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=08H, AL=00H

Version

0.9

Brief

Physical Address Mapping

Input

```
AX = 0800H
BX:CX = physical address of memory
SI:DI = size of region to map (bytes)
```

Return

```
if function successful
Carry flag = clear
BX:CX = linear address that can be used to access the physical memory
```

```
if function unsuccessful
Carry flag = set
AX = error code
8003H   system integrity (DPMI host memory region)
8021H   invalid value (address is below 1 MB boundary)
```

Notes

Converts a physical address into a linear address. This function allows device drivers running under DPMI hosts which use paging to reach physical memory that is associated with their devices above the 1 MB boundary Examples of such devices are the Weitek numeric coprocessor (usually mapped at 3 GB), buffers that hold scanner bit maps, and high-end displays that can be configured to make display memory appear in extended memory.

It is the caller's responsibility to allocate and initialize a descriptor for access to the memory.

This function should only be used by clients that absolutely require direct access to a memory mapped device at physical addresses above 1 MB. Clients should not use this function to access memory below the 1 MB boundary (the real mode addressable region). See also Int 31H Functions 0002H, 0508H, and 0509H.

When this function is called, the DPMI host either creates page table entries that directly map the physical addresses requested and returns the linear address of the created page table entries, or else just returns the linear address of the memory region that is already used to map the requested device. For example, if the client attempts to map a Weitek coprocessor and the host already has a linear region set up to map the Weitek chip and virtualize it, it would simply return the linear address of the existing region. If the host does not virtualize the Weitek chip, it would create 16 page table entries that map the 64 KB Weitek address space and return a linear address corresponding to the new page table entries.

If the host is not virtualizing the device, it must disable any memory caching on the mapped pages; in particular, on the 80486 the host must set the PCD (page cache disable) bit in the page table entries.

The host is permitted to fail any memory mapping call. However, the host should support this function whenever possible, to achieve compatibility with application programs that use memory-mapped devices of which the host is not aware. Useful guidelines are that the host should fail any attempt to map addresses below 1 MB, or addresses which the host considers to be general-purpose RAM memory. Attempts to map any other physical address should succeed, since the host should either (a) already know about the device and be able to return a linear address used to access the device, or (b) assume the program is attempting to map a legitimate device of which the host has no knowledge.

Programs and device drivers which need to perform DMA I/O to physical addresses in a virtualized hardware environment should use the Virtual DMA Services (see the Glossary entry for the Virtual DMA Services Specification). Also see page 10 of the DPMI execution environment section.

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