

---

Subject: Re: CALL\_EXTERNAL Problems with IDL  
Posted by [Bob\[2\]](#) on Wed, 04 Aug 2004 11:35:07 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Peter, I am not sure where that -7 comes from, but it does show you are talented. ;) A few comments on your query.

You want to make a "shared library" for call\_external. Shared libraries come up with different names on different platforms, a vocabulary nuisance. On Linux, there are some compile flags to set. Take a look at your documentation. Also,

<http://www.fortran-2000.com/ArnaudRecipes/sharedlib.html>

is handy for sorting out terminology. I use gcc to compile my source files and than I usually use ld to build a shared library.

Another thing you might want to do is include the IDL header file, idl\_export.h, which is somewhere in your IDL files. E.g., in one of my C programs, I have

```
#include "/Applications/idl_6.0/external/include/idl_export.h"
```

So your 'int' type has to be switched to 'IDL\_INT' -- though 'double' is OK as is. I have noticed that on my system (OS-X, Darwin), I have to leave the main() function as type 'int' or the compiler chokes. There is probably a flag to set in gcc that will allow 'IDL\_INT main( void )' but I have not dug it out.

Third: one thing I noticed a while ago is that when you are running idlde and testing your .so shareable object, you have to reset the IDL session after you modify and recompile the .so because IDL still has the previous .so loaded. Just a guess, but this might be where that sweet value of -7 comes from. There are some postings on this reset trick in the newsgroup a few months ago.

Stay working on it, and reading the Linux documentation.

- Thomas

PeterOut wrote:

```
> I am using IDL Version 6.0 (linux x86 m32) on Red Hat Linux release 9.  
>  
> I have been trying to get CALL_EXTERNAL to do *something* for me so I  
> wrote some very simple code, Generic.c, which follows. I also define  
> the functions in Generic.h.  
> -----
```

```
> #include <stdio.h>
> #include <stdlib.h>
> #include "Generic.h"
>
> int Simple()
> {
>     return 16;
> }
>
> int PowerOf2()
> {
>     int output;
>     output=32;
>     return output;
> }
> -----
> I then compile this code with
>
>> gcc -c Generic.c -o Generic.o
>
> and confirm that the functions are there thus.
>
>> nm Generic.o
>
> 0000000a T PowerOf2
> 00000000 T Simple
>
> This is what I now get on IDL.
> IDL> print,call_external('/home/me/Generic.o','Simple',/i_value)
> % CALL_EXTERNAL: Error loading sharable executable.
>     Symbol: Simple, File = /home/me/Generic.o
>     /usr/local/rsi/idl_6.0/bin/bin.linux.x86/libidl.so.6.0:
>     undefined symbol: Simple
>
> IDL> print,call_external('/home/me/Generic.o','PowerOf2',/i_value )
>     0
> IDL> print,call_external('/home/me/Generic.o','PowerOf2',/i_value )
>     -7
>
> OK. So how is Simple undefined?
> And how does it get 0 and then 7 from 32?
>
> Many thanks in advance for any help,
> Peter.
```

---