

# Re: How to do atomic read?

---

*Source:*

<http://www.tech-archive.net/Archive/Development/microsoft.public.development.device.drivers/2007-06/msg00608.html>

---

- *From:* "Andrew Sha" <[universalkludge@xxxxxxxxxxx](mailto:universalkludge@xxxxxxxxxxx)>
  - *Date:* Wed, 20 Jun 2007 22:24:26 -0500
- 

Exactly

the InterlockedExchange() implementation in the ntkrnlmp looks like

```
xchg edx, [ecx]
mov eax, edx
retn
..
```

"Anton Bassov" <[AntonBassov@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:AntonBassov@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx)> wrote in message [news:E43947F7-0A80-4461-8C18-E3D3C7486E0B@xxxxxxxxxxxxxxxxxxx](mailto:news:E43947F7-0A80-4461-8C18-E3D3C7486E0B@xxxxxxxxxxxxxxxxxxx)

rather than by InterlockedXXX functions. These functions use LOCK prefix that

not all of them

Actually, it looks like they all do (at least in the user mode under XP SP2), although, according to their names, not all of them have to – I fully agree with you here. For example, if InterlockedExchange() relied upon XCHG instruction, as its name suggests, it would not have to use LOCK prefix, because LOCK is automatically assumed for XCHG instruction. However, for this or that reason it uses CMPXCHG exactly like InterlockedCompareExchange() does, so that it needs LOCK prefix.

Anton Bassov

"Andrew Sha" wrote:

Re: How to do atomic read?

rather than by InterlockedXXX functions. These functions use LOCK prefix that

not all of them

"Anton Bassov" <AntonBassov@xxxxxxxxxxxxxxxxxxxxxxxxxxxx> wrote in message  
[news:F4120CD8-0F84-48CC-8AE5-FCFD99D831B0@xxxxxxxxxxxxxxxxxxxx](mailto:news:F4120CD8-0F84-48CC-8AE5-FCFD99D831B0@xxxxxxxxxxxxxxxxxxxx)

But atomic reads are silly.

Actually, they are not, but in context of this discussion they are – atomicity of reads and writes should be achieved by proper alignment, rather than by InterlockedXXX functions. These functions use LOCK prefix that does not apply to MOV instruction, i.e. to reads and writes – if you use it with MOV, CPU is going to raise an exception. Therefore, InterlockedXXX functions should be used not for reads and writes but for increments, decrements, exchanges and other operations that can be used with LOCK prefix ( please consult Volume 2 of Intel Developer's Manual for the list of instructions that can be used with LOCK prefix )

When it comes to atomicity of reads and writes, it should be achieved simply by proper alignment. One-byte memory access with MOV instruction is always atomic, as well as as WORD and DWORD ones that are aligned on respectively WORD and DWORD boundary.

Re: How to do atomic read?

Anton Bassov

"Mark Roddy" wrote:

Odbell@xxxxxxxxx wrote:

On Jun 20, 9:11 am,  
"Alexander Grigoriev"  
<a...@xxxxxxxxxxxxxx>  
wrote:

<Odb...@xxxxxxxxx>  
wrote in  
message

But,  
in  
case  
I  
am  
not  
seeing  
the  
obvious,  
what's  
wrong  
with:  
LONG  
asignee  
=  
InterlockedExchange(&var,  
var);

First, it's  
NOT  
atomic.  
There is a  
window for  
change.  
This IS  
atomic:

LONG  
asignee =  
InterlockedCompareExchange(&var,  
0, 0);

Re: How to do atomic read?

Thanks for the correction.  
Can you explain why?

Yeah, I'd be fascinated by that explanation.  
But atomic reads are  
silly.  
What is the point?