

Re: problem to understand behavior of I/O manager

Source:

<http://www.tech-archive.net/Archive/Development/microsoft.public.development.device.drivers/2008-02/msg00508.html>

- *From:* "David Craig" <drivers@xxxxxxxxxx>
 - *Date:* Fri, 22 Feb 2008 23:42:31 -0800
-

That deduction seems to be a rather large leap. The driver would report an error such as STATUS_NO_MEMORY (?) to indicate a problem with processing the request. Most storage drivers, when written correctly, will ensure that any needed buffers, work items, context structures, and others are preallocated in the dispatch routine. The first processing on completion of a storage request is done at DIRQL, though that may be in the Microsoft written bus driver such as pci.sys. The second is at DISPATCH_LEVEL where some processing may require running at PASSIVE_LEVEL. That processing would be scheduled via a worker thread, but almost no one would wait until the time to schedule a work item to allocate memory since it can fault if the system is low on memory in the nonpaged pool. In normal circumstances most storage drivers do not need to use a worker thread. They have to be written to handle page files for the OS and cannot take any hit on memory allocation.

"Kalle Olavi Niemitalo" <kon@xxxxxx> wrote in message
news:877igwtc03.fsf@xxxxxxxxxxxxxxxxxxxxxxxx

"Maxim S. Shatskih" <maxim@xxxxxxxxxxxxxxxxxxxx> writes:

Do these mean that, if Win32 has already reported
ERROR_PENDING
to the application, the kernel cannot later run out of memory
when trying to report the completion of the I/O?

The kernel itself will not, but the particular driver can.

I don't quite see what you mean.
Are you implying that a driver might for example attempt to
allocate a work item from which it would call IoCompleteRequest,
and if IoAllocateWorkItem fails, the driver would just forget
about the IRP?
To me, that would seem a bug in the driver.

Re: problem to understand behavior of I/O manager