INT 2F 1508 - CD-ROM - ABSOLUTE DISK READ

Category: d - disk I/O enhancements

Inp.:
- AX = 1508h
- ES:BX -> buffer
- CX = drive number (0=A:)
- SI:DI = starting sector number
- DX = number of sectors to read

Return: CF set on error
- AL = error code (0Fh invalid drive,15h not ready)
- CF clear if successful

Note: returns error 15h (not ready) under Windows95 if the starting sector number is less than 10h

SeeAlso: AX=1509h
INT 2F
Copied from Ralf Brown's Interrupt List

INT 31 0300 - DPMI 0.9+ - SIMULATE REAL MODE INTERRUPT

Category: E - DOS extenders

Inp.:
- AX = 0300h
- BL = interrupt number
- BH = flags
  - bit 0: reset the interrupt controller and A20 line (DPMI 0.9)
    reserved, must be 0 (DPMI 1.0+)
  - others: reserved, must be 0
- CX = number of words to copy from protected mode to real mode stack
- ES:(E)DI = selector:offset of real mode call structure (see #03148)

Return: CF clear if successful
- real mode call structure modified (all fields except SS:SP, CS:IP filled with return values from real mode interrupt)
- CF set on error
- AX = error code (DPMI 1.0+) (8012h,8013h,8014h,8021h)(see #03143)
- protected mode stack unchanged

Notes: 16-bit programs use ES:DI as pointer, 32-bit programs use ES:EDI
- CS:IP in the real mode call structure is ignored for this call,
  instead, the indicated interrupt vector is used for the address
  the flags in the call structure are pushed on the real mode stack to
  form an interrupt stack frame, and the trace and interrupt flags are
  clear on entry to the handler
- DPMI will provide a small (30 words) real mode stack if SS:SP is zero
  the real mode handler must return with the stack in the same state as

Republished - 12th November 2007 – Robert Yates
it was on being called

SeeAlso: AX=0302h, AX=FF01h, INT 21/AX=2511h, INT 21/AH=E3h "OS/286"
SeeAlso: INT 2C/AX=0026h, INT 2F/AX=FB42h/BX=000Dh

INT 31
;---------------------------------------------------------------------------------
Table 03148
Format of DPMI real mode call structure:
Offset Size Description
00h DWORD EDI
04h DWORD ESI
08h DWORD EBP
0Ch DWORD reserved (00h)
10h DWORD EBX
14h DWORD EDX
18h DWORD ECX
1Ch DWORD EAX
20h WORD flags
22h WORD ES
24h WORD DS
26h WORD FS
28h WORD GS
2Ah WORD IP
2Ch WORD CS
2Eh WORD SP
30h WORD SS
INT 31 0300
;---------------------------------------------------------------------------------
Table 03143
Values for DPMI 1.0 error code:
0000h-7FFFh DOS error passed through by DPMI
8001h unsupported function
8003h system integrity would be endangered
8004h deadlock detected
8005h pending serialization request cancelled
8010h out of DPMI internal resources
8011h descriptor unavailable
8012h linear memory unavailable
8013h physical memory unavailable
8014h backing store unavailable
8015h callback unavailable
8016h handle unavailable
8017h maximum lock count exceeded
8018h shared memory already serialized exclusively by another
8019h shared memory already serialized shared by another client
8021h invalid value for numeric or flag parameter
8022h invalid segment selector
8023h invalid handle
8024h invalid callback

Republished - 12th November 2007 – Robert Yates
8025h  invalid linear address
8026h  request not supported by hardware
INT 31 0000

;--------------------------------------------------------------------------
INT 2F 1510 - CD-ROM v2.10+ - SEND DEVICE DRIVER REQUEST

Copied from Ralf Brown's Interrupt List

Inp.:
AX = 1510h
CX = CD-ROM drive letter (0 = A, 1 = B, etc)
ES:BX -> CD-ROM device driver request header (see #02597 at AX=0802h)

Return: CF clear if device driver has been called (check the request header's
        status word to determine whether an error has occurred)
        ES:BX request header updated
        CF set if device driver has not been called
        AX = error code (000Fh = invalid drive, 0001h = invalid function)
        ES:BX request header unchanged

Notes:  MSCDEX initializes the device driver request header's subunit field
        based on the drive number specified in CX
        MSCDEX v2.21 through v2.25 (at least) return error code AX=0001h if
        nested calls are attempted

BUGS:   Novell DOS 7 NWCDEX prior to the 12/13/94 update did not initialize
        the subunit field
        Windows95 sets CF if CX isn't a CD-ROM drive but leaves CF unchanged
        if the drive is in fact a CD-ROM

SeeAlso: AX=0802h
;--------------------------------------------------------------------------

Table 02597
Format of device driver request header:
Offset  Size    Description
 00h    BYTE    length of request header
 01h    BYTE    subunit within device driver
 02h    BYTE    command code (see #02595)
 03h    WORD    status (filled in by device driver) (see #02596)
---DOS---
 05h  4 BYTES  reserved (unused in DOS 2.x and 3.x)
 09h  DWORD   (European MS-DOS 4.0 only) pointer to next request header in
        device's request queue
        (other versions) reserved (unused in DOS 2.x and 3.x)
---STARLITE architecture---
 05h  DWORD   pointer to next request header
 09h  4 BYTES  reserved
---command code 00h---
 0Dh    BYTE    (ret) number of units
 0Eh  DWORD   (call) pointer to DOS device helper function (see #02599)
        (European MS-DOS 4.0 only)
        (call) pointer past end of memory available to driver (DOS 5+)
        (ret) address of first free byte following driver

Republished - 12th November 2007 – Robert Yates
12h DWORD (call) pointer to commandline arguments
(ret) pointer to BPB array (block drivers) or
0000h:0000h (character drivers)

16h BYTE (DOS 3.0+) drive number for first unit of block driver (0=A)
---European MS-DOS 4.0---
17h DWORD pointer to function to save registers on stack
---DOS 5+ ---
17h WORD (ret) error-message flag
0001h MS-DOS should display error msg on init failure

---command code 01h---
0Dh BYTE media descriptor
0Eh BYTE (ret) media status
00h don't know
01h media has not changed
FFh media has been changed

0Fh DWORD (ret, DOS 3.0+) pointer to previous volume ID if the
OPEN/CLOSE/RM bit in device header is set and disk changed

---command code 02h---
0Dh BYTE media descriptor
0Eh DWORD transfer address
-> scratch sector if NON-IBM FORMAT bit in device header set
-> first FAT sector otherwise
12h DWORD pointer to BPB (set by driver) (see #01663 at INT 21/AH=53h)

---command codes 03h,0Ch---
(see also INT 21/AX=4402h"DOS 2+",INT 21/AX=4403h"DOS")
0Dh BYTE media descriptor (block devices only)
0Eh DWORD transfer address
12h WORD (call) number of bytes to read/write
(ret) actual number of bytes read or written

---command codes 04h,08h,09h (except Compaq DOS 3.31, DR DOS 6)---
0Dh BYTE media descriptor (block devices only)
0Eh DWORD transfer address
12h WORD byte count (character devices) or sector count (block devices)
14h WORD starting sector number (block devices only)
16h DWORD (DOS 3.0+) pointer to volume ID if error 0Fh returned
1Ah DWORD (DOS 4.0+) 32-bit starting sector number (block devices with
device attribute word bit 1 set only) if starting sector
number above is FFFFh (see INT 21/AH=52h)

---command codes 04h,08h,09h (Compaq DOS 3.31, DR DOS 6)---
0Dh BYTE media descriptor (block devices only)
0Eh DWORD transfer address
12h WORD byte count (character devices) or sector count (block devices)
14h DWORD 32-bit starting sector number (block devices only)

Note: to reliably determine which variant of the request block for
functions 04h,08h,09h has been passed to the driver, check
the length field as well as the word at offset 14h. If the
length is 1Eh and 14h=FFFFh, use the DWORD at 1Ah as the
starting sector number; if the length is 18h, use the DWORD
at 14h; otherwise, use the WORD at 14h.
---command code 05h---
0Dh   BYTE   byte read from device if BUSY bit clear on return
---command codes 06h,07h,0Ah,0Bh,0Dh,0Eh,0Fh---

no further fields

---command code 10h---
0Dh   BYTE   unused
0Eh   DWORD   transfer address
12h   WORD   (call) number of bytes to write
(ret) actual number of bytes written
---command codes 11h,12h---
0Dh   BYTE   reserved
---command code 15h---
no further fields
---command codes 13h,19h---
0Dh   BYTE   category code
00h unknown
01h COMn:
03h CON
05h LPTn:
07h mouse (European MS-DOS 4.0)
08h disk
9Eh (STARLITE) Media Access Control driver
0Eh   BYTE   function code
00h (STARLITE) MAC Bind request
0Fh   WORD   copy of DS at time of IOCTL call (apparently unused in DOS 3.3)
SI contents (European MS-DOS 4.0)
11h   WORD   offset of device driver header (see #01646)
DI contents (European MS-DOS 4.0)
13h   DWORD   pointer to parameter block from INT 21/AX=440Ch or AX=440Dh
---command codes 80h,82h---
0Dh   BYTE   addressing mode
00h HSG (default)
01h Phillips/Sony Red Book
0Eh   DWORD   transfer address (ignored for command 82h)
12h   WORD   number of sectors to read
(if 0 for command 82h, request is an advisory seek)
14h   DWORD   starting sector number
logical sector number in HSG mode
frame/second/minute/unused in Red Book mode
(HSG sector = minute * 4500 + second * 75 + frame - 150)
18h   BYTE   data read mode
00h cooked (2048 bytes per frame)
01h raw (2352 bytes per frame, including EDC/ECC)
19h   BYTE   interleave size (number of sectors stored consecutively)
1Ah   BYTE   interleave skip factor
(number of sectors between consecutive portions)
---command code 83h---
0Dh   BYTE   addressing mode
00h HSG (default)
01h Phillips/Sony Red Book
0Eh DWORD transfer address (ignored)
12h WORD number of sectors to read (ignored)
14h DWORD starting sector number (see also above)
---command code 84h---
0Dh BYTE addressing mode
  00h HSG (default)
  01h Phillips/Sony Red Book

0Eh DWORD starting sector number (see also above)
12h DWORD number of sectors to play
---command codes 85h,88h---
no further fields
---command codes 86h,87h---
0Dh BYTE addressing mode
  00h HSG (default)
  01h Phillips/Sony Red Book
0Eh DWORD transfer address (ignored in write mode 0)
12h WORD number of sectors to write
14h DWORD starting sector number (also see above)
18h BYTE write mode
  00h mode 0 (write all zeros)
  01h mode 1 (default) (2048 bytes per sector)
  02h mode 2 form 1 (2048 bytes per sector)
  03h mode 2 form 2 (2336 bytes per sector)
19h BYTE interleave size (number of sectors stored consecutively)
1Ah BYTE interleave skip factor
  (number of sectors between consecutive portions)