Subject: Re: IDL & F90
Posted by nhbkmich on Wed, 02 Jul 1997 07:00:00 GMT
View Forum Message <> Reply to Message

Chuck Rohde (carohde@mtu.edu) wrote:

: Has anybody used IDL in conjunction with Fortran 90? I can get them to pass
: values, but not arrays. I'm using a C style passing convention (compiler
: flag) but I still can't seem to resolve the arrays correctly. I've dumped
: the
: pointer locations and the mem. address of the arrays seems out of place
: (differing by several orders of magnitude from the pointers to the simple
: integers). This is either extremely lousy memory management on NT's part
: (something I wouldn't put by it) or something is wrong... I suspect the
: latter
: because when I try to resolve the array pointer, it biffs and seg faults
: on me. Any ideas would be helpful.

: Thanks

: chuck

Hi Chuck!

Yes, I've used IDL's call_external to Fortran 90 several times successfully.
This was under Unix, so I don't really know whether my experiences are of
any help for you. But maybe there are some similarities.

First, passing arrays with assumed-shape arguments on the f90 side will be
very difficult, due to the dope vector passing mechanism, f90 usually uses
with this kind of argument. Your weird pointer values are possibly caused
by the f90 routine interpreting the pointer to the array as a pointer to a
dope vector. Use fixed-size or assumed-size instead.

Since no f90 compiler, I have access to, can handle the C calling convention,
I always put a C wrapping function in between and compile the f90 part as
usual. Example:

A Fortran 90 function, returning the product of all elements of an array:

```
  FUNCTION prod(n, array)
    IMPLICIT NONE
    INTEGER             :: prod
    INTEGER             :: n
    INTEGER, DIMENSION(n):: array

    prod = PRODUCT(array)
```

```
  END FUNCTION prod
```

The corresponding C wrapper. You will probably have to use a different name than "prod_" for the f90 function, according to the naming conventions of your f90 compiler:

```
  int prod_(int *n, int *array);  /* f90 prototype */

  int prod_argv(int iargc, void *argv[])
  {
    return prod_(argv[0], argv[1]);
  }
```

These functions are compiled and put into a shared object (called DLL under Windows?). I chose the name "prod.so" here. The call from IDL then looks like:

```
  FUNCTION prod, array
    RETURN, CALL_EXTERNAL('prod.so', 'prod_argv', $
                          N_ELEMENTS(array), $
                          LONG(array), $
                          VALUE = [0b, 0b])
  END
```

This example works with IDL 4.0.1 under HP-UX 10.20 together with NAG f90 (and a libf90.sl shared library, kindly supplied by NAG on request. For some reasons, most vendors for HP compilers apparently think that dynamic linkage is not of interest with f90...)

Hope this helps and I'm curious about your experiences!
Michael
--

 ********** **********
Michael Steffens Institut f. Meteorologie u. Klimatologie
Tel.: +49-511-7624413 Universitaet Hannover
email: Michael.Steffens@mbox.muk.uni-hannover.de Herrenhaeuser Str. 2
steffens@muk.uni-hannover.de D-30419 Hannover

PGP fingerprint = FA BE 6C 1C F6 C3 EC 33 DD 42 6B 7F DE CF 84 B8