

Note: This API calls are shared between DOS and Win16 personality.

DPMI is a shared interface for DOS applications to access Intel 80286+ CPUs services. DOS DMPI host provides core services for protected mode applications. Multitasking OS with DOS support also provides DMPI in most cases. Windows standard and extended mode kernel is a DPMI client app. Standard and extended mode kernel differs minimally and shares common codebase. Standard Windows kernel works under DOSX extender. DOSX is a specialized version of 16-bit DPMI Extender (but it is standard DPMI host). Standard mode is just DPMI client, exnhanced mode is DPMI client running under Virtual Machime Manager (really, multitasker which allow to run many DOS sessions). Both modes shares DPMI interface for kernel communication. The OS/2 virtual DOS Protected Mode Interface (VDPMI) device driver provides Version 0.9 DPMI support for virtual DOS machines. Win16 (up to Windows ME) provides Version 0.9 DPMI support. Windows in Standard Mode provides DPMI services only for Windows Applications, not DOS sessions.

DPMI host often merged with DPMI extender. Usually DPMI extender provide DPMI host standard services and DOS translation or True DPMI services.

2021/08/05 10:15 · prokushev · 0 Comments

Int 31H, AH=03H, AL=05H

Version

0.9

Brief

Get State Save/Restore Addresses

Input

```
AX = 0305H
```

Return

```
Carry flag = clear (this function always succeeds)
AX = size of buffer in bytes required to save state
BX:CX = real mode address of routine used to save/restore state
SI:(E)DI = protected mode address of routine used to save/restore state
```

Notes

Returns the addresses of two procedures used to save and restore the state of the current task's registers in the mode which is not currently executing.

The real mode address returned by this function in BX:CX is called only in real mode to save/restore the state of the protected mode registers. The protected mode address returned by this function in SI:(E)DI is called only in protected mode to save/restore the state of the real mode registers; 16-bit programs should call the address in SI:DI, 32-bit programs should call the address in SI:EDI. Registers for the current mode can be saved by simply pushing them on the stack.

Both of the state-save procedures are entered by a FAR CALL with the following parameters:

AL = 0 to save state, or 1 to restore state ES:(E)DI = (selector or segment):offset of state-save buffer

The state-save buffer must be at least as large as the value returned in AX by Int 31H Function 0305H. The state save/restore procedures do not modify any registers. For a further discussion of use of the state save/restore procedures, see that page.

Some DPMI hosts will not require the state to be saved, indicating this by returning a buffer size of zero in AX. In such cases, the addresses returned by this function can still be called, although they will simply return without performing any useful function.

Clients do not need to call the state save/restore procedures before using Int 31H Functions 0300H, 0301H, or 0302H. The state save/restore procedures are provided specifically for clients that use the raw mode switch services.

A client can use the function to save its state in the destination mode before switching modes using the raw mode switch or issuing real-mode calls from a protected mode hardware interrupt handler. Refer to that page for the detailed information on stacks and mode switching.

See also

Note

Text based on http://www.delorie.com/djgpp/doc/dpmi/

DPMI	
Process manager	**INT 2FH** 1680H, 1687H
Signals	
Memory manager	
Misc	**INT 2FH** 1686H, 168AH
Devices	

2021/08/13 14:23 · prokushev · 0 Comments

From:
http://osfree.su/doku/ - **osFree wiki**

Permanent link:
http://osfree.su/doku/doku.php?id=en:docs:dpmi:api:int31:03:05

Last update: **2021/08/27 03:51**