

Re: DirectShow Thread Priority

Source:

<http://www.tech-archive.net/Archive/Development/microsoft.public.win32.programmer.directx.video/2005-04/msg00>

- *From:* "Peter Duniho" <NpOeStPeAdM@xxxxxxxxxxxxxxxxxxxx>
 - *Date:* Tue, 5 Apr 2005 00:31:08 -0700
-

"Iain" <Iain@xxxxxxxxxxxxxxxxxxxx> wrote in message
news:1gq72hal9yibx.9896zxrj580w.dlg@xxxxxxxxxxxxxx
> I'm not persuaded that it does run at a higher priority. Certainly the
> audio renderer does not.

Even if so, I think Tim's main point still stands. It is hard to know which threads are the important ones, even if you identify the ones that are new after you run the graph.

Also, it's unlikely that a performance issue would be related to thread priority anyway. Thread priority is only one factor with respect to which thread gets CPU time when. Most threads will wind up yielding long before their slice is up, usually because they wind up blocked on i/o or waiting for another thread.

IMHO, the original poster's goal isn't reasonable. Assuming there are no other CPU-intensive tasks going on during video rendering, the rendering will take most of the CPU time anyway. That will happen naturally as the thread (presumably) would be one of the few threads (or the only thread) on the system actually using 100% of its time slice. If there ARE other CPU-intensive tasks, then either they are important or they are not. If they are, then starving them for video isn't reasonable. If they aren't, then why are they running in the first place?

Messing with thread priority is almost always the wrong strategy. Almost always, the existing priorities are set that way for a reason.

Just as an example: one video application (if I recall correctly, it's TMPGEnc) sets its priority to a high setting while encoding, rather than the default normal setting. The application doesn't get its encoding done any faster; it just winds up making everything else on the system run like crap. The CPU time it takes to handle mouse-clicks, or even running a solitaire card game, is completely inconsequential compared to the CPU time required by the encoder, and doing stuff like that won't result in any noticeable difference in performance by the encoder.

If I were trying to run a 3D game, or another encoder, or something like that while the application was encoding, then sure...the application would

Re: DirectShow Thread Priority

suffer and not encode as quickly. But doing that would be dumb.

I realize the rendering example is a little different. But it's not much