Subject: Bug? yea or nay.
Posted by Vapuser on Thu, 15 Feb 2001 19:42:22 GMT
View Forum Message <> Reply to Message

I greatly simplified the structure in question to elucidate the point.

```
IDL> print,!version
{ mipseb IRIX unix 5.3 Nov 11 1999}

IDL> r={a:0.d, c:0L} & print,n_tags(r,/length),totsize(r)
16          12
```

Totsize is routine I wrote to calculate the size of things, primarily structures, by recursing through the thing and adding up the sizes of the various components. Lots of the structures I use here at work are not laid out very well, so I had to come up with some better method than hand counting to determine the size of structures.

```
IDL> help,r,/st
** Structure <1007e308>, 2 tags, length=16, refs=1:
   A               DOUBLE          3.1415927
   C               LONG              0
```

My understanding of the way C padded structures was that you could always avoid the problem of padding if you started with the largest quantities first and then worked your way down to the smallest. And that's the way the structure that alerted me to this oddity was laid out, first doubles, then longs and floats.

Does it also pad the structure so that it's an integral number of whatever is the largest item? That's what seems to be happening.

The strange thing is that the real structure, for which n_tags was reporting the wrong size, was then used to correctly read data from a binary file. It didn't read all the data, of course, because the calculation of the number of records in the file was incorrect due to the incorrect record size reported by n_tags.

And we verified, using fstat, that the read had read nrecs*totsize(r) bytes of data from the file, so each record was being filled with totsize(r) bytes and all the quantities looked correct. If it had been reading and filling each record with n_tags(r) bytes, there should have been a problem starting with the second record, since it would've been off by n_tags(r)-totsize(r) bytes.

So the low level I/O routines were correctly filling the
structure, treating each as a structure of totsize(r) bytes, when
n_tags was returning the size of some padded variant.

As you can see:

```
IDL> rr=replicate(r,3)
IDL> rr.a=!pi*(findgen(3)+1)
IDL> rr.c=findgen(3)
IDL> openw,1,
IDL> writeu,1,rr
IDL> close,1

% ls -al /tmp/n_tags.test
-rw-r--r--    1 vapuser  vapusers      36 Feb 15 11:28 /tmp/n_tags.test


IDL> openr,1,'/tmp/n_tags.test'
IDL> rr.a=0
IDL> rr.c=-1
IDL> rr.c=-1
IDL> print,rr
{       0.0000000         -1}{       0.0000000         -1}{       0.0000000
         -1}
IDL> readu,1,rr
IDL> print,rr
{       3.1415927          0}{       6.2831855          1}{       9.4247780
          2}
```

What say you? Bug or not?

whd
--
William Daffer: 818-354-0161: William.Daffer@jpl.nasa.gov