Subject: Fortran External Calls
Posted by jpgrimes on Thu, 17 Dec 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Hi,
I just need help with a simple (I htink) problem.
Basically I am having a bit of trouble understanding the little
bit of documentation in my idl manuals on call_extern. I just have a f90
function that I want to call from idl. The manuals refer to some advanced
idl manual which I don't seem to have.
I found the faq and they have a very promising site at
eos.crseo.ucsb.edu:/pub/idl/idl-fortran.Z
but their isn't any such file at this site.
Anyway I think all I need is a solid example of it being done.
Don't really understand how to do the argv stuff in fortran. What I am
trying is definetely not working.
Thanks in advance.
John Grimes

--
****
* The good ended happily, and the bad unhappily. That is what *
* Fiction means. *
* - Oscar Wilde *
****
John Grimes - Physics Grad Student at U of Chicago
Home me@johngrimes.com 5400 S. Ingleside Ave Apt #3 (773)363-4869
Physics jp-grimes@uchicago.edu Lab for Astrophysics-Office#207 702-0162
http://student-www.uchicago.edu/users/jpgrimes/

Subject: Re: Fortran External Calls
Posted by rmlongfield on Fri, 18 Dec 1998 08:00:00 GMT
View Forum Message <> Reply to Message

In article <F43Aqt.I3H@midway.uchicago.edu>,
jpgrimes@midway.uchicago.edu (john peter grimes) wrote:
> Hi,
> I just need help with a simple (I htink) problem.
> Basically I am having a bit of trouble understanding the little
> bit of documentation in my idl manuals on call_extern. I just have a f90
> function that I want to call from idl.

Hi John Peter:
I found the IDL example files very confusing. Here is
a straightforward sample of a call_external using a C program to call the
Fortran program.
1) IDL program: idl_ex.pro : Note ALL arrays and variables

defined FIRST in IDL
2) C program: exc.c
3) Fortran program: exf.f
4) make file to compile before running in IDL (Must exit IDL
if you recompile!!!): Make_ex

I have run it on an SGI and IDL 5.0 with Fortran 77. You should be able to just run the make command (make -f Make_ex) and then run idl_ex in IDL. I hope it works with Fortran 90. Please let me know. Good Luck.

Rose

```
 ------------------------------------------------------------ -----------------
-- ---idl_ex.pro
 ------------------------------------------------------------ --

PRO idl_ex
; Define arrays
a_surface = DOUBLE(.19)
sza = DOUBLE(45)
vza= DOUBLE(55)
ra = DOUBLE(65)
nbcloud=14
tau_c = DBLARR(nbcloud)
reflectivity = DBLARR(nbcloud)
print,'before call to external '
result_rtau =  CALL_EXTERNAL('exc.so','exc',a_surface,sza,vza,ra, $
 tau_c,reflectivity)
print,'Returned value from rtau: ',result_rtau
print,'--------------------Result of CALL_EXTERNAL----------------------'
print,'tau ',' reflectivity'
FOR k=0,nbcloud-1 DO BEGIN
print,tau_c[k],reflectivity[k]
ENDFOR

end

---exc.c  ------------------------------------------------------------ --
/* This program serves as a wrapper for the fortran program exf.f */
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

long exc(int argc,void *argv[])
{
extern void exf_();

static int i,j;
```

```
static double tau_c[14],reflectivity[14] ;
static double a_surface, sza_deg, vza_deg, ra_deg;

a_surface = *(double *)argv[0];
sza_deg= *(double *)argv[1];
vza_deg= *(double *)argv[2];
ra_deg= *(double *)argv[3];
/* Check to see if values correspond with IDL input */
fprintf(stderr,"%lf %lf %lf %lf \n",a_surface,sza_deg,vza_deg,ra_deg);
fprintf(stderr," -------------Call fortran program----------------- \n");

  exf_(&a_surface,&sza_deg,&vza_deg,&ra_deg,ta u_c,reflectivity);

fprintf(stderr,"----------- Returned balues from Fortran: ---------\n");
 for (i=0;i<14;++i)
   {
    *((double *)argv[4] + i) = tau_c[i] ;
    *((double *)argv[5] + i) = reflectivity[i] ;
     fprintf(stderr," %lf %lf \n",tau_c[i] ,reflectivity[i]);
       }
/* Send some strange number as a check to IDL */
return(33);
}
---exf.f  ------------------------------------------------------------ --
      subroutine exf(albedo,solzen,satzen,phi,tau_c,
     6    reflectivity)
      integer i
      double precision tau_c(14),reflectivity(14)
      double precision albedo,solzen,satzen,phi
C Check to see if input values passed correctly
      write(*,*)albedo,solzen,satzen,phi
C Calculate new values to be returned to C program and IDL
      do 20 i=1,14
        tau_c(i) = i*2.0
       reflectivity(i) = i/10.
 20   continue
      return
      end

---Make_ex  ------------------------------------------------------------ --

LDFLAGS= -lF77 -lU77 -lI77 -lisam -lm OBJS = exc.o exf.o exc : $(OBJS) ld
-shared -o exc.so $(OBJS) $(LDFLAGS) exc.o : exc.c   cc -c -KPIC exc.c -o
exc.o exf.o : exf.f    f77 -c -KPIC exf.f -o exf.o
 ------------------------------------------------------------ -----------------
--

----------== Posted via Deja News, The Discussion Network ==----------
```

Subject: Re: Fortran
Posted by steinhh on Thu, 08 Apr 1999 07:00:00 GMT
View Forum Message <> Reply to Message

In article <ojXO2.50044$qt5.6315@news.rdc2.occa.home.com>
"H T Onishi" <htonishi@home.com> writes:

> I'm just starting a project that requires developing a GUI for
> a large FORTRAN code. Work will be done on a PC. We are
> considering IDL, however, there is a "Just Say No" warning in
> one of the IDL reference books. Developing a C wrapper is
> formidable because the FORTRAN code is formidable. Has anyone
> successfully linked IDL and FORTRAN? Any suggestions (such as
> avoid at all costs)? Any comparisons with Visual Basic?

I don't see why writing C wrappers should be such a big job
just because the FORTRAN code is large -- it just depends on
how many "entry points" you need to have from your GUI to your
FORTRAN code. If you do the type checking etc in IDL, writing C
wrappers can be done almost automatically. (And using the
utilities available in callable IDL, doing the type
checking/conversion in C is almost as easy).

True, there is a bit of a threshold when you first start to
make call_external/linkimage/dlm code, but if it's a big
project, the initial time spent learning the game will be small
compared to the rest of the project.

Anyway, I would rate the job of writig C wrappers as
significantly smaller than the job of writing the GUI in the
first place (although this does not take into account writing
the FORTRAN "entry point" routines, if they need to be written,
and are complex).

I don't have any experience with Visual Basic, but if your GUI
is going to display data from the FORTRAN code, I'd say you're
probably better off with IDL - I don't believe VB could have
any large benefits over IDL wrt. ease of use for writing GUIs
that could possibly outweigh IDL's data display capability..

Of course, this all depends on your (and other programmers')
previous experience with VB versus IDL, as well.

And yes - I have linked in 3rd party FORTRAN routines as DLMs
by writing C wrappers. FORTRAN routines for calculating angular

size distances from redshifts given arbitrary cosmological parameters is now a seamless part of IDL here (just like system functions).

Best of luck

Stein Vidar

Subject: Re: Fortran
Posted by Michael Asten on Fri, 09 Apr 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Dont know where the 'just say no' reference comes from - my idl course notes of a couple of years ago (thanks David) list at least 4 ways of doing the link, starting with the ultra-simple approach of using the idl to setup input for a fortran executable, where the said fortran is initiated by the idl code with a system call, and reads its input from the file generated by the idl code.
I have used this ultra-simple way quite sucessfully, since my fortran and idl codes do not have to interact, and my concern has been to get the results, not write code. How sophicated you need to be (and hence how much you use c-wrappers, dlls, and the like) will perhaps depend on how much interaction you need with your core fortran routines.

Regards,
Michael Asten

H T Onishi wrote:

> I'm just starting a project that requires developing a GUI for a large
> FORTRAN code. Work will be done on a PC. We are considering IDL, however,
> there is a "Just Say No" warning in one of the IDL reference books.
> Developing a C wrapper is formidable because the FORTRAN code is formidable.
> Has anyone successfully linked IDL and FORTRAN? Any suggestions (such as
> avoid at all costs)? Any comparisons with Visual Basic?
>
> Thanks,
> Howard Onishi

Subject: Re: Fortran
Posted by mallors on Sun, 11 Apr 1999 07:00:00 GMT
View Forum Message <> Reply to Message

In article <ojXO2.50044$qt5.6315@news.rdc2.occa.home.com>,
"H T Onishi" <htonishi@home.com> writes:

> I'm just starting a project that requires developing
> a GUI for a large
> FORTRAN code. Work will be done on a PC. We are
> considering IDL, however,
> there is a "Just Say No" warning in one of the IDL
> reference books.
> Developing a C wrapper is formidable because the
> FORTRAN code is formidable.
> Has anyone successfully linked IDL and FORTRAN?
> Any suggestions (such as
> avoid at all costs)? Any comparisons with Visual Basic?

Several years ago, I wasted countless hours trying
to get FORTRAN and IDL to work together via CALL_EXTERNAL.
The problem, I eventually found out, that you
can (safely) do NO I/O in the FORTRAN code. FORTRAN
I/O through CALL_EXT will seem to work Ok on some
platforms, but will break IDL on others.
This warning is now added to the IDL
documentation - do not ignore it! If your FORTRAN has
I/O scattered throughout, you will have a
rough time.

If your GUI does not have to be extremely complex,
you can write it directly in FORTRAN. Check out

http://www.cs.ubishops.ca/ljensen/fortran/dislin/dislin.html

Regards,

-bob mallozzi

--
~~~~~~ ~~~~~~~~~~
Robert S. Mallozzi 256-544-0887
Mail Code ES 84
Work: http://www.batse.msfc.nasa.gov/ Marshall Space Flight Center
Play: http://cspar.uah.edu/~mallozzir/ Huntsville, AL 35812
~~~~~~ ~~~~~~~~~~

Subject: Re: Fortran
Posted by mallors on Mon, 12 Apr 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Yes, Paul is correct, I should have been more
specific. It's terminal input that seems to
cause problems.

I also assumed the original poster did not
want to edit the FORTRAN, as was my case.

Regards,

-bob

In article <3711EC13.7EF73D55@sandia.gov>,
>
> I have been using an IDL / Fortran interface for over 10 years and had
> no problem with file I/O.
> I agree that using terminal I/O is to be avoided. If you have a big
> Fortran library for file I/O, use it with no problems. For terminal
> input, I have no suggestions other than returning to IDL. For terminal
> output the IDL routine IDL_Message can be easily linked to your
> Fortran.   A sample program which has
> been used with Sun, HP, SGI, DEC, MAC and WIN is below.
>
> Paul Mix, lpmix@sandia.gov
>
> *deck idl_write
>   subroutine idl_write(INPUT, FLAG)
> cDEC$ ALIAS IDL_Message, '_IDL_Message'
> cDEC$ ATTRIBUTES C :: IDL_Message
> c
> c   This routine writes a string to an IDL display
> c
> c   If flag is zero, no leading characters will be printed.
> c   If flag is not zero, then the value of the IDL system variable
> c   !MSG_PREFIX will be printed. (Default value = '% ')
> c
>   CHARACTER *(*) INPUT
>   CHARACTER *1 CHAR
>   INTEGER FLAG
>   INTEGER I, LEN, DONE
>   INTEGER MAXLEN
>   PARAMETER (MAXLEN=512)
>   CHARACTER*(MAXLEN) TEMP
> C   Some Fortran compilers require external definitions for IDL routines
>
>   EXTERNAL IDL_Message !$pragma C(IDL_Message)

```
>  C     check the length and set the last character to a char(0)
>        I = LEN(INPUT)
>        DONE = 0
>        DO WHILE (I .GT. 0 .AND. DONE .EQ. 0)
>          IF (ICHAR(INPUT(I:I)) .GE. 33 .AND.
>       &    ICHAR(INPUT(I:I)) .LE. 126) THEN
>            DONE = 1
>          ELSE
>            I = I-1
>          ENDIF
>        ENDDO
>
>        IF (I .GT. 0) THEN
>          IF (I .GE. MAXLEN) I = MAXLEN-1
>          TEMP = INPUT(1:I)//CHAR(0)
>        ELSE
>          TEMP = ' '//CHAR(0)
>        ENDIF
>        IF (FLAG .EQ. 0) THEN
>          CALL IDL_Message(%VAL(-1), %VAL(262144), %REF(TEMP))
>        ELSE
>          CALL IDL_Message(%VAL(-1), %VAL(0), %REF(TEMP))
>        ENDIF
>  c  ********************************************************** **********
>
>        RETURN
>        END
>
>
```

--

~~~~~~ ~~~~~~~~~~
Robert S. Mallozzi 256-544-0887
Mail Code ES 84
Work: http://www.batse.msfc.nasa.gov/ Marshall Space Flight Center
Play: http://cspar.uah.edu/~mallozzir/ Huntsville, AL 35812
~~~~~~ ~~~~~~~~~~

Subject: Re: Fortran
Posted by L. Paul Mix on Mon, 12 Apr 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Robert S. Mallozzi wrote:

> In article <ojXO2.50044$qt5.6315@news.rdc2.occa.home.com>,
> "H T Onishi" <htonishi@home.com> writes:
> <deleted>j

>
> Several years ago, I wasted countless hours trying
> to get FORTRAN and IDL to work together via CALL_EXTERNAL.
> The problem, I eventually found out, that you
> can (safely) do NO I/O in the FORTRAN code. FORTRAN
> I/O through CALL_EXT will seem to work Ok on some
> platforms, but will break IDL on others.
> This warning is now added to the IDL
> documentation - do not ignore it! If your FORTRAN has
> I/O scattered throughout, you will have a
> rough time.
>
> If your GUI does not have to be extremely complex,
> you can write it directly in FORTRAN. Check out
>
> http://www.cs.ubishops.ca/ljensen/fortran/dislin/dislin.html
>
> Regards,
>
> -bob mallozzi
>
> --
> ~~~~~~ ~~~~~~~~~
> Robert S. Mallozzi   256-544-0887
>   Mail Code ES 84
> Work: http://www.batse.msfc.nasa.gov/   Marshall Space Flight Center
> Play: http://cspar.uah.edu/~mallozzir/   Huntsville, AL 35812
> ~~~~~~ ~~~~~~~~~

I have been using an IDL / Fortran interface for over 10 years and had
no problem with file I/O.
I agree that using terminal I/O is to be avoided. If you have a big
Fortran library for file I/O, use it with no problems. For terminal
input, I have no suggestions other than returning to IDL. For terminal
output the IDL routine IDL_Message can be easily linked to your
Fortran.   A sample program which has
been used with Sun, HP, SGI, DEC, MAC and WIN is below.

Paul Mix, lpmix@sandia.gov

```
*deck idl_write
     subroutine idl_write(INPUT, FLAG)
cDEC$ ALIAS IDL_Message, '_IDL_Message'
cDEC$ ATTRIBUTES C :: IDL_Message
c
c     This routine writes a string to an IDL display
c
c     If flag is zero, no leading characters will be printed.
```

```
c       If flag is not zero, then the value of the IDL system variable
c         !MSG_PREFIX will be printed. (Default value = '% ')
c
      CHARACTER *(*) INPUT
      CHARACTER *1 CHAR
      INTEGER FLAG
      INTEGER I, LEN, DONE
      INTEGER MAXLEN
      PARAMETER (MAXLEN=512)
      CHARACTER*(MAXLEN) TEMP
C   Some Fortran compilers require external definitions for IDL routines

      EXTERNAL IDL_Message !$pragma C(IDL_Message)
C     check the length and set the last character to a char(0)
      I = LEN(INPUT)
      DONE = 0
      DO WHILE (I .GT. 0 .AND. DONE .EQ. 0)
        IF (ICHAR(INPUT(I:I)) .GE. 33 .AND.
     &    ICHAR(INPUT(I:I)) .LE. 126) THEN
          DONE = 1
        ELSE
          I = I-1
        ENDIF
      ENDDO

      IF (I .GT. 0) THEN
        IF (I .GE. MAXLEN) I = MAXLEN-1
        TEMP = INPUT(1:I)//CHAR(0)
      ELSE
        TEMP = ' '//CHAR(0)
      ENDIF
      IF (FLAG .EQ. 0) THEN
        CALL IDL_Message(%VAL(-1), %VAL(262144), %REF(TEMP))
      ELSE
        CALL IDL_Message(%VAL(-1), %VAL(0), %REF(TEMP))
      ENDIF
c  ************************************************************ **********

      RETURN
      END
```