Subject: Re: Do I need a DLM Wrapper for this?
Posted by b_gom on Wed, 10 Dec 2003 19:59:12 GMT
View Forum Message <> Reply to Message

Brian,

IF you mean 'can I do this with CALL_EXTERNAL', then the answer is yes; you don't really need a DLM. DLMs are much more convenient, however, once you get used to the syntax. In general, I think you want a DLM when you need access to IDL internals, and you want your code to be called as if it were a built-in IDL function. It's also easier to get data types and memory screwed up when using CALL_EXTERNAL.

I've posted a sample of how you can do this with a DLM at:
http://people.uleth.ca/~brad.gom/fftw/
You'll need to unzip the VC++ project and the FFTW library into c:\fftw. If you don't have VC++, then you should be able to figure out the code from fftw_dlm.c
The code only calls the real, 1-D FFT, but it should be clear how to extend it to the other functions. Note this code is non-polished.

Good luck

Brad

"Brian" <brian.huether@NOdlrSPAM.de> wrote in message
news:<42248903bfffbe75ba4af3e74473e176@news.teranews.com>...
> I am somewhat new to IDL, so you can imagine how confusing I find the notion
> of DLM wrappers...
>
> In any case, I have made several posts about the FFTW3 dll, and I am getting
> closer to being able to get it working, but now I have come across DLMs, and
> am wondering if I need one (and in general wondering under which
> circumstances I need one).
>
> Do I need to write a DLM wrapper to truly use this fftw3 dll? In the manual
> it says
>
> The basic usage of FFTW to compute a one-dimensional DFT of size N is
> simple, and it
> typically looks something like this code:
> #include <fftw3.h>
> ...
> {
> fftw_complex *in, *out;
> fftw_plan p;
> ...
> in = fftw_malloc(sizeof(fftw_complex) * N);

> out = fftw_malloc(sizeof(fftw_complex) * N);
> p = fftw_plan_dft_1d(N, in, out, FFTW_FORWARD, FFTW_ESTIMATE);
> ...
> fftw_execute(p); /* repeat as needed */
> ...
> fftw_destroy_plan(p);
> fftw_free(in); fftw_free(out);
> }
> (When you compile, you must also link with the fftw3 library,
> e.g. -lfftw3 -lm on
> Unix systems.)
> First you allocate the input and output arrays. You can allocate them in any
> way that
> you like, but we recommend using fftw_malloc, which behaves like malloc
> except that it
> properly aligns the array when SIMD instructions (such as SSE and Altivec)
> are available
> (see Section 3.1.1 [SIMD alignment and
> tw malloc], page 15).
>
> I am just confused how I pass an array in IDL to this dll.
>
> -brian