

Re: MSDN volatile sample

Source: <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.language/2007-12/msg01119.html>

- *From:* "Alexander Grigoriev" <alegr@xxxxxxxxxxxxxx>
 - *Date:* Sun, 30 Dec 2007 07:37:11 -0800
-

A thread running in a tight loop with Sleep(0) will consume all remaining CPU time, AND won't let lower priority background threads to run. It won't starve other threads of the same priority, though.

"George" <George@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote in message news:4FB8F7B7-4452-435C-81AE-6267ED829E4C@xxxxxxxxxxxxxxxxxxxxxx

Thanks Alexander,

You mean Sleep(0) will consume 100% CPU time? I can not believe it because in MSDN, it is mentioned that Sleep(0) will let current thread *If you specify 0 milliseconds, the thread will relinquish the remainder of its time slice but remain ready*. I assume it means the current thread will contribute its remaining running time to other ready status thread.

[http://msdn2.microsoft.com/en-us/library/ms686298\(VS.85\).aspx](http://msdn2.microsoft.com/en-us/library/ms686298(VS.85).aspx)

Do you have a test program which proves Sleep(0) will still occupy 100% CPU time?

regards,
George

"Alexander Grigoriev" wrote:

Sleep(0) is causing kernel trip anyway (lots of them in the usual case, since it's called in a loop), compared to only 2 for WFSO/SetEvent pair. And this loop doesn't allow to stop processor clock to reduce CPU power. When I see an application consuming 100%, that gives me doubts about its general quality, and I get rid of it.

Re: MSDN volatile sample

"Alex Blehman" <tkfx.REMOVE@xxxxxxxxxx> wrote in message
news:uFdF0\$WSIHA.3940@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

"Alexander Grigoriev" wrote:

Anyway, the sample illustrates very poor synchronization practice which should not be followed. Proper signalling using events makes volatile qualifier unnecessary.

There is nothing wrong with this technique if it used correctly. Synchronization with events or other kernel objects brings the high price of user space to kernel space trip. It is not uncommon that you need to synchronize within single process only. In these cases volatile variables may provide adequate solution.

Alex