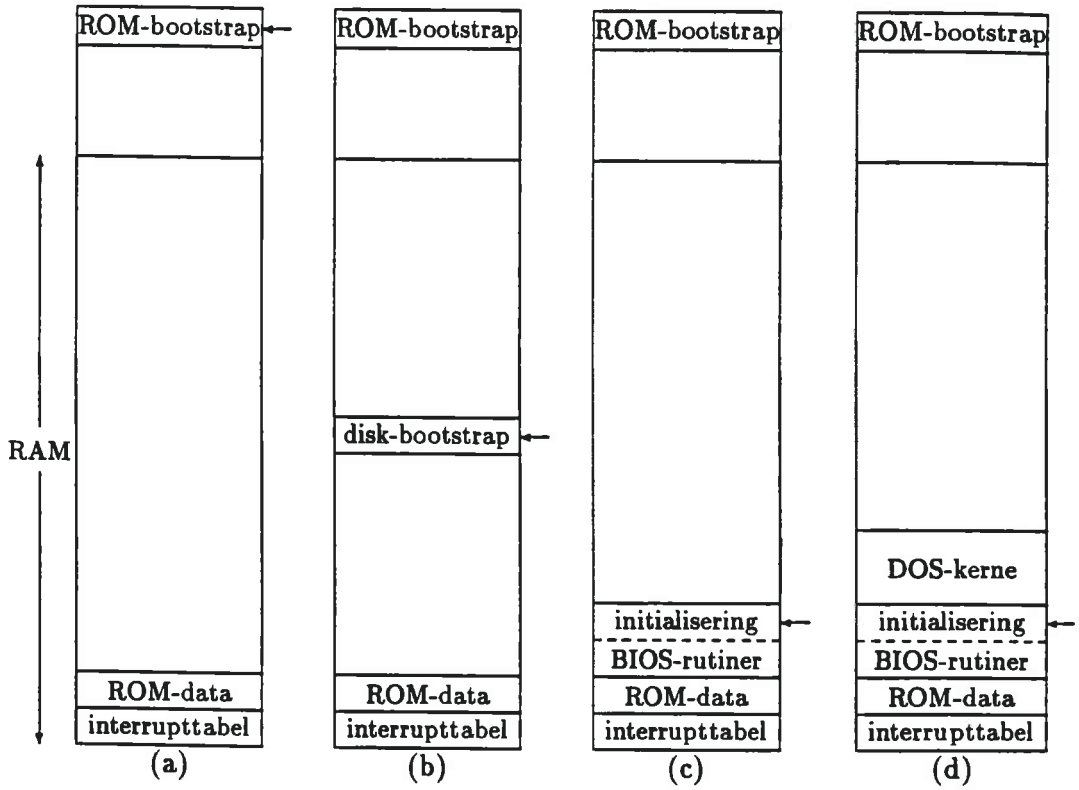
Adresserummet under DOS

Således anvendes memory under DOS

	XMS (extended memory) XMS kan ikke anvendes direkte i DOS programmer, men bruges til RAMDISK, CACHE m.m. Ved brug af EMM386 kan XMS anvendes til simulering af EMS	?? MB
110000h ---->	HMA (evt. flyttes DOS hertil)	64 KB
100000h ---->	ROM BIOS (incl. bootstrap)	16 KB
F0000h ---->	UPPER MEMORY (residente prog./dosdrivere)	64 KB
	ROM BIOS for video, harrddisk m.m.	
C0000h ---->	Video-memory COLOR TEKST/grafik	32 KB
B8000h ---->	Video-memory MONO TEKST/grafik	32 KB
B0000h ---->	Video-memory EGA/VGA grafik/ramfonte	64 KB
A0000h ---->	Transient program areal (brugerprogrammer)	640 KB
	Residente del af DOS	
	ROM BIOS og BASIC dataareal	
00500h ---->	ROM BIOS dataareal	
00400h ---->	Interrupt adresse tabel	
00000h ---->		

5.3.a

Opstartsfaser for DOS

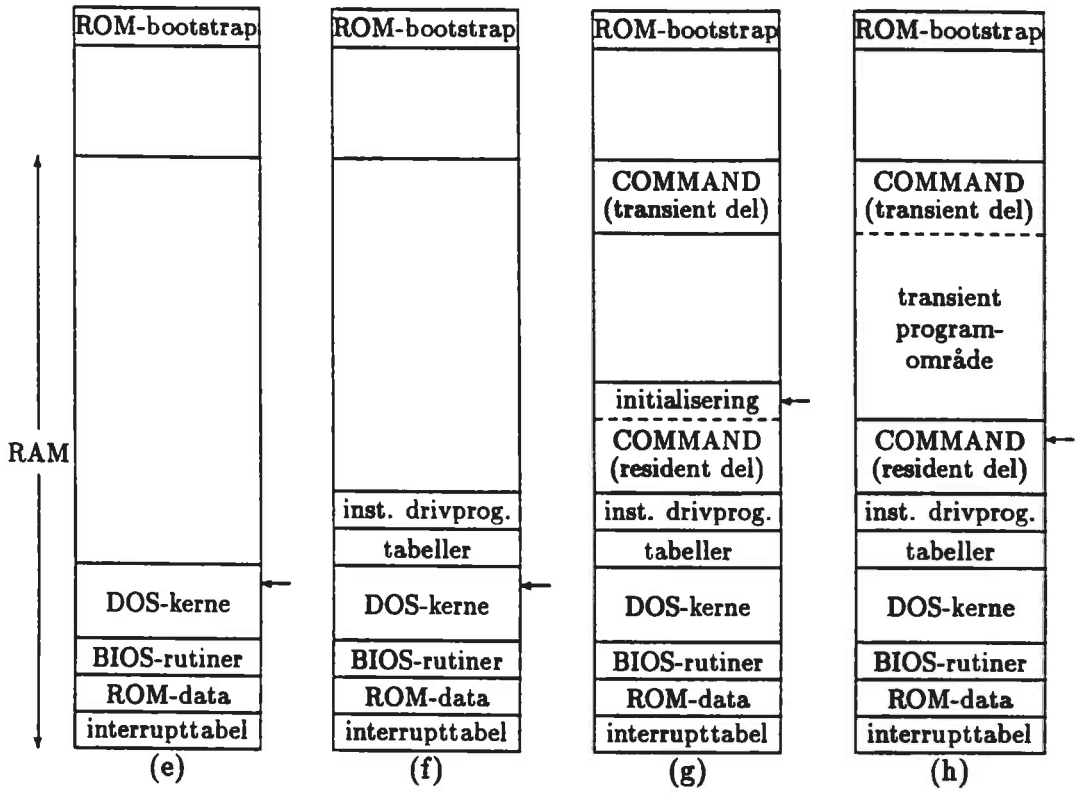


Figur 17.1: DOS-"boot"-proces

Maskinen starter, når der sættes strøm til systemet eller efter systemreset (aktiveret af den velkendte <Ctrl><Alt><Del>-tastekombination), udførelse af et program på adresse 0FFFF0H (i ROM). Dette program foretager en grundlæggende systemtest, initialiserer interruptvektortabellen og visse ROM BIOS-dataområder, og afslutter med udførelse af ROM-bootstrap programmet, som forsøger at indlæse en "diskboot-record" (se Figur 17.1 (a)). På figurerne er udførselspunktet antydet med en pil.

Hvis en sådan "boot-record" findes (på disketten i A:, eller på harddisken C:), indlæses diskbootstrapprogrammet til lageret, og udførelsen af det begyndes (se Figur 17.1 (b)), ellers startes den ROM residente BASIC fortolker.

Disk boot-record'en indeholder et lille program, der har til formål at kontrollere om DOS-systemfilerne findes i rodkataloget på boot-disken. Hvis disse filer er til stede (i korrekt rækkefølge, med IBMBIO.COM som den første fil på disken), indlæses IBMBIO.COM-filen til lageret. IBMBIO.COM-filen består egentlig af to komponenter, dels *BIOS-rutinerne* med forskellige hardwarespecifikke drivprogrammer, samt en *initialiseringsdel.* Initialiseringsdelen startes (Figur 17.1 (c)),



Figur 17.2: DOS-"boot"-proces (fortsat)

og blandt andet foretages her indlæsning af selve *DOS-kernen*, (Figur 17.1 (d)), fra filen IBMDOS.COM, og ved en lille tryllekunst relokeres denne til sin endelige position umiddelbart efter BIOS-rutinerne (Figur 17.2 (e)).

Herefter gives kontrol til DOS-kernen (Figur 17.2 (e)), som opbygger sine interne arbejdstabeller for diskfiladministration og lageradministration. Det er på dette tidspunkt, at en eventuel CONFIG.SYS-fil indlæses og behandles, og installering af eventuelle *installerbare drivprogrammer* foretages (Figur 17.2 (f)).

Nu er DOS som sådan på plads, og det sidste trin i bootprocessen er indlæsning af en *user-shell*, som i de fleste tilfælde blot er den sædvanlige kommandofortolker (fra COMMAND.COM-filen). DOS-kernen indlæser COMMAND.COM og påbegynder udførelsen af initialiseringsdelen, som blandt andet foretager behandling af en eventuel AUTOEXEC.BAT-fil.

Tilsidst får den egentlige kommandofortolker kontrol, tilkendegivet ved, at den viser sin "prompt", hvor den afventer indtastning af brugerkommandoer (Figur 17.2 (g) og (h)). Lagerområdet benyttet til initialiseringsdelen bliver frigivet, og det udgør sammen med resten af RAM'en og området hvortil den transiente del af kommandofortolkeren er indlæst, det såkaldte *transiente programområde*, hvori DOS kan bringe andre programmer til udførelse.