Subject: Re: Scalar passing with call_external
Posted by Allan Whiteford on Tue, 25 Mar 2008 09:30:18 GMT
View Forum Message <> Reply to Message

Brian Larsen wrote:
> All,
>
> I am having a heck of a time with something here. I am trying to
> understand the behavior of a C routine that I am trying to call from
> IDL.
>
> Given this C code: (ctest.c)
> void ctest (float a, float b, float *c, float v[3])
> {
> float tmp;
> tmp = a*b;
> v[0] = tmp;
> v[1] = a-b;
> v[2] = a+b;
> *c = 66;
> }
>
> And this IDL code: (ctest.pro)
> make_dll, 'ctest', 'ctest', 'ctest', compile_directory = '.'
> a = 5.
> b = 3.
> c = [7.]
> v = fltarr(3)
> ans = call_external('ctest.so', 'ctest', a, b, c, v, /auto_glue, value
> = [1, 1, 0, 0])
> print, ans, a, b, c, v
> end
>
> I expect the output to be:
> <undetermined> 5.0 3.0 66.0
> 15.0 2.0 8.0
>
> But instead I get:
> IDL> .run ctest
> -1073774376 5.00000 3.00000 7.00000
> 15.0000 2.00000 8.00000
>
> So the root of the question is how do I pass scalar information back
> out of the C routine to the IDL? As that looks to be failing. Anyone
> have any useful tips here?
>
> Cheers,
>

> Brian
>
> ---------- ----------
> Brian Larsen
> Boston University
> Center for Space Physics

Brian,

Nothing useful besides a fairly unhelpful "it worked for me"...

```
IDL> .r ctest
% Compiled module: $MAIN$.
   1115947008      5.00000      3.00000      66.0000
      15.0000      2.00000      8.00000
IDL> print,!version
{ x86 linux unix linux 6.2 Jun 20 2005      32      64}
```

with identical results on each of:

```
{ x86_64 linux unix linux 6.3 Mar 23 2006      64      64}
{ x86 linux unix linux 7.0 Oct 25 2007      32      64}
{ sparc sunos unix Solaris 6.2 Jun 20 2005      32      64}
{ sparc sunos unix Solaris 6.4 Mar 24 2007      64      64}
{ sparc sunos unix Solaris 7.0 Oct 25 2007      64      64}
```

which were chosen to be sufficiently random across architecture, version number and 32/64 bit. It could be a specific compiler or architecture thing I guess. If you happen to be using one of the above then I'd check compilers rather than IDL.

It might also me that your compiler isn't overwriting the intermediate shared object file, have you tried explicitly deleting it and running the test again? Note also that usually you need to exit IDL and re-run it to pick up any changes in a shared object file (unless you specify /unload to call_external but that might be implicit when you use auto_glue).

Sorry if you've tried everything in the above paragraph, it's all I can think of though.

Thanks,

Allan