Why is the maximum boot.ini delay 11 million seconds?

devblogs.microsoft.com/oldnewthing/20081113-00

November 13, 2008

Raymond Chen

I mentioned in passing that the maximum delay you can specify in boot.ini is about 11 million seconds. I'm disappointed but sadly not surprised that everybody focused on that number and completely missed the point of the article.

First of all, the value of 11 million was not a conscious limitation. It's just an artifact of other limitations. The delay is specified in seconds in boot.ini, but internally it is converted to BIOS clock ticks. (Remember, this is a boot loader; there's not much infrastructure available yet.) The conversion is done in 32-bit arithmetic, and 4 billion BIOS clock ticks at 18.2 ticks per second is about 110 million seconds.

Wait, but you said 11 million seconds, not 110 million seconds.

Well, yes, but the conversion is done in 32-bit integer arithmetic: `BiosTicks = BootIniSeconds * 182 / 10`. A value for BootIniSeconds larger than about 11 million will result in a signed integer overflow in the intermediate product.

That's why the limit is 11 million seconds. It's a storage limitation, not an arbitrary cutoff. And it is indeed the limit of a natural data type, just hidden behind some other computations. (SB figured this out, though I fear that this shout-out will only serve to encourage people to speculate even more wildly than they already do in the hopes of earning a future shout-out.)

Now you'd think this would be plenty enough of a boot delay. I mean, it's already much longer than the maximum amount you can delay with a single call to `Sleep` or `WaitForSingleObject`. And who would ever need a computer to pause *four months* before booting? Yet there's one commenter who thought 11 million seconds wasn't enough: "What if I want a timeout of 129 days?" If you need a timeout of 129 days, then go buy a custom hardware timer device that waits 129 days after the onset of power before allowing current to flow. This same commenter has difficulty comprehending that just because you're capable of delaying something for a specific length of time and you're capable of never doing it at all, that doesn't mean that you are also capable of delaying something for an arbitrary length of time.

One commenter suggested that the problem could be solved by using a bignum library. Before you solve a problem, you first have to be sure that what you have really is a problem in the first place. I have yet to hear of any customers who are running into the 11 million second limit. Using bignums here would be a solution in search of a problem. "Hey, let's consume valuable developer and tester resources in order to add a bunch of code to a tightly-constrained environment to cover a case that *nobody cares about*." (And I failed to live up to my promise to Ulric merely to incorporate this lecture by reference in the future.)

Raymond Chen

Follow