Changes in NVMe Revision 1.3

This document is intended to help the reader understand changes in the NVMe revision 1.3 specification.

One category of changes is new features that are optional capabilities. These features may be implemented by a revision 1.3 compatible device based on market need.

Another category is changes from past behavior. One set of changes here includes extensions of past functionality. Another set of changes is incompatible changes from past behavior, or new mandatory requirements for how a feature shall be implemented (where there may have been multiple reasonable interpretations of previous specification language).

The changes included in NVMe revision 1.3 were developed in Technical Proposals. The Technical Proposal documents are available to NVMe members in the Kavi repository at: https://workspace.nvmexpress.org/apps/org/workgroup/allmembers/documents.php?folder_id=148.

New Features

This section describes new features in NVMe revision 1.3.

- Identify Namespace return list of Namespace Identifiers (**mandatory**)
  - A list of Namespace Identification Descriptor structures (refer to Figure 116) is returned to the host for the namespace specified.
  - References:
    - NVMe revision 1.3 section 5.15
    - Technical Proposal 019

- Device Self-Test (optional)
  - Defines a self-test command used for both a short and long test operation.
  - References:
    - NVMe revision 1.3 section 5.8
    - Technical Proposal 001a

- Sanitize (optional)
  - Defines a command that alters all user data in the NVM subsystem such that recovery of any previous user data is not possible.
  - This command is functionally equivalent to the command of the same name in SATA and SAS implementations.
  - References:
    - NVMe revision 1.3 section 8.15 and section 5.21
    - Technical Proposal 004

- Directives (optional)

    - Defines a new mechanism for host to NVM subsystem or host to controller information exchange, including as part of IO (e.g., read, write) commands.
    - The first major Directive defined is Streams which may be used to optimize data placement to increase endurance for NAND based SSDs.
    - References:
        - NVMe revision 1.3 section 9
        - Technical Proposal 008

- Boot Partitions (optional)
    - Defines a simple bootstrap mechanism to initialize the controller without setting up Submission and Completion Queues in memory.
    - This capability is most often used in mobile systems, and is functionally equivalent to a feature provided in eMMC devices.
    - References:
        - NVMe revision 1.3 section 8.13
        - Technical Proposal 003

- Telemetry (optional)
    - Defines enhancements that allow the device to report telemetry opaque data that is either host or controller initiated. Typically, this data is triggered due to a system crash.
    - This capability is in use in other interfaces, like SATA and SAS, and this feature provides similar functionality in NVMe.
    - References:
        - NVMe revision 1.3 section 5.14.1.7 (Telemetry Host-Initiated Log), 5.14.1.8 (Telemetry Controller-Initiated Log), and section 8.14
        - Technical Proposal 009

- Virtualization Enhancements (optional)
    - Defines enhancements for NVMe to be utilized in a virtualized environment where there are both physical and virtual controllers.
    - Specifically, the NVMe model has primary controllers (which may be SR-IOV Physical Functions) and secondary controllers (which may be SR-IOV Virtual Functions) that may be used to flexible assign resources, like queues, from a primary controller to a secondary controller.
    - References:
        - NVMe revision 1.3 section 8.5 and section 5.26 (Virtualization Management command)
        - Technical Proposal 010

- NVMe-MI Management Enhancements (optional)
    - Added NVMe-MI Receive and NVMe-MI Send commands to the Admin command set in order to tunnel NVMe-MI messages.
    - Added FRU Globally Unique Identifier to be able to determine if a controller is part of the same Field Replaceable Unit.
    - References:

        - NVMe revision 1.3 section 5.17 (NVMe-MI Receive), 5.18 (NVMe-MI Send), and FGUID field in the Identify Controller data structure (Figure 109)
        - Technical Proposal 018

- Host Controlled Thermal Management (optional)
    - Provides a mechanism for the host to configure a controller to automatically transition between active power states or perform vendor specific thermal management actions in order to attempt to meet thermal management requirements specified by the host.
    - References:
        - NVMe revision 1.3 section 8.4.5 and Identify Controller data structure (Figure 109)
        - Technical Proposal 012

- Timestamp (optional)
    - Defines a way to tell the device the current time since midnight, January 1 1970 UTC, which may be used in vendor specific ways (e.g., correlating an event to a "real time").
    - There is no defined use for this capability.
    - References:
        - NVMe revision 1.3 section 5.22.1.14
        - Technical Proposal 002

- Emulated Controller Performance Enhancement (optional)
    - Defines the Doorbell Buffer Config command that may be used by emulated controllers (e.g., software defined NVMe controllers) to improve performance.
    - This feature is not typically supported by a physical / hardware based NVMe controller.
    - References:
        - NVMe revision 1.3 section 7.13 and section 5.7 (Doorbell Buffer Config command)
        - Technical Proposal 024

Changes from Past Behavior

This section describes changes (some incompatible) from past NVMe behavior.

- Get Log Page command change to add Retain Asynchronous Event functionality
    - A new bit has been added to allow an asynchronous event to not be cleared when reading the associated log page.
    - This functionality is important if you have two or more entities that may be reading the log page (e.g., a management entity that is reading an error log) to ensure only one entity is responsible for clearing asynchronous events.
    - New requirement / incompatible change in section 5.14.1.1:
        - "If the log page is full when a new entry is generated, the controller inserts the new entry into the log and discards the oldest entry."
    - References:
        - NVMe revision 1.3 section 5.14
        - Technical Proposal 005

- Globally Unique Updates
    - Defines whether the controller ever re-uses an NGUID or EUI64 value for a namespace identifier. Specifically, the following is added in the Namespace Features field of Identify Namespace with corresponding changes in the NGUID and EUI64 definitions.
        - "Bit 3 if set to '1' indicates that the non-zero NGUID and non-zero EUI64 fields for this namespace are never reused by the controller. If cleared to '0', then the NGUID and EUI64 values may be reused by the controller for a new namespace created after this namespace is deleted. This bit shall be cleared to '0' if both NGUID and EUI64 fields are cleared to 0h. Refer to section 7.9."
    - References:
        - NVMe revision 1.3 section 5.15 and Identify Namespaces data structure (Figure 114)
        - Technical Proposal 006

- SGL Dword Simplification
    - Defines a mechanism for a device to support SGL Data Blocks that require Dword alignment, rather than byte alignment.
    - The functionality is defined in a backwards compatible manner, such that older software that does not understand the Dword vs byte alignment restriction will not detect support for SGLs in devices that have more limited support.
    - A minor revision of the Technical Proposal was completed to fix the status code of SGL Data Block Granularity Invalid from 15h (which is used for a security operation called Operation Denied) to 1Eh.
    - References:
        - NVMe revision 1.3 section 4.4 and the SGL Support (SGLS) field in the Identify Namespace data structure (Figure 109)
        - Technical Proposal 007a

- Firmware Update Granularity
    - The alignment and granularity of data provided in the Firmware Image Download command is specified in the Firmware Update Granularity (FWUG) of the Identify Controller data structure.
    - If this field is observed by host software, it ensures that firmware updates will proceed smoothly.
    - References:
        - NVMe revision 1.3 section 8.1 and the Firmware Update Granularity (FWUG) field in the Identify Namespace data structure (Figure 109)
        - Technical Proposal 019

- Namespace Optimal IO Boundary
    - A field, Namespace Optimal IO Boundary (NOIOB) field indicates the optimal IO boundary for a namespace. The host should construct read and write commands that do not cross the IO boundary to achieve optimal performance.
    - References:

        - NVMe revision 1.3 Namespace Optimal IO Boundary (NOIOB) field in the Identify Namespace data structure (Figure 109)
        - Technical Proposal 019

- Non-Operational Power State Permissive Mode
    - When in Non-Operational Power State Permissive Mode, the controller may temporarily exceed the power limits of any non-operational power state, up to the limits of the last operational power state, to run controller initiated background operations in that state.
    - References:
        - NVMe revision 1.3 section 5.22.1.17 and section 8.4.1.
        - Technical Proposal 019

- Data Transfer Direction for opcodes shall be valid
    - The lower two bits for opcodes indicate the data transfer direction. In NVMe revision 1.3, these bits shall be valid if a data transfer actually takes place.
    - References:
        - NVMe revision 1.3 Opcodes for Admin Commands (Figure 41 and Figure 42) and Opcodes for NVM Commands (Figure 188)
        - Technical Proposal 019

- Reservations Changes
    - The Ignore Existing Key functionality was changed to improve compatibility with SCSI implementations of reservations.
    - References:
        - NVMe revision 1.3 section 8.8 and Reservation commands in section 6
        - Technical Proposal 019

- Operation Denied status code
    - The TCG SIIS specification uses status code 15h for certain error cases. The specification defines 15h as “Operation Denied” and retains “Access Denied” as 86h for other conditions.
    - References:
        - NVMe revision 1.3 Generic Status Codes in Figure 31
        - Technical Proposal 019

- Deallocated Value for Logical Block Data
    - Added mechanism to determine values returned for Deallocated logical blocks.
    - Added mechanism for the host to request deallocation as part of Write Zeroes.
    - References:
        - NVMe revision 1.3 Deallocate Logical Block Features (DLFEAT) field in Identify Namespace data structure (Figure 114), section 6.7.1.1 (Deallocate), section 6.16 (Write Zeroes)
        - Technical Proposal 019