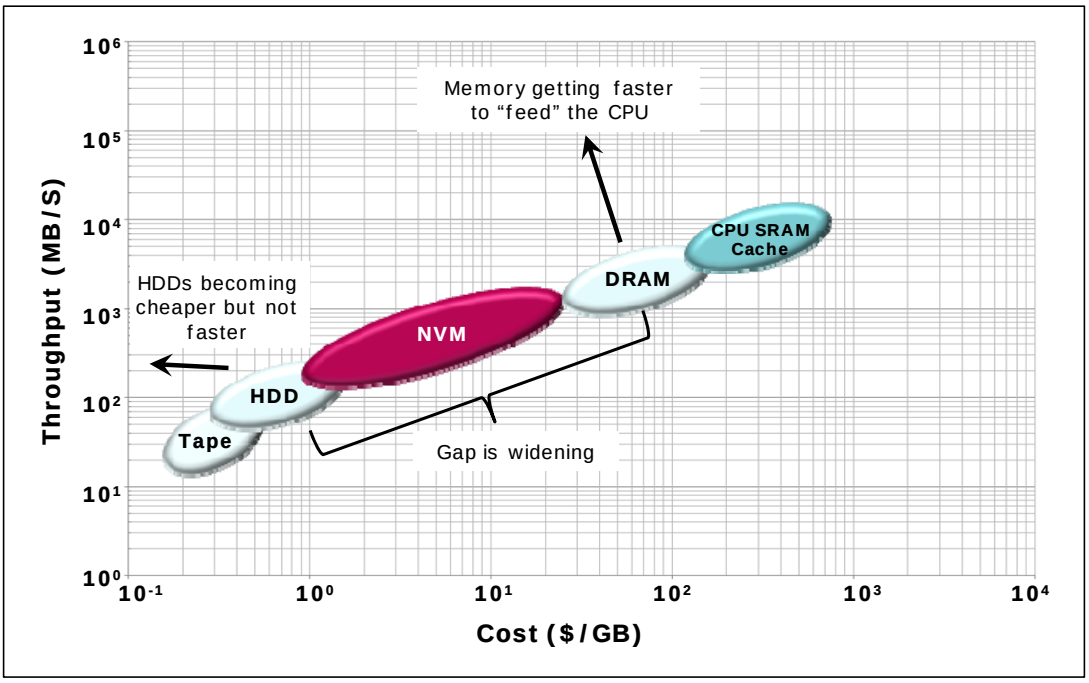
NVM Express Explained

Why NVM Express?

There is an increasing gap in the price/performance of DRAM and hard drives, as shown in Figure 1. NVM in the form of solid state drives (SSDs) is filling this gap, creating an "I/O Memory Tier" in the system.

Figure 1: NVM Filling Price/Performance Gap

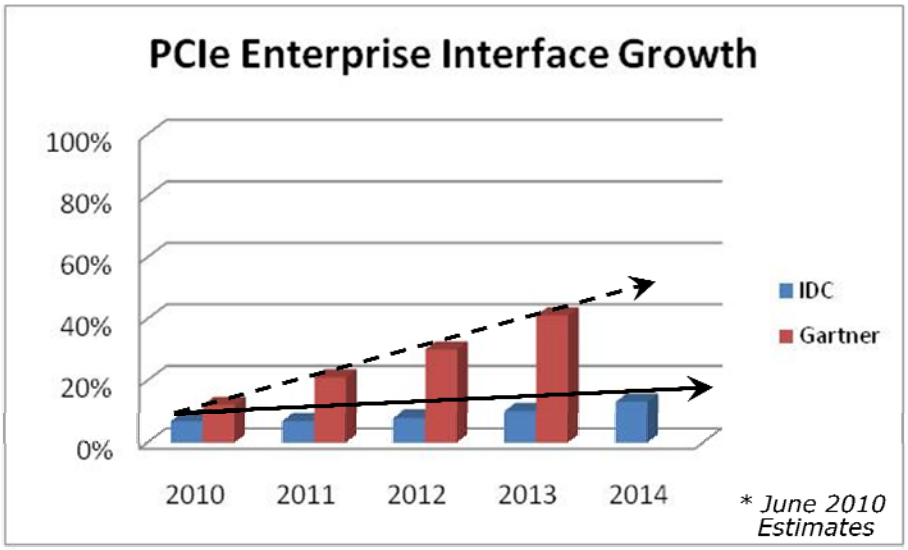


PCI Express (PCIe) connectivity on platforms continue to rise. For example, the Intel® 5000 Chipset included 24 lanes of PCIe Gen1 that then scaled on the Intel® 5520 Chipset to 36 lanes of PCIe Gen2, increasing both number of lanes and doubling bandwidth per lane. PCIe is the highest performance I/O interface today delivering 1 GB/s per lane with PCIe Gen3.

PCIe SSDs are being delivered to the market today with unmatched performance. Many PCIe SSDs support PCIe Gen2 with 8 lanes, delivering over 3 GB/s. Moving to PCIe Gen3, this doubles the bandwidth that can be delivered in a single device to 6 GB/s. Latency is reduced by several microseconds due to a faster interface as well as the ability to directly attach to the chipset or CPU. In addition, direct attach to the chipset or CPU may eliminate an external host bus adapter which saves 7 to 10 Watts of power and over $10 in cost.

Analysts see a great opportunity for PCIe SSDs in the Enterprise market segment, as shown by the forecasts in Figure 2. To enable the faster growth curve, a standard driver and consistent feature set would be valuable.

Figure 2: Analyst Projection of PCIe Interface Growth in Enterprise SSDs



Adoption of PCIe SSDs is inhibited by different implementations and unique drivers. Each SSD vendor provides a driver for each OS that OEMs must validate. Each SSD vendor implements a different subset of features in a different way leading to needless extra qualification effort by the OEM.

To enable faster adoption and interoperability of PCIe SSDs, industry leaders have defined the NVM Express standard. The standard includes the register programming interface, command set, and feature set definition. This enables standard drivers to be written for each OS and enables interoperability between implementations that shortens OEM qualification cycles.

NVM Express defines an optimized command set that is scalable for the future and avoids burdening the device with legacy support requirements. However, there are existing applications and software infrastructure built upon the SCSI architectural model. The Workgroup is defining a translation document that defines a mapping between SCSI and NVM Express specifications to enable a seamless transition to NVM Express by preserving existing software infrastructure investments. This translation may be done as a layer within the NVM Express driver. The document is targeted for publication in May.

NVM Express Overview

NVM Express is a scalable host controller interface designed to address the needs of Enterprise and Client systems that utilize PCI Express based solid state drives. The interface provides an optimized command issue and completion path. It includes support for parallel operation by supporting up to 64K command queues within an I/O Queue. Additionally, support has been added for many Enterprise capabilities like end-to-end data protection (compatible with T10 DIF and DIX standards), enhanced error reporting, and virtualization.

The interface has the following key attributes:

- Does not require uncacheable / MMIO register reads in the command issue or completion path.
- A maximum of one MMIO register write is necessary in the command issue path.
- Support for up to 64K I/O queues, with each I/O queue supporting up to 64K commands.
- Priority associated with each I/O queue with well defined arbitration mechanism.
- All information to complete a 4KB read request is included in the 64B command itself, ensuring efficient small random I/O operation.
- Efficient and streamlined command set.
- Support for MSI/MSI-X and interrupt aggregation.

- Support for multiple namespaces.
- Efficient support for I/O virtualization architectures like SR-IOV.
- Robust error reporting and management capabilities.

The specification defines a streamlined set of registers whose functionality includes:

- Indication of controller capabilities
- Status for device failures (command status is processed via CQ directly)
- Admin Queue configuration (I/O Queue configuration processed via Admin commands)
- Doorbell registers for scalable number of Submission and Completion Queues

The capabilities that the controller supports are indicated in the Controller Capabilities (CAP) register and as part of the Controller and Namespace data structures returned in the Identify command. The Identify Controller data structure indicates capabilities and settings that apply to the entire controller. The Identify Namespace data structure indicates capabilities and settings that are specific to a particular namespace.

Enhanced NVMHCI is based on a paired Submission and Completion Queue mechanism. Commands are placed by host software into the Submission Queue. Completions are placed into an associated Completion Queue by the controller. Multiple Submission Queues may utilize the same Completion Queue. The Submission and Completion Queues are allocated in host memory.

An Admin Submission and associated Completion Queue exist for the purpose of device management and control – e.g., creation and deletion of I/O Submission and Completion Queues, aborting commands, etc. Only commands that are part of the Admin Command Set may be issued to the Admin Submission Queue.

An I/O Command Set is used with an I/O queue pair. This specification defines one I/O Command Set, named the NVM Command Set.

Host software creates queues, up to the maximum supported by the controller. Typically the number of command queues created is based on the system configuration and anticipated workload. For example, on a four core processor based system, there may be a queue pair per core to avoid locking and ensure data structures are created in the appropriate processor core's cache. Figure 3 provides a graphical representation of the queue pair mechanism, showing a 1:1 mapping between Submission Queues and Completion Queues. Figure 4 shows an example where multiple I/O Submission Queues utilize the same I/O Completion Queue on Core B.

Figure 3: Queue Pair Example, 1:1 Mapping

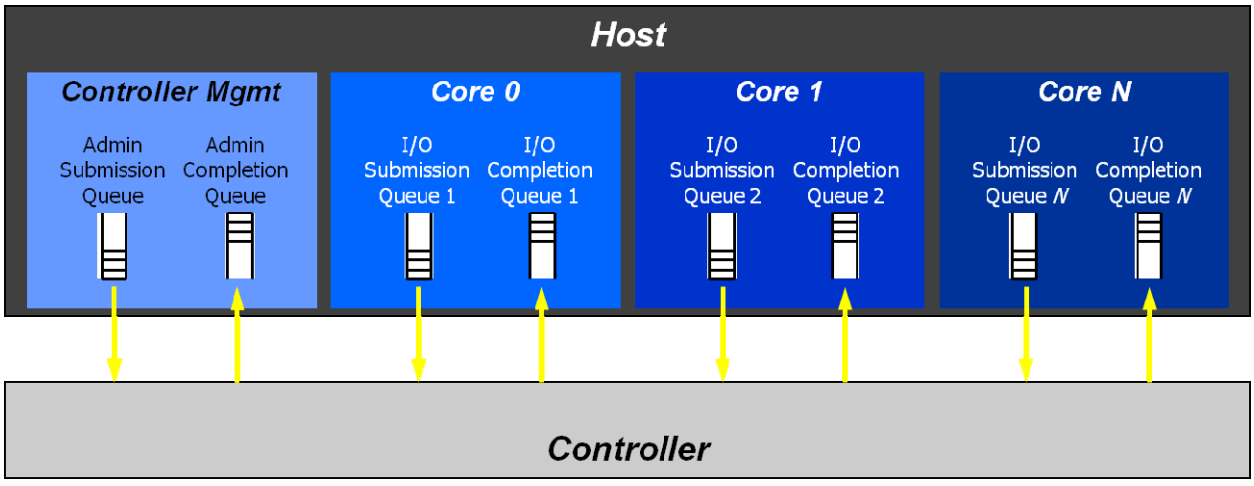
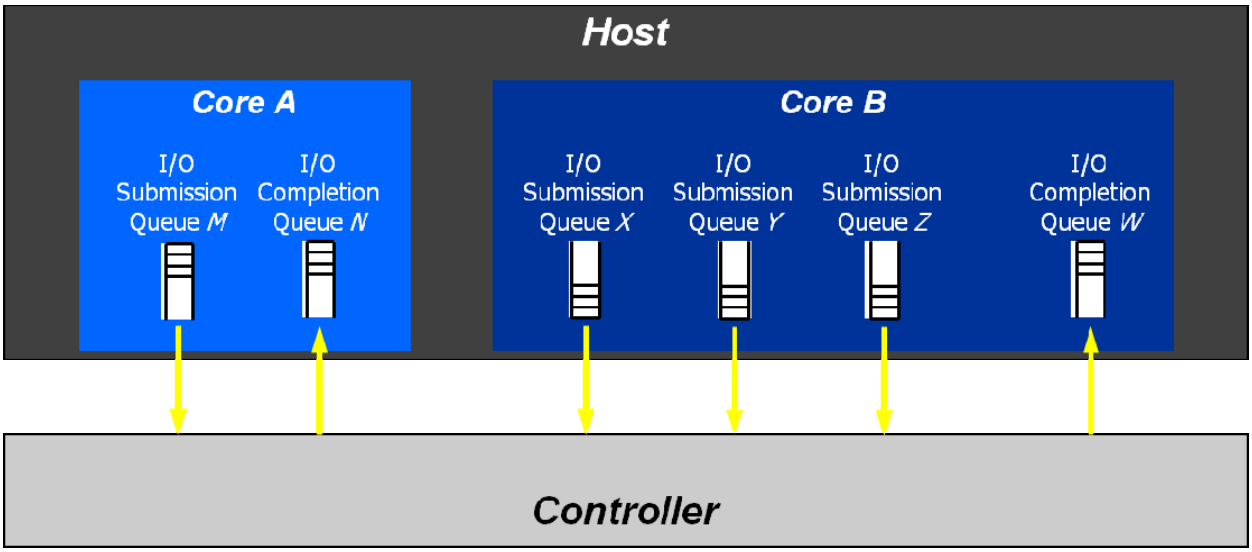


Figure 4: Queue Pair Example, *n*:1 Mapping



A Submission Queue (SQ) is a circular buffer with a fixed slot size that the host uses to submit commands for execution by the controller. The host updates the appropriate SQ Tail doorbell register when there are one to *n* new commands to execute. The previous SQ Tail value is overwritten in the controller when there is a new doorbell register write. The controller fetches SQ entries in order from the Submission Queue, however, it may then execute those commands in any order.

Each Submission Queue entry is a command. The command is 64 bytes in size. The physical memory locations in host memory to use for data transfers are specified using Physical Region Page (PRP) entries. Each command may include two PRP entries. If more than two PRP entries are necessary to describe the data buffer, then a pointer to a PRP List that describes a list of PRP entries is provided.

A Completion Queue (CQ) is a circular buffer with a fixed slot size used to post status for completed commands. A completed command is uniquely identified by a combination of the associated SQ identifier and command identifier that is assigned by host software. Multiple Submission Queues may be associated with a single Completion Queue. This feature may be used where a single worker thread processes all command completions via one Completion Queue even when those commands originated from multiple Submission Queues. The CQ Head pointer is updated by host software after it has processed completion entries indicating the last free CQ entry. A Phase (P) bit is defined in the completion entry to indicate whether an entry has been newly posted without consulting a register. This enables host software to determine whether the new entry was posted as part of the previous or current round of completion notifications. Specifically, each round through the Completion Queue locations, the controller inverts the Phase bit.

Enterprise & Client Recommendations

NVM Express is a flexible specification, allowing solutions to be built that span the needs of Client and Enterprise market segments. Figure 5 defines the features that are recommended for SSDs built for an Enterprise or Client system.

Figure 5: Enterprise & Client Recommendations

Feature	Enterprise Recommendation	Client Recommendation
Number of Queues	16 to 128	2 to 8
Physically Discontiguous Queues	Implementation choice	No
Logical Block Size	4KB	4KB
Interrupt Support	MSI-X	MSI-X
Arbitration Mechanism	Weighted Round Robin with Urgent Priority Class or Round Robin	Round Robin
PCIe Advanced Error Reporting	Yes	Yes
Firmware Update	Required	Required
End-to-end Data Protection	Yes	No
SR-IOV Support	Yes	No
Security Send & Receive	Yes	Yes