Subject: Building sharable object libraries for CALL_EXTERNAL
Posted by Octavi Fors on Fri, 12 Mar 1999 08:00:00 GMT
View Forum Message <> Reply to Message

<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
<html>
Hello,
<p>This is my first trial calling a C-routine from IDL with CALL_EXTERNAL,
and I have some problems with building routine.so file. It's just
a test, so the routine it's very simple:
<p>#include <math.h>

#include <stdio.h>

#include <stdlib.h>

#include <fcntl.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <string.h>
<p>float** routine(argc,argv)

int argc;

void *argv[];

{

int n, x, y;

float *input;

float **output;

int i,j;
<p> input = (float *) argv[0];

 n=*(int *) argv[1];

 x=*(int *) argv[2];

 y=*(int *) argv[3];
<p> output=(float **)malloc(4*n);

 for(i=0;i<n;i++) output[i]=(float*)malloc(4*x*y);
<p>for(j=0;j<n;++j)

for(i=0;i<x*y;++i)

output[j][i]=input[i];
<p>return(output);

}
<p>First, I try to create routine.o with:

$ cc -pic -fsingle -c routine.c

no problem.
<p>Second, when I try to create routine.so with:

$ ld -o routine.so -z text routine.o (changing '-assert
pure-text' to '-z text' because it's a SunOS Version 5.5.1)

 Undefined   &
amp;nbsp;   & amp;nbsp;   &
amp;nbsp;
first referenced

 symbol   &amp

;nbsp;   &amp ;nbsp;   &amp
;nbsp;   &amp ;nbsp;
in file

 malloc   &amp
;nbsp;   &amp ;nbsp;   &amp
;nbsp;   &amp ;nbsp;
routine.o (symbol belongs to implicit dependency /usr/lib/libc.so.1)

 .mul   &n
bsp;   &n bsp;   &n
bsp;   &n bsp;   &n bsp;
routine.o (symbol belongs to implicit dependency /usr/lib/libc.so.1)

 mcount   &amp
;nbsp;   &amp ;nbsp;   &amp
;nbsp;   &amp ;nbsp;
routine.o

ld: fatal: Symbol referencing errors. No output written
to routine.so
<p>It seems to need /usr/lib/libc.so.1. So I add it in the command line:

$ ld -o routine.so -z text routine.o /usr/lib/libc.so.1

 Undefined   &
amp;nbsp;   & amp;nbsp;   &
amp;nbsp;
first referenced

 symbol   &amp
;nbsp;   &amp ;nbsp;   &amp
;nbsp;   &amp ;nbsp;
in file

 mcount   &amp
;nbsp;   &amp ;nbsp;   &amp
;nbsp;   &amp ;nbsp;
routine.o

ld: fatal: Symbol referencing errors. No output written
to routine.so
<p>Malloc error is gone, but mcount (?) remains. Does anybody know what's
wrong with it?
<p>Thans in advance,
<p>Octavi.
<pre>--
 == =====

Octavi Fors Aldrich

Astronomy Department</pre>

<pre>Physics Faculty</pre>

<pre>Avgda. Diagonal 647
08028 Barcelona

SPAIN

Telf: 34-934021122
Fax: 34-934021133
e-mail: octavi@fajnm1.am.ub.es

========== ===== </pre>
 </html>

Subject: Re: Building sharable object libraries for CALL_EXTERNAL
Posted by Octavi Fors on Sat, 13 Mar 1999 08:00:00 GMT
View Forum Message <> Reply to Message

<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
<html>
"Eric J. Korpela" wrote:
<blockquote TYPE=CITE>In article <36E9649A.7F5C8B65@fajnm1.am.ub.es>,

Octavi Fors <octavi@fajnm1.am.ub.es> wrote:

>

>Hello,

>

>This is my first trial calling a C-routine from IDL with CALL_EXTERNAL,

>and I have some problems with building routine.so file. It's just
a

>test, so the routine it's very simple:

>

>Second, when I try to create routine.so with:

>$ ld -o routine.so -z text routine.o (changing '-assert pure-text'
to

>'-z text' because it's a SunOS Version 5.5.1)
<p>I think you need to specify the -G flag to ld when creating a shared

object file under Solaris.</blockquote>
I tried your advice with -G flag:
<p>$ld -o routine.so -z text -G routine.o
Text relocation remains   &amp
p;nbsp;   & p;nbsp;   &amp
p;nbsp;
referenced
    against symbol   &amp
;nbsp;   &amp ;nbsp;
offset   in file
<unknown>

 0x1c   &n bsp; routine.o
<unknown>

0x20   &n bsp; routine.o
.mul   &n bsp;   &n
bsp;   &n bsp;   &n
bsp;   &n bsp;
0xb0   &n bsp; routine.o
.mul   &n bsp;   &n
bsp;   &n bsp;   &n
bsp;   &n bsp;
0x120   & nbsp; routine.o
mcount   &amp ;nbsp;   &amp
;nbsp;   &amp ;nbsp;   &amp
;nbsp;
0x28   &n bsp; routine.o
ld: fatal: relocations remain against allocatable but non-writable sections

but it also seems to fail. The error message seems to indicate some
problem with

relocations. What do you think?

Taking a look at the routine.c code (it's in the first message), I
noticed that the

**pointer (output) I'm passing back to IDL is malloced in the C-routine,
in line:
<p>output=(float **)malloc(4*n);

 for(i=0;i<n;i++) output[i]=(float*)malloc(4*x*y);
<p>Is there something wrong with this?

I tried not to malloc output inside routine.c (commenting previous
lines), but last

error messages still remains.
<p>Any ideas are welcome.
<p>Octavi.
<p>--
========== =====

Octavi Fors Aldrich

Astronomy Department

Physics Faculty

Avgda. Diagonal 647
08028 Barcelona
SPAIN

Telf: 34-934021122
Fax: 34-934021133
e-mail: octavi@fajnm1.am.ub.es

========== =====

 </html>

Subject: Re: Building sharable object libraries for CALL_EXTERNAL
Posted by korpela on Mon, 15 Mar 1999 08:00:00 GMT
View Forum Message <> Reply to Message

In article <36EAAE98.8F975873@fajnm1.am.ub.es>,
Octavi Fors <octavi@fajnm1.am.ub.es> wrote:
>
> I tried your advice with -G flag:
>
> $ld -o routine.so -z text -G routine.o
> Text relocation remains   referenced
>   against symbol   offset   in file
> <unknown>   0x1c   routine.o
> <unknown>   0x20   routine.o
> .mul   0xb0   routine.o
> .mul   0x120   routine.o
> mcount   0x28   routine.o
> ld: fatal: relocations remain against allocatable but non-writable sections

Hmmmm. I just tried your routine with gcc, (we don't have Sun's compilers)
% uname -a
SunOS sagan 5.6 Generic_105181-09 sun4u sparc SUNW,Ultra-5_10
% gcc -fPIC -c routine.c
% /usr/ucb/ld -o t1.so -z text -G routine.o

Under gcc, -fPIC and -fpic generate different code. Could that be the
problem?

Eric
--
Eric Korpela | An object at rest can never be
korpela@ssl.berkeley.edu | stopped.
Click for home page.

Subject: Re: Building sharable object libraries for CALL_EXTERNAL
Posted by rmlongfield on Wed, 17 Mar 1999 08:00:00 GMT
View Forum Message <> Reply to Message

In article <36E9649A.7F5C8B65@fajnm1.am.ub.es>,
  Octavi Fors <octavi@fajnm1.am.ub.es> wrote:
> Hello,
>
> This is my first trial calling a C-routine from IDL with CALL_EXTERNAL,

> and I have some problems with building routine.so file. It's just a
> test, so the routine it's very simple:

Hello Octavi, I am a few days late, but do not know if you have solved your problem. I work on an SGI and have the following Makefile,where grouptest is your example program. I recall that it took a lot of testing and searching in the SGI man pages to find the right flags. It was a lot simpler than the sample programs provided by IDL. ---------- LDFLAGS= -lm

```
OBJS =  grouptest.o

grouptest : $(OBJS)
 ld -shared -o grouptest.so $(OBJS) $(LDFLAGS)

grouptest.o : grouptest.c
 cc -c -KPIC grouptest.c -o grouptest.o
```

This compiles ok on my SGI. When run from IDL, the only problem is with the returned value of output.

```
--IDL sample file ---->
pro grouptest
a = FLOAT(1.4)
b= LONG(2)
c = LONG(7)
d = LONG(9)
output =  CALL_EXTERNAL('grouptest.so','grouptest',a,b,c,d)
print,'Returned value from group test: ',output
end
-------------------------
```

If you can search back in this newsgroup a couple of months (Dejanews, for example, can be used to search archives) you will find a lot of discussion about CALL_EXTERNAL and pointers. Maybe it is different for a SUN, but I am pretty sure that you cannot pass a pointer with the C return statement. Note also that all memory for variables sent through CALL_EXTERNAL must first be specified in IDL. This may also be a source of error. Another note is that, if you change your C code, you must exit IDL before running again from within IDL.

I use CALL_EXTERNAL with C and Fortran programs in a simple way which works. Please feel free to ask further questions and I will try to help.

Good Luck!
Rose

Subject: Re: Building sharable object libraries for CALL_EXTERNAL
Posted by Octavi Fors on Thu, 18 Mar 1999 08:00:00 GMT
View Forum Message <> Reply to Message

<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
<html>
"Eric J. Korpela" wrote:
<blockquote TYPE=CITE>In article <36EAAE98.8F975873@fajnm1.am.ub.es>,

Octavi Fors <octavi@fajnm1.am.ub.es> wrote:

>

>I tried your advice with -G flag:

>

>$ld -o routine.so -z text -G routine.o

>Text relocation remains   &am
p;nbsp;   &am p;nbsp;   &am
p;nbsp;
referenced

>   against symbol   &amp
;nbsp;   &amp ;nbsp;
offset   in file

><unknown>

0x1c   &n bsp; routine.o

><unknown>

0x20   &n bsp; routine.o

> .mul   &n bsp;   &n
bsp;   &n bsp;   &n
bsp;   &n bsp;
0xb0   &n bsp; routine.o

> .mul   &n bsp;   &n
bsp;   &n bsp;   &n
bsp;   &n bsp;
0x120   & nbsp; routine.o

> mcount   &amp
;nbsp;   &amp ;nbsp;   &amp
;nbsp;   &amp ;nbsp;
0x28   &n bsp; routine.o

>ld: fatal: relocations remain against allocatable but non-writable
sections
<p>Hmmmm. I just tried your routine with gcc, (we don't have Sun's
compilers)

% uname -a

SunOS sagan 5.6 Generic_105181-09 sun4u sparc SUNW,Ultra-5_10

% gcc -fPIC -c routine.c

% /usr/ucb/ld -o t1.so -z text -G routine.o
<p>Under gcc, -fPIC and -fpic generate different code. Could that
be the

problem?
<p>Eric

--

Eric Korpela   &am
p;nbsp;   &am p;nbsp;   &am
p;nbsp;
| An object at rest can never be

korpela@ssl.berkeley.edu

| stopped.

<a href="<a
href="http://sag-www.ssl.berkeley.edu/~korpela">http://sag-www.ssl.berkeley.edu/~korpela">
Click
for home page.</blockquote>
Hello!
<p>Talking with my system manager, he told me that the right compilation
option is SunSolaris instead SusOs (in IDL Users Guide 4.0 they are considered
separately). In that way, I proceed with the SunSolaris options, described
in the IDL Users Guide:
<p>$ uname -a

SunOS mizar 5.5.1 Generic_103640-05 sun4u sparc
SUNW,Ultra-Enterprise
<p>$ cc -G -Kpic -c routine.c

$ ld -G -o routine.so routine.o
<p>and no problem!
<p>Thanks all!
<pre>--
 ========== =====

Octavi Fors Aldrich

Departament d'Astronomia i Meteorologia
Facultat de Fisica
Avgda. Diagonal 647
08028 Barcelona
SPAIN

Telf: 34-934021122
Fax: 34-934021133
e-mail: octavi@fajnm1.am.ub.es

 ========== ===== </pre>

 </html>

Subject: Re: Building sharable object libraries for CALL_EXTERNAL
Posted by Dr. G. Scott Lett on Mon, 22 Mar 1999 08:00:00 GMT
View Forum Message <> Reply to Message

I don't think this is true. Functions like IDL_MakeStruct(), IDL_ImportArray() and IDL_ImportNamedArray(), are handy for building IDL variables with memory you allocate.

However, the following code will most likely not produce the desired results. Reading the External Development Guide is a recommended first step, especially the chapter on IDL Internals: Variables.

> output=(float **)malloc(4*n);
> for(i=0;i<n;i++) output[i]=(float*)malloc(4*x*y);

Scott

>
> I thought it was a no-no to allocate memory inside of an external
> routine, that is to be passed back to IDL. The Advanced Developer's
> Guide has always stated that all memory must be allocated prior to
> calling the routine. Is that not always the case?
>
> Even if you get this to compile, I would be very wary of the results.
>
> Dave
> --
>
> ~~~~~~ ~~~~~~
> David S. Foster Univ. of California, San Diego
> Programmer/Analyst Brain Image Analysis Laboratory
> foster@bial1.ucsd.edu Department of Psychiatry
> (619) 622-5892 8950 Via La Jolla Drive, Suite 2240
> La Jolla, CA 92037
> ~~~~~~ ~~~~~~

Subject: Re: Building sharable object libraries for CALL_EXTERNAL
Posted by David Foster on Mon, 22 Mar 1999 08:00:00 GMT
View Forum Message <> Reply to Message

Octavi Fors wrote:

> Taking a look at the routine.c code (it's in the first message), I
> noticed that the
> **pointer (output) I'm passing back to IDL is malloced in the
> C-routine, in line:
>
> output=(float **)malloc(4*n);
> for(i=0;i<n;i++) output[i]=(float*)malloc(4*x*y);
>
> Is there something wrong with this?

I thought it was a no-no to allocate memory inside of an external routine, that is to be passed back to IDL. The Advanced Developer's Guide has always stated that all memory must be allocated prior to calling the routine. Is that not always the case?

Even if you get this to compile, I would be very wary of the results.

Dave
--

~~~~~~ ~~~~~~~
David S. Foster Univ. of California, San Diego
Programmer/Analyst Brain Image Analysis Laboratory
foster@bial1.ucsd.edu Department of Psychiatry
(619) 622-5892 8950 Via La Jolla Drive, Suite 2240
La Jolla, CA 92037
~~~~~~ ~~~~~~~