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=06H

Version

0.9

Brief

Get Raw Mode Switch Addresses

Input

```
AX = 0306H
```

Return

```
Carry flag = clear (this function always succeeds)
BX:CX = real-to-protected mode switch address
SI:(E)DI = protected-to-real mode switch address
```

Notes

The address returned in BX:CX must only be called in real mode to switch into protected mode. The address returned in SI:(E)DI must only be called in protected mode to switch into real mode; 16-bit programs should call the address returned by this function in SI:DI, while 32-bit programs should call the address returned in SI:EDI.

The mode switch procedures are entered by a FAR JMP to the appropriate address with the following parameters:

```
AX = new DS
CX = new ES
DX = new SS
(E)BX = new (E)SP
SI = new CS
(E)DI = new (E)IP
```

The processor is placed into the desired mode, and the DS, ES, SS, (E)SP, CS, and (E)IP registers are updated with the specified values; in other words, execution of the client continues in the requested mode at the address provided in registers SI:(E)DI. The values specified to be placed into the segment registers must be appropriate for the destination mode; if invalid selectors are supplied when switching into protected mode, an exception will occur.

The values in (E)AX, (E)BX, (E)CX, (E)DX, (E)SI, and (E)DI after the mode switch are undefined. (E)BP will be preserved across the mode switch call so it can be used as a pointer. On an 80386 or later CPU, the FS and GS segment registers will contain zero after the mode switch.

If interrupts are disabled when the mode switch procedure is invoked, they will not be re-enabled by the DPMI host (even temporarily).

It is up to the client to save and restore the state of the task when using this function to switch modes. This usually requires using the state save/restore procedures whose addresses are returned by Int 31H Function 0305H.

Clients may find it more convenient to use Int 31H Functions 0300H, 0301H, and 0302H for mode switching than this function.

See also

Note

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

DPMI	
Process manager	**INT 2FH** 1680H, 1687H
Signals	
Memory manager	

DPMI	
Misc	**INT 2FH** 1686H, 168AH
Devices	

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