Subject: Re: About the bits reserved for float variable
Posted by David Fanning on Fri, 21 May 2004 15:07:25 GMT
View Forum Message <> Reply to Message

James Kuyper writes:

> It can do that by not representing every integer value in that range. A
> 32-bit type can represent a maximum of 2^32 different values. An
> ordinary 32 bit integer type represents 2^32 consecutive integer values.
> A 32-bit IEEE format floating point number represents a slightly smaller
> set of values (because some of the bit patters represent +infinity,
> -infinity, denormalized numbers, and NaNs), but those values are very
> closely spaced near 0, and more widely spaced out the larger the values
> are, which allows them to cover a much larger dynamic range.
>
> To be specific, an IEEE format number contains a sign bit, a mantissa,
> an exponent, and has an implicit offset which is used to interpret the
> value. The value represented by such a number is
>
> (-1)^sign *(1 + mantissa/2^n)*2^(exponent+offset)
>
> where 'n' is the number of bits in the mantissa, and offset is negative.
> Note that this formula provides no way to represent 0 (the mantissa is
> never negative). As a special exception, a mantissa and exponent that
> are both zero are treated as representing 0, rather than 2^offset, which
> is what the general formula would call for.
>
> Thus, for any given value of 'k' within a certain range, this format can
> represent exactly 2^n different value x in the range 2^k <= x < 2^k+1,
> evenly spaced within that interval.

Nuno, aren't you glad you asked. :-)

This kind of answer has always fallen into the
"Too Much Information" category for me. I think of
it this way, you can have fast or accurate, but you
can't have both. That's about as much as I've ever
needed to know using a computer. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/