Subject: Re: Calling fortran under unix
Posted by korpela on Wed, 02 Oct 1996 07:00:00 GMT
View Forum Message <> Reply to Message

In article <32523466.41C6@mod5.ag.rl.ac.uk>,
Simon Williams  <williams@mod5.ag.rl.ac.uk> wrote:
> I gather that the CALL_EXTERNAL interface is rather different
> under Unix, and that I might have to write some sort of
> intermediate "wrapper" between the idl and the fortran.
>
> I would be very grateful if anyone could show me how to do
> it for a simple example:
>
>       subroutine simple(a,b,c)
>
>       integer a,b,c
>
>       c=a+b
>
>       end

You'll have to write a C wrapper that calls your fortran subroutine.
Here's an example that may work on your above example.

```
 ------------------------------------------------------------ --------------
/* here's the prototype for the fortran function */
void simple_(int *a, int *b, int *c);

int wrapper(int argc, void *argv[])
{
  int *a, *b, *c;

  /* check the number of arguments */
  if (argc == 3) {
    a=(int *)argv[0];
    b=(int *)argv[1];
    c=(int *)argv[2];
    simple_(a,b,c);
    return(0);
  } else {
    return(-1);
  }
}
 ------------------------------------------------------------ --------------
```

you'd hace to link the wrapper function and the fortran function into the
same shared library.  Under SunOs the commands would be...

% gcc -Wall -fpic -c wrapper.c

% f77 -PIC -c simple.f
% ld -o simple.so -assert pure-text simple.o wrapper.o -lF77

Under your unix they may be different.

The call_external statement would be:

error_code=call_external('simple.so','wrapper',a,b,c)

---

The general rules for linking C and Fortran are

1) Most varieties of fortran append an underscore to function names.
   The fortran function hello() becomes the C function hello_()

2) Fortran subroutine and function parameters are passed as pointers.
   subroutine garbage(a,b,c) INTEGER*4 a, REAL b, INTEGER*2 c
   becomes void garbage_(long *a, float *b, short *c)

3) Watch your types! "integer" is not necessarily "int". In your fortran
   code use integer*2 and integer*4 in the C code use short and long.
   Use real and double precision in fortran. float and double in C.
   None of the above is guaranteed to work, but it's a starting point.

4) Fortran arrays are passed as a pointer to the first element of the array
   subroutine garbage(a,b,c) INTEGER*4 a(100), REAL B(100), INTEGER*2 c(100)
   becomes void garbage_(long *a, float *b, short *c)

5) To avoid all these problems, skip fortran and move straight to C.

Eric

--
Eric Korpela | An object at rest can never be
korpela@ssl.berkeley.edu | stopped.
<a href=" http://www.cs.indiana.edu/finger/mofo.ssl.berkeley.edu/korpe la/w">
Click here for more info.</a>