

Re: Seeing VERSIONINFO under Vista?

Source: <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.mfc/2007-05/msg02150.html>

- *From:* Joseph M. Newcomer <newcomer@xxxxxxxxxxxxx>
 - *Date:* Mon, 28 May 2007 13:23:47 -0400
-

The pin limitation was technical decision based on the amount of silicon required to create more output pins, the physical package limitations (per-pin cost is one of the dominant costs of the packaging), and the likelihood that more pins will be needed. For example, the AMD64 provides only 40 output pins allowing only 2**40 bytes of physical memory (well, that's 1TB if I've counted correctly). By the time memory technology gets to the point where we can put 1TB on a machine (it wasn't too many years ago when I was impressed by a site that had 1GB machines connected to 1TB disk systems; most of my machines have >1GB and I just bought a 1TB hard drive for \$400) we will see more output pins on the chips. All they need are a few years' "head room". The original PC supported 1MB of physical memory, because nobody could afford more than 64K. I remember upgrading our first PCs to 512M of memory, at outrageous expense. Then I bought my PC/AT with 3MB of memory. Today, I consider 2GB the minimum I'd buy.

On Mon, 28 May 2007 12:49:54 +0100, Daniel James <wastebasket@xxxxxxxxxxxxxxxxxxxxx> wrote:

In article news:<3cfj53dorqk36q109hdk9r5to5sv7fmm1i@xxxxxxx>, Joseph M. Newcomer wrote:

This was not an architectural restriction of the x86 family (which always had supported 32-bit addresses, however weirdly configured) but a pin limitation issue in the implementaiton of the 286.

That's not true in any meaningful sense -- the x86 architecture always supported addresses consisting of a 16-bit segment (or selector) part and a 16-bit offset, but there was never, on any x86 processor, an addressing mode that simply combined them to make a 32-bit linear address (and, as you correctly point out, the CPU didn't have enough pins to address that amount of RAM anyway).

It wasn't until the '386 that 32-bit addressing became possible, and that used 32-bit registers to contain the 32-bit linear address, with the selector registers used to control higher-level memory management.

it had been designed so that once it entered virtual mode, it could not be reset to real mode

Re: Seeing VERSIONINFO under Vista?

The reasoning, of course, from the CPU designers' POV was that if you were using protected-mode code you would have a protected-mode operating system (e.g. Xenix) and would not need to go back to real mode except for a full restart. The reasoning from the BIOS designers' POV was that you might want to get at the other 15MB of addressing space from real mode (e.g. for a RAM-disk) so they had to allow SOME way to do it.

Yes, it was a definite failure of the chip designer to see reality, and reality was that you needed the BIOS. And real-mode device drivers. So the hack was to provide a way to return to real mode. Later chip sets did not suffer from this delusion, and therefore it was easy to return to real mode if you needed it.

What it did was have a line in one of the I/O ports that actually reset the processor (I have this vague memory of port 60h or 61h). So when the processor reset, it started up in real mode.

Yes, I recall the hack. The PC tells the keyboard controller to apply a reset to the whole system -- very slow and painful.

You do know that Windows didn't use that technique, don't you? Some clever chappie discovered that if you load the interrupt descriptor table with an invalid address and then cause an interrupt the CPU then causes another interrupt to report the 'no IDT' fault condition, causes another interrupt to report the fault again, which blows the (very small) hardware interrupt stack and then resets the CPU to real mode ... this is much quicker than getting the keyboard controller to apply an external reset pulse.

<http://www.x86.org/productivity/triplefault.htm>

The reason for the SysEnter instruction was that when Intel went to places like Microsoft, Sun (Solaris/86), and I'm not sure if it was SCO or Netware for Unix, the folks who did FreeBSD, and I think Linux was out at the time, and others, and said "what can we do to enhance the chip set", the answer was uniformly "a fast kernel entry mechanism". This was the performance bottleneck. The double-fault technique was faster than the reset technique, but it was still unacceptable. The overhead for Int 2E (the old Windows kernel entry call) was also too high. The result was the SysEnter instruction for at least the virtual entry, and I think there is also a much better real-mode entry, which I think is required for doing things like motherboard power-down.

joe

Cheers,
Daniel.

Re: Seeing VERSIONINFO under Vista?

Joseph M. Newcomer [MVP]

email: newcomer@xxxxxxxxxxxx

Web: <http://www.flounder.com>

MVP Tips: http://www.flounder.com/mvp_tips.htm

.