Subject: Re: Can DLLs Multi-task?
Posted by Peter Mason on Thu, 03 Oct 2002 22:32:09 GMT
View Forum Message <> Reply to Message

Here's my "string and glue" approach to this sort of problem. I haven't done image-capturing like this in IDL, but I have done other threaded programs.
The temperature-monitoring calls are probably okay as they are (fast?) but the image-capturing call needs some work. From the sound of it, it is a synchronous call. (Gianluca is basically asking about ways around synchronous calls.) Some wrapper functions are required to make it run in a separate thread so that it is asynchronous as far as your IDL program is concerned. Briefly (assuming you'll be capturing many images): A call to create a C-side thread, a call to trigger an image capture (which is then done in the separate thread), possibly a status-polling call, possibly one or two retrieve-capture calls, and finally a call to end the thread and clean up once you're done with all your image capturing.
If the image-capturing call can be passed an array (allocated on the IDL side with a statement like img=BYTARR(x,y)) into which to store the image then life is relatively simple. All you need is some way for the C routine to let the calling IDL program know when the image is ready. The simplest approach is to write a little C routine that just reports whether or not the image-capturing thread is busy (or something like that). (This is off the main IDL thread, not the image-capture thread.) You'd then poll this off a timer event on the IDL side. A more sophisticated approach is to issue an event from the C image-capture thread to IDL when the image is ready, along the lines that Rick has described. (I haven't used IDL_WidgetStubIssueEvent() myself, but I have other, less elegant ways of achieving this sort of thing.)
If the image-capturing insists on its own address for the image then things could be a bit more complicated. Three possibilities come to mind. If the image is always a fixed size you could allocate it up front in a "heap" variable on the IDL side, pass this to the image-capture trigger call, and just make the image-capture call copy the image across when it's ready.
The next approach is to have an "image size" C routine that you call from IDL when the image is ready, allocate an array for the image in IDL, then call an "image retrieve" C routine that simply copies the image to this array. The slickest approach requires that you build a DLM rather than just use CALL_EXTERNAL. Here, the "image-retrieve" C routine returns an IDL array of the captured image. This is very easy to do with RSI's IDL_ImportArray() C function.

The mechanics of threading on Win32 are relatively straightforward. The hard parts are knowing what to thread (not a problem here) and retaining a clear understanding of which thread might be accessing what memory when, especially on multiprocessor computers (often a problem; nasty little teeth, too). There are various ways to prevent memory-access conflicts (e.g., Win32 critical sections, events, mutexes), but simply taking care

with your "logic flow" goes a long way towards avoiding them.

Cheers
Peter Mason

"David Fanning" <david@dfanning.com> wrote in message
news:MPG.1805ef78a52763df9899d1@news.frii.com...
> Folks,
>
> Here is a question from an IDL user who has no newsgroup
> access. He asked me if I would pass this question on
> to you:
>
> ;**
> I am developing an IDL program which have to call some external C
> routines (actually some DLL, because I work on Windows environment).
> I am wondering if it is possible in some ways to call more than one DLL
> at the same time, I.e. in multitasking, so telling to IDL not to wait
> for the DLL return.
>
> For example I need to launch a DLL to continuously monitor some
> temperature sensors, but in the meanwhile I have to run another DLL to
> read an image from a CCD device. I would need to have the temperature
> variables continuously updated by the first DLL so that I can read and
> display their values while getting the image from the CCD device.
> Is all that possible with IDL?
> (I have IDL 5.5)
>
>
> Thank you very very much for your valuable help,
>
> Gianluca Li Causi
> licausi@coma.mporzio.astro.it
> ;**
>
> Cheers,
>
> David
>
> --
> David W. Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Phone: 970-221-0438, E-mail: david@dfanning.com
> Coyote's Guide to IDL Programming: http://www.dfanning.com/
> Toll-Free IDL Book Orders: 1-888-461-0155