Calling C from Fortran

Calling C from Fortran

The Unix portable Fortran-77 compiler (`f77`) is written almost entirely in C. The second pass of the compiler is the same one used by the C compiler, and most `f77` library routines are simply interfaces to corresponding C library routines. However, since Fortran does not support data structures like those used in C, you may not be able to take advantage of all the functionality that the `curses` library offers. Manipulating windows with `curses` is especially difficult. If you are interested in using routines involving data structures, you should probably use C instead of Fortran.

To call C routines from a Fortran program, you will have to write some C code. Fortran passes arguments by reference or address, so the C function has to be prepared to accept the variable as an address. This means that you will have to write functions in C that are called from Fortran that set up the arguments properly before calling the library function. Schematically, this might be something like this:

In the C source file:

```
foo_(bar) /* See below for information on the underscore! */
 int *bar; /* Variables are passed by address. */
```

In the Fortran source file:

```
call foo(baz) /* Assuming that "baz" is an integer. */
```

The underscore is important because Fortran uses the character to keep its symbols straight.

If you were calling the `curses` routine `move()`, you might do something like this:

The call to the C interface functions are made in the Fortran source file (named `test.f`):

```
call initscr()
 call clear()
 .
 .
 .
 call move(x, y)
 .
 .
 .
 call refresh()
 call endwin()
 end
```

...where x and y are integers specifying the new coordinates.

The C source file (named `curses.c`) contains the interface routine to the `curses` library function `move()`, along with the other C functions that provide an interface to the some other `curses` functions:

```
#include <curses.h>

 initscr_()
 {
   initscr();
 }

 clear_()
 {
   clear();
 }

 move_(x, y)
 int *x, *y;  /* These are pointers  */
 {
   move(*x, *y);
 }

 refresh_()
 {
   refresh();
 }

 endwin_()
 {
   endwin();
 }
```

The routines are compiled by using these commands:

```
cc -c curses.c
 f77 test.f curses.o -lcurses -ltermcap
```

If you are using macros defined in `/usr/include/curses.h` in your Fortran file, be warned that they assume conventions of the C language. Be aware that this may affect the results you obtain when using them in Fortran.