

Re: High-performance IO

But eventually they must issue a DMA transfer request (does anyone still use PIO?) and they work only with physical memory. That rises two questions:

a) what is the upper limit of a DMA disk transfer, how is it computed on a Windows XP-based machine and by whom?

b) is the underlying hardware able to access physical memory above 4GiB? There may still be some 32-bit oddities etc.

I know that AWE can give the programmer a way to use huge memory, but the problem is whether a physical memory block from the AWE area is guaranteed to be reachable via PCI etc. If one completes a data block in AWE and issue an IO request, it would not be desirable to receive an error code like `E_INVALID_PHYSICAL_MEMORY_RANGE` or something. I ask, because the following excerpt from MSDN is the seed of doubt:

"A similar restriction is that AWE window address ranges and memory pools cannot be used as data buffers for graphics or video calls."

And since AGP and PCI buses are being implemented in a quite similar way, this remark naturally scales up the problem to a question "does this restriction apply to disks too?"

Unfortunately, there is not much information available at the level of detail I would like to know... :-)

The only problem is that AWE requires `SeLockMemory` privilege

I believe it will not be a big problem, especially that it is just an alternative control flow path -- without that privilege my application will simply use the old good `VirtualAlloc`-based way. But indeed, the temptation to use AWE is very strong... :-)

Best regards
Piotr Wyderski

PS. The current, improved implementation is damn fast and no longer is the performance limiting factor. :-)