Subject: Re: Reading 32-bit complex numbers in IDL (16-bit real / 16-bit imaginary)
Posted by Waqas A. Qazi on Mon, 15 Aug 2011 17:58:43 GMT

View Forum Message <> Reply to Message

On Aug 13, 6:26 am, Wox <s...@nomail.com> wrote:

```
>  When I say "convert integer to float" I don't mean that you would
>  convert 10 to 10.0 for example. The value of the integer is not
>  concidered at all, it's the "bit-content" that is used. The integer is
>  just a bag of bits that represent a foating point number according to
>  the IEEE754 standard. That's why the function is call BINARYTOFLOAT
>  and not INTEGERTOFLOAT or something. In the example I gave:
>
>  IDL> integer='3555'x
>  IDL> f=binarytofloat(integer,precision=0)
>  IDL> print,integer
>    13653
>  IDL> print,f
>      0.333252
>
>  You can see that the integer value 13653 has nothing to do with
>  0.33325... However both numbers have the same binary representation,
>  namely
>
>  IDL> print,integer,format='(b016)'
>  0011010101010101
>
>  Note: Since IDL can't handle 16bit floats, the binary representation
>  of f (32bit float) isn't the same anymore as that of 13653. The value
>  of f is correct however: the value represented by 13653 under the
>  half-precision IEEE 754 convention.
```

Great, thanks Wox, I understand now.

For the specific data I am working with, its not half-precision per se, because the 2-byte binary numbers are defined as integers, not floats. So after reading the 2-byte integers from the binary file, I will not need to convert to any float. I can read each alternating 2-byte integer into separate real and imaginary part arrays, and then input these two arrays directly into the complex function (data type converter) in IDL.

Thanks,
Waqas.