Subject: Re: read a C written binary file with IDL
Posted by Paul Van Delst[1] on Fri, 10 Feb 2012 14:09:14 GMT
View Forum Message <> Reply to Message

Have you tried ASSOC? It has a PACKED keyword you can play with:

<quote>
PACKED
When ASSOC is applied to structures, the default action is to map the actual definition of the structure for the current
machine, including any holes required to properly align the fields. (IDL uses the same rules for laying out structures
as the C language). If the PACKED keyword is specified, I/O using the resulting variable instead works in the same
manner as READU and WRITEU, and data is moved one field at a time and there are no alignment gaps between the fields.
</quote>

So you might want to experiment with ASSOC and /PACKED, e.g.

```
mystruct={a:0L, b:0LL, c:0L,d:0.0}
openr, 1, filename, /swap_if_big_endian
a = assoc(1, {a:0L, b:0LL, c:0L,d:0.0})
  or
a = assoc(1, {a:0L, b:0LL, c:0L,d:0.0},/packed)
help, a[0]
```

cheers,

paulv

bing999 wrote:
> Hi,
>
> I am having a problem with reading a C written binary file with IDL.
> It may come from differences of type definitions between C and IDL but
> I could not really figure out from Google...
>
> In C, it writes a structure containing the following variable types:
>
> struct MyStruct
> {
> int a;
> long long b;
> int c;
> float d;
> };

>
> Then, in IDL, I read this with:
>
> MyStruct = {$
> a : 0L, $
> b : 0LL, $
> c : 0L, $
> d : 0.0 $
> }
>
> openr, 1, filename, /SWAP_IF_BIG_ENDIAN
> readu, 1, MyStruct
> close, 1
>
>
> but this gives me wrong values.
>
> Did I miss something about the type conversion??
>
> If someone could please clarify this, it would really help!
> Thanks !
>
>
>
>
>