Subject: Re: Commutativity of multiplication
Posted by JD Smith on Wed, 25 Oct 2006 23:09:03 GMT
View Forum Message <> Reply to Message

On Wed, 25 Oct 2006 14:54:59 -0700, Sven Geier wrote:

> David Fanning wrote:
> [...]
>> It also explains this:
>>
>> IDL> f = 500L
>> IDL> g = 1UL
>> IDL> help, g*(-f)
>> <Expression> ULONG = 4294966796
>> IDL> help, (-f)*g
>
> Wow. This is ... uhm ... "more interesting than I thought". There's whole
> realms of oddity here that I never knew existed:
>
> IDL> help,f,g
> F LONG = -500
> G ULONG = 1
> IDL> print,f*g
> -500
> IDL> print,g*f
> 4294966796
> IDL> if f*g eq g*f then print,f*g," equals ",g*f
> -500 equals 4294966796

The bit pattern for these two numbers is *exactly* the same. The only difference is the "type" that IDL assigns them, which affects how the numbers are printed (and only how they are printed). That type (for IDL) is determined by the "left-most" variable or constant in the expression, with the exception that if anything to the right has a larger range, it will be used as the type (e.g. promotion to float, double, ULONG64, etc.), hence:

IDL> f=1.0
IDL> l=100UL
IDL> help,l+f
<Expression> FLOAT = 101.000

When the range is the same, e.g. UL and L, leftmost always wins.

Commutation hasn't been broken, only "type commutation", which doesn't really exist. For all purposes, given the limitations of integer representation in computers, -500 and 4294966796 *are* the same. I could just as easily claim that "adding and subtracting 1 is broken":

```
IDL> print, 4294967295UL + 1UL
           0

IDL> print,0b - 1b
 255
```

JD
