High-performance multithreading is very hard

devblogs.microsoft.com/oldnewthing/20040528-00

May 28, 2004

Raymond Chen

Among other things, you need to understand weak memory models. Hereby incorporating by reference Brad Abrams' discussion of volatile and MemoryBarrier(). In particular, Vance Morrison's discussion of memory models is important reading. (Though I think Brad is being too pessimistic about volatile. Ensuring release semantics at the store of "singleton" is all you really need – you want to make sure the singleton is fully constructed before you let the world see it. volatile here is overkill.) Vance's message also slyly introduces the concepts of "acquire" and "release" memory semantics. An interlocked operation with "acquire" semantics prevents future reads from being advanced to before the acquisition. An interlocked operation with "release" semantics prevents previous writes from being delayed until after the release.

In the absence of explicitly-named memory semantics, the Win32 Interlocked* functions by default provide full memory barrier semantics. However, some functions, like InterlockedIncrementAcquire, forego the full memory barrier semantics and provide only acquire or release semantics.

Raymond Chen

Follow