Subject: Re: Read & write data files b/w IDL & Fortran 90
Posted by Paul Van Delst[1] on Thu, 08 Apr 2004 21:45:16 GMT

View Forum Message <> Reply to Message

bridgemat wrote:
> As someone who knows just the basics of IDL and the basics plus a bit
> more of Fortran 90, I'm sure this is just some rookie mistake, so
> hopefully someone out there can help me out.
>
> I'm trying to write an idl array to a file, read the file in Fortran
> 90, manipulate the Fortran array, and then write that new array to a
> file that I then read into IDL. Here's how I write the original array
> to a file:
>
> test=indgen(3,4,5)
> openw,lun,'testidl.dat',/get_lun
> writeu,lun,test
> free_lun,lun

Hello,

You need to use the /f77_unformatted keyword on the IDL open. This will stick the 4-byte delimiter at the beginning and end of each output record. Alternatively, you can specify direct access with a set record length in your f90 OPEN statement.

> Another issue: array dimensions! My IDL book tells me that both IDL
> and Fortran arrays are column-major. Although I've been using Fortran
> longer than IDL, I haven't done much w/ multi-dimensional arrays, so I
> looked in my Fortran book, and it says that Fortran is row-major!
> Ack!!! So does that mean my Fortran array should be declared as
> DIMENSION(5,4,3) in this case? I tried that, too, but got the same
> error...

No. Declare the IDL and Fortran arrays the same way. I can never remember what the hell column- or row-major means. I just remember that IDL is same as Fortran, and C is the opposite.

paulv

Subject: Re: Read & write data files b/w IDL & Fortran 90
Posted by wmconnolley on Thu, 08 Apr 2004 22:00:06 GMT

View Forum Message <> Reply to Message

bridgemat <bridgemat@yahoo.com> wrote:
> test=indgen(3,4,5)
> openw,lun,'testidl.dat',/get_lun
> writeu,lun,test

> free_lun,lun

Isn't there a /f77 keyword? Fortran is a bit weird: it adds the record
length at the start (and end) of each record; IDL needs to know you
want to write a fortran-style file.

-W.

--
William M Connolley | wmc@bas.ac.uk | http://www.antarctica.ac.uk/met/wmc/
Climate Modeller, British Antarctic Survey | Disclaimer: I speak for myself
I'm a .signature virus! copy me into your .signature file & help me spread!

Subject: Re: Read & write data files b/w IDL & Fortran 90
Posted by bridgemat on Fri, 09 Apr 2004 19:35:54 GMT
View Forum Message <> Reply to Message

I did try the /f77_unformatted keyword when writing the original file,
but I still couldn't get it to work. It shouldn't have anything to do
w/ using Fortran 90 instead of 77, right? Or am I doing something
altogether wrong in my Fortran program?

-B

Subject: Re: Read & write data files b/w IDL & Fortran 90
Posted by bridgemat on Fri, 09 Apr 2004 20:20:49 GMT
View Forum Message <> Reply to Message

Also, I did try doing it the opposite way - creating an unformatted
file in F90 and reading into IDL using the /f77_unformatted keyword.
It almost worked. The IDL array was filled with half of the F90 array
elements, having a 0 assigned to each element in between, like this:

```
1   0   2
0   3   0
4   0   5
0   6   0

7   0   8
0   9   0
10  0  11
0  12   0
```

Could this have to do with using F90 instead of F77?

-Bridget

Subject: Re: Read & write data files b/w IDL & Fortran 90
Posted by Paul Van Delst[1] on Fri, 09 Apr 2004 20:37:27 GMT
View Forum Message <> Reply to Message

bridgemat wrote:

> I did try the /f77_unformatted keyword when writing the original file,
> but I still couldn't get it to work. It shouldn't have anything to do
> w/ using Fortran 90 instead of 77, right? Or am I doing something
> altogether wrong in my Fortran program?

Hello,

I just grabbed both your codes and mucked about and had a "d'oh" episode.

You create the data in IDL using INDGEN == 2 byte ints. In your f90 code you read the data with the default int which is 4 bytes. That's why you're getting the I/O error.

Two options:

1) In IDL create the data like so:

```
   test=LINDGEN(3,4,5)
```

And read it with your current f90 code.

Or

2) Create the data file with your current IDL code and in Fortran90, define your integer arrays like so:

```
   INTEGER, PARAMETER :: ip = SELECTED_INT_KIND(4)
   INTEGER(ip), DIMENSION(3,4,5)::arr_in,arr_out
```

The integer kind parameter from SELECTED_INT_KIND(4) should give you the kind type for a 2-byte integer.

I prefer option (1) coz it's the simplest, but YMMV.

cheers,

paulv

Subject: Re: Read & write data files b/w IDL & Fortran 90
Posted by Paul Van Delst[1] on Fri, 09 Apr 2004 20:44:47 GMT
View Forum Message <> Reply to Message

Paul Van Delst wrote:
> bridgemat wrote:
>
>> I did try the /f77_unformatted keyword when writing the original file,
>> but I still couldn't get it to work. It shouldn't have anything to do
>> w/ using Fortran 90 instead of 77, right? Or am I doing something
>> altogether wrong in my Fortran program?
>
>
> Hello,
>
> I just grabbed both your codes and mucked about and had a "d'oh" episode.
>
> You create the data in IDL using INDGEN == 2 byte ints. In your f90 code
> you read the data with the default int which is 4 bytes. That's why
> you're getting the I/O error.
>
> Two options:
>
> 1) In IDL create the data like so:
>   test=LINDGEN(3,4,5)
>   And read it with your current f90 code.
>
> Or
>
> 2) Create the data file with your current IDL code and in Fortran90,
> define your integer arrays like so:
>   INTEGER, PARAMETER :: ip = SELECTED_INT_KIND(4)
>   INTEGER(ip), DIMENSION(3,4,5)::arr_in,arr_out
>
> The integer kind parameter from SELECTED_INT_KIND(4) should give you the
> kind type for a 2-byte integer.
>
> I prefer option (1) coz it's the simplest, but YMMV.

Apologies for my own followup, but in either option above you'll still need the
/f77_unformatted keyword in the IDL routine.

cheers,

paulv

Subject: Re: Read & write data files b/w IDL & Fortran 90
Posted by bridgemat on Wed, 14 Apr 2004 20:45:47 GMT

View Forum Message <> Reply to Message

Hey, thanks Paul! I'll give it a try. My actual data is all float/real

kind of stuff, so I hopefully wouldn't have this problem there. I was
just trying to see if I could do it with an "easy" case, but I guess
simple isn't always easy! :)

-Bridget

Paul Van Delst <paul.vandelst@noaa.gov> wrote in message
news:<c572r2$tuc$1@news.nems.noaa.gov>...
> Paul Van Delst wrote:
>> bridgemat wrote:
>>
>>> I did try the /f77_unformatted keyword when writing the original file,
>>> but I still couldn't get it to work. It shouldn't have anything to do
>>> w/ using Fortran 90 instead of 77, right? Or am I doing something
>>> altogether wrong in my Fortran program?
>>
>>
>> Hello,
>>
>> I just grabbed both your codes and mucked about and had a "d'oh" episode.
>>
>> You create the data in IDL using INDGEN == 2 byte ints. In your f90 code
>> you read the data with the default int which is 4 bytes. That's why
>> you're getting the I/O error.
>>
>> Two options:
>>
>> 1) In IDL create the data like so:
>>   test=LINDGEN(3,4,5)
>>   And read it with your current f90 code.
>>
>> Or
>>
>> 2) Create the data file with your current IDL code and in Fortran90,
>> define your integer arrays like so:
>>   INTEGER, PARAMETER :: ip = SELECTED_INT_KIND(4)
>>   INTEGER(ip), DIMENSION(3,4,5)::arr_in,arr_out
>>
>> The integer kind parameter from SELECTED_INT_KIND(4) should give you the
>> kind type for a 2-byte integer.
>>
>> I prefer option (1) coz it's the simplest, but YMMV.
>
> Apologies for my own followup, but in either option above you'll still need the
> /f77_unformatted keyword in the IDL routine.
>
> cheers,
>

> paulv

Subject: Re: Read & write data files b/w IDL & Fortran 90
Posted by Paul Van Delst[1] on Wed, 14 Apr 2004 20:56:29 GMT
View Forum Message <> Reply to Message

bridgemat wrote:
> Hey, thanks Paul! I'll give it a try. My actual data is all float/real
> kind of stuff, so I hopefully wouldn't have this problem there. I was
> just trying to see if I could do it with an "easy" case, but I guess
> simple isn't always easy! :)

Hello,

Best thing is to present the case that is causing the problem.

You're right, though. You shouldn't have the problem I rabbited on about with regular floats - assuming you have the default declarations in IDL, e.g.

```
  x = 0.0
```

and Fortran90,

```
  REAL :: x
  x = 0.0
```

If you have either

```
  ; IDL
  x = 0.0d
```

or

```
  ; Fortran90
  INTEGER, PARAMETER :: dp = SELECTED_REAL_KIND( 15 )
  REAL( dp ) :: x
  x = 0.0_dp
```

but not both, then the same problem may be occurring.

BTW, I read/write datafiles in IDL and write/read them in Fortran90 all the time (both the sequential and direct unfromatted variety). Never had any problems - as long as I'm on the same system. Otherwise you may have byte-sex issues, i.e. reading files created on a little-endian system (PC,Alpha) on a big-endian one (IBM,SGI,Sun,HP,...) or vice-versa.

paulv