

Re: Role of the code segment register during "far" CALLs

Source:

<http://www.tech-archive.net/Archive/Development/microsoft.public.win32.programmer.kernel/2006-08/msg00826.htm>

- *From:* "Ranjit" <ranjitiyer@xxxxxxxxxx>
 - *Date:* 20 Aug 2006 11:06:24 -0700
-

Got it..Thank you everyone.

Ranjit

Carl Daniel [VC++ MVP] wrote:

Ranjit wrote:

Thanks Anton,

I read some of your article from which i understood that Windows doesn't use call gates to implement intersegment calls. I will continue reading it to understand how intersegment far calls are made across different privilege levels. Should be interesting.

But how is a cross-segment call within the *same* privilege levels pulled off – eg, a call from my main method into CoInitialize, which is a user-level library call and the code segment of which would have the privilege level as the code segment of my "main" program. Is the code segment selector for ole32.dll's code section embedded within the address of the CoInitialize function? Also, why do i not see the value of the CS register change when that CALL to CoInitialize happened even though this was a far call (am assuming)

It's simply a near call – CS isn't changed – EVER – unless you transition into the kernel via one of the mechanisms Anton has already explained.

If you're reading something about how Windows works that's implying that each module has it's own segments, then you're reading about 16-bit windows. 32- and 64-bit windows have always used the "flat model" that basically (or completely, in the case of IA-64) ignores segmentation.

–cd

Re: Role of the code segment register during "far" CALLs