LEGAL NOTICE:

This erratum to the NVM Express revision 1.3 specification is proprietary to the NVM Express, Inc. (also referred to as "Company") and/or its successors and assigns.

NOTICE TO USERS WHO ARE NVM EXPRESS, INC. MEMBERS: Members of NVM Express, Inc. have the right to use and implement this erratum to the NVM Express revision 1.3 specification subject, however, to the Member's continued compliance with the Company's Intellectual Property Policy and Bylaws and the Member's Participation Agreement.

NOTICE TO NON-MEMBERS OF NVM EXPRESS, INC.: If you are not a Member of NVM Express, Inc. and you have obtained a copy of this document, you only have a right to review this document or make reference to or cite this document. Any such references or citations to this document must acknowledge NVM Express, Inc. copyright ownership of this document. The proper copyright citation or reference is as follows: **"© 2007 - 2019 NVM Express, Inc. ALL RIGHTS RESERVED."** When making any such citations or references to this document you are not permitted to revise, alter, modify, make any derivatives of, or otherwise amend the referenced portion of this document in any way without the prior express written permission of NVM Express, Inc. Nothing contained in this document shall be deemed as granting you any kind of license to implement or use this document or the specification described therein, or any of its contents, either expressly or impliedly, or to any intellectual property owned or controlled by NVM Express, Inc., including, without limitation, any trademarks of NVM Express, Inc.

LEGAL DISCLAIMER:

THIS DOCUMENT AND THE INFORMATION CONTAINED HEREIN IS PROVIDED ON AN "**AS IS**" BASIS. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, NVM EXPRESS, INC. (ALONG WITH THE CONTRIBUTORS TO THIS DOCUMENT) HEREBY DISCLAIM ALL REPRESENTATIONS, WARRANTIES AND/OR COVENANTS, EITHER EXPRESS OR IMPLIED, STATUTORY OR AT COMMON LAW, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, VALIDITY, AND/OR NONINFRINGEMENT.

All product names, trademarks, registered trademarks, and/or servicemarks may be claimed as the property of their respective owners.

NVM Express Workgroup
c/o VTM Group
3855 SW 153rd Drive
Beaverton, OR 97003 USA
info@nvmexpress.org

NVM Express™ Technical Errata

Errata ID	**006**
Revision Date	**3/20/2019**
Affected Spec Ver.	**NVM Express™ 1.3c**
Corrected Spec Ver.	

Errata Author(s)

Name	Company
Fred Knight	NetApp Inc.
Mike Allison	Intel
David Black, Austin Bolen	Dell EMC
Judy Brock	Samsung
John Geldman, Julien Margetts	Toshiba
John Geldman, Julien Margetts	Toshiba Memory
Peter Onufryk	Microchip
Curtis Ballard	HPE
Paul Suhler	Micron
Randy Jennings, Roland Dreier	Pure Storage
Yoni Shternhell, Dave Landsman, Nadesan Narenthiran	WDC
Eric Peterson	Synopsys
Craig Lucero	HP
Parita Siddharth Dhir	Cadence
Martin Petersen	Oracle
Stacey Secatch	Seagate
Stephen Bates	Eideticom

Errata Overview

Ensure Fabric devices are account for as necessary.

Update conventions to include decimal and binary units and US format numeric separators.

Clarify several definitions

Add reference to the PCIe specification where RTD3 is defined. Added other required references

Clarify instances where “may only” was used in the text (to clarify if it was a permissive or restrictive requirement)

Update references to units to indicate if they are decimal or binary

Clarify that changes to the CC.IOCQES and CC.IOSQES fields produce undefined results after queues that use those values have been created.

Clarified that the CSTS.RDY is cleared to ‘0’ when the controller is ready to be re-enabled.

Move the description about alignment requirements for the CMB from the description for the offset register to the description for the base address register.

In the MPTR field and the SGL Data Block descriptor, add the same description about Qword alignment checks that are already used for several other address fields.

Clarify “Missing Fused Command” error description

Clarify additional uses (after a controller reset) for “Command Abort Requested” error code

Remove use of undefined CDW1

Clarify that the Identify command use of NSID depends on the CNS value

Clarification of AEN deliveries and added recommendations for handling multiple concurrent events

Clarify additional Fused command error cases

Added an already described error case to the list of completion status values for the Create I/O Completion Queue command and the Create I/O Submission Queue command

Clarifications in the Device-Self Test command and the DSTO bit (removed duplicate text)

Clarifications on Firmware Commit command usage

Clarification of the meaning of the units of values in the “Data Units Read and Data Units Written fields

Clarification that DI errors might (or might not) count intentional errors (those caused by Write Uncorrectable commands)

Clarification of Power Management

Clarify the sections of the specification that already define required behavior if an NSID that is not active is used in the Identify command (CNS=3h).

Updated comparative terms (larger, smaller, above, below) to use more precise mathematical terms (greater than, less than)

Clarify the error to return for an Identify command (CNS=12h) with NSID set to FFFFFFFFh.

Remove some duplicate text in the Set Features command that is also included in the operational description (section 7.8)

Describe handling of attempts to change a non-changeable feature.

Clarification of Autonomous power state transitions

Clarification of the meaning of the HMB Memory Return bit

Clarification of the Timestamp feature operation and impact of making this feature saveable

Clarification of Host Identifier usage

Clarification of the Reservation Persistence feature operation and the impact of this feature being saveable. This includes a new recommendation that this feature not be saveable.

Clarification that features are actually set when a Set Features command completes with success

Clarifications on the Format command and interactions with bits in the FNA field

Clarification of Sanitize interactions with Firmware activation with pending reset (only one type of reset had been described, this clarification adds the other reset types)

Clarification of FUA behavior for the Compare command, the Read command, the Write command, and the Write Zeroes command

Clarification that the length field in Range entries for the DSM command are 1-based values

Examples added to provide clarification on deallocation

Clarification of Write Zeroes command and deallocate interactions

Clarification of when NVM Subsystem Reset is initiated

Clarification of Controller Level Reset

Added configuration of CC.IOSQES and CC.IOCQES to the Queue Setup and Initialization sequence to match requirements described in other parts of the specification.

Clarification of NQNs.

Clarification of Keep Alive operations

Clarification of the Firmware Update Process to include all of the “pending reset” possibilities, rather than just one. Clarification of interactions between firmware update and power states. Clarification of actions associated with a host overwriting the firmware in the active firmware slot.

Explicit reference to the CRC-16 algorithm used for Protection Information checking.

Clarification of Protection Information checks associated with Compare commands.

Clarification of the definition of RTD3 and its relationship to the PCIe D3cold power state.

Clarification that RPMB targets and Boot partitions may be shared by controllers.

Added missing Security Send 2 step from the RPMB Block Write Flow

Clarification of Device Self-test operations being aborted by Format commands.

Clarifications about Stream resources

Clarifications related to PCIe error handling and Fatal condition handling

Clarify that the “Firmware Activation Requires Reset” error status applies to Controller Level Reset; Conventional Reset and NVM Subsystem Reset each already have their own unique status values.

Clarifications related to Idle Power and Idle Transition Power state

Clarifications in the MDTS field

Incompatible changes listed in the Incompatible Changes section

Revision History

Revision Date	Change Description
2/07/18	Initial creation
2/22/18	Bring in carry over from ECN-005
3/14/18	Bring in Power Management (and a few other things) from ECN-005
09/14/18	Begin integration of ECN-006 email requests
11/14/18	Incorporate feedback from dedicated meeting
11/16/18	Incorporate feedback from regular working group meeting
12/12/18	Incorporate feedback from dedicated meeting and emails
1/3/19 + 1/4/19	Incorporate feedback from dedicated meeting
1/24/19	Move out things to ECN-007 that will take more time.
1/31/19	Incorporate Overview summary and verify the incompatible changes list
2/28/19	Incorporate first round of 30-day feedback
3/5/19	Incorporate another round of 30-day feedback
3/7/19	Timestamp Origin field capitalized in section 5.21.1.14
3/20/19	Ratified

Incompatible Changes

Commands that request an unsupported fused operation fail. However, the specification did not specify the error. The error is now specified. Refer to section 4.10.

Inconsistent descriptions for temperature thresholds (was both “greater than” in some places and “greater than or equal to” in other places). Make it consistent to be “greater than or equal to”. The same applies to the “less than” vs. “less than or equal to”. Refer to Figure 48 and Figure 93.

Requirements for handling a Get Log Page command with invalid LPOL and LPOU fields was not specified. The required error is now specified. Refer to section 5.14.

The IDLP field description did not reference sanitize operations. That description was enhanced to specify that the device is not considered to be idle if a sanitize operation is in process. Refer to section 5.15.3.

Requirements for validation in the LBA Range Type feature were made too strict in a previous ECN; those requirements have been relaxed (“shall” changed to “may”) to improve compatibility with earlier versions of the specification. A Set Features for the LBA Range Type with NSID = FFFFFFFFh is now required to return an error. Refer to section 5.21.1.3.

The text describing requirements for operation of the Sync bit and Origin field in the Timestamp data structure was enhanced to include additional cases. Refer to section 5.21.1.14.

The interactions between overlapping Format commands and other commands allowed for the Format command to fail and for those other commands to fail. However, the specification did not specify the error. The error is now specified. Refer to section 5.23

The Security Receive command and Security Send command specified a failure status code that does not exist (Invalid Parameter). The correct error code is Invalid Field in Command. Refer to section 5.25 and section 5.26.

The Reservation Register command did not explain interactions between the CPTPL field and the saveable Reservation Persistence Feature. This interaction is now specified and includes additional requirements. Refer to Figure 220 in section 6.11.

Commands with the NSID set to FFFFFFFFh require reservation permissions on all namespaces impacted by that command. If a reservation is present on any of the impacted namespaces that does not grant permission, then the command fails with Reservation Conflict status. This had been previously assumed but is now specifically stated. Refer to section 8.8.

Preempting did not specify the error if the CRKEY did not match; the error is now specified. Refer to section 8.8.7.

When terminating a DST operation as a result of processing a Format command with “the same” NSID value, the meaning of the NSID being “the same” was vague. This has been clarified and now includes several specific cases where the DST operation is terminated in this situation. Refer to section 8.11.

Releasing of Stream Resources when a namespace is deleted was not described completely (it described only Stream Identifiers and not Stream Resources). Stream resources and Stream Identifiers are both released. Refer to section 9.3.

Description of Specification Changes

Modify a portion of section 1.4 (Theory of Operation) and section 1.5 (Conventions) as follows:

1.4 Theory of Operation

The NVM Express scalable ~~host controller~~ interface is designed to address the needs of Enterprise and Client systems that utilize PCI Express based solid state drives or fabric connected devices. The interface provides optimized command submission and completion paths. It includes support for parallel operation by supporting up to 65,535 I/O Queues with up to 64 Ki - 1 outstanding commands per I/O Queue.

…

1.5 Conventions

Hardware shall return ‘0’ for all bits and registers that are marked as reserved, and host software shall write all reserved bits and registers with the value of ‘0’.

Inside the register sections (i.e., section 2 and section 3), the following terms and abbreviations are used:

RO	Read Only
RW	Read Write
R/W	Read Write. The value read may not be the last value written.
RWC	Read/Write '1' to clear
RWS	Read/Write '1' to set
Impl Spec	Implementation Specific – the controller has the freedom to choose its implementation.
HwInit	The default state is dependent on NVM Express controller and system configuration. The value is initialized at reset, for example by an expansion ROM, or in the case of integrated devices, by a platform BIOS.
Reset	This column indicates the value of the field after a reset.

…

A 0's based value is a numbering scheme in which the number 0h represents a value of 1h and thus produces the pattern of 0h represents 1h, 1h represents 2h, 2h represents 3h, etc. In this numbering scheme, there is no method to represent the value of 0h. Values in this specification are 1-based (i.e., the number 1h represents a value of 1h, 2h represents 2h, etc.) unless otherwise specified.

~~When a size is stated in the document as KB, the convention used is 1KB = 1024 bytes.~~ Size values are shown in binary units or decimal units. The symbols used to represent these values are as follows:

Decimal		Binary	
Symbol	Power (base-10)	Symbol	Power (base-2)
kilo / k	10^3	kibi / Ki	2^{10}
mega / M	10^6	mebi / Mi	2^{20}
giga / G	10^9	gibi / Gi	2^{30}
tera / T	10^{12}	tebi / Ti	2^{40}
peta / P	10^{15}	pebi / Pi	2^{50}
exa / E	10^{18}	exbi / Ei	2^{60}
zetta / Z	10^{21}	zebi / Zi	2^{70}
yotta / Y	10^{24}	yobi / Yi	2^{80}

The ^ operator is used to denote the power to which that number, symbol, or expression is to be raised.

Some parameters are defined as an ASCII string. ASCII strings shall contain only code values 20h through 7Eh. For the string "Copyright", the character "C" is the first byte, the character "o" is the second byte, etc. The string is left justified and shall be padded with spaces (ASCII character 20h) to the right if necessary. A hexadecimal ASCII string is an ASCII string that uses a subset of the code values: "0" to "9", "A" to "F" uppercase, and "a" to "f" lowercase.

Hexadecimal (i.e., base 16) numbers are written with a lower case "h" suffix (e.g., 0FFFh, 80h). Hexadecimal numbers larger than eight digits are represented with an underscore character dividing each group of eight digits (e.g., 1E_DEADBEEFh).

Binary (i.e., base 2) numbers are written with a lower case "b" suffix (e.g., 1001b, 10b). Binary numbers larger than four digits are written with an underscore character dividing each group of four digits (e.g., 1000_0101_0010b).

All other numbers are decimal (i.e., base 10). A decimal number is represented in this specification by any sequence of digits consisting of only the Western-Arabic numerals 0 to 9 not immediately followed by a lower-case b or a lower-case h (e.g., 175). This specification uses the following conventions for representing decimal numbers:

a) the decimal separator (i.e., separating the integer and fractional portions of the number) is a period;

b) the thousands separator (i.e., separating groups of three decimal digits in a portion of the number) is a comma;
c) the thousands separator is used in only the integer portion of a number and not the fractional portion of a number; and
d) the decimal representation for a year does not include a comma (e.g., 2018 instead of 2,018).

…

Modify a portion of section 1.6 (Definitions) as follows:

1.6 Definitions

1.6.12 firmware slot

A firmware slot is a location in the controller used to store a firmware image. The controller stores ~~between~~ from one ~~and~~ to seven firmware images. ~~When downloading new firmware to the controller, host software has the option of specifying which image is replaced by indicating the firmware slot number.~~

…

1.6.21 Namespace ID (NSID)

An identifier used by a controller to provide access to a namespace or the name of the field in the SQE that contains the namespace identifier (refer to Figure 11). Refer to section 6.1 for the definitions of valid NSID, invalid NSID, active NSID, inactive NSID, allocated NSID, and unallocated NSID.

…

1.6.25 private namespace

A namespace that ~~may~~ is only able to be attached to one controller at a time. A host may determine whether a namespace is a private namespace or may be a shared namespace by the value of the Namespace Multi-path I/O and Namespace Sharing Capabilities (NMIC) field in the Identify Namespace data structure.

1.6.26 Runtime D3 (Power Removed)

In Runtime D3 (RTD3) main power is removed from the controller. Auxiliary power may or may not be provided. For PCI Express, RTD3 is the $D3_{cold}$ power state (refer to section 8.4.4).

…

Modify a portion of section 1.9 (References) as follows:

1.9 References

ISO 8601, Data elements and interchange formats – Information interchange – Representations of dates and times. Available from http://www.iso.org.

…

UEFI Specification Version 2.7A, September 2017. Available from http://www.uefi.org.

Advanced Configuration and Power Interface (ACPI) Specification, Version 6.2 Errata A, September 2017. Available from http://www.uefi.org.

…

Modify a portion of section 2.4 (MSI-X Capability) as follows:

2.4 MSI-X Capability (Optional)

…

The Table BIR and PBA BIR data structures may be allocated in either BAR0-1 or BAR4-5 in implementations. These tables should be 4 KiB aligned.

…

Modify a portion of section 3.1 (Register Definition) as follows:

3.1 Register Definition

…

3.1.3 Offset 0Ch: INTMS – Interrupt Mask Set

…

Bit	Type	Reset	Description
31:00	RW~~1~~S	0h	**Interrupt Vector Mask Set (IVMS):** This field is bit significant. If a '1' is written to a bit, then the corresponding interrupt vector is masked from generating an interrupt or reporting a pending interrupt in the MSI Capability Structure. Writing a '0' to a bit has no effect. When read, this field returns the current interrupt mask value within the controller (not the value of this register). If a bit has a value of a '1', then the corresponding interrupt vector is masked. If a bit has a value of '0', then the corresponding interrupt vector is not masked.

3.1.4 Offset 10h: INTMC – Interrupt Mask Clear

…

Bit	Type	Reset	Description
31:00	RW~~1~~C	0h	**Interrupt Vector Mask Clear (IVMC):** This field is bit significant. If a '1' is written to a bit, then the corresponding interrupt vector is unmasked. Writing a '0' to a bit has no effect. When read, this field returns the current interrupt mask value within the controller (not the value of this register). If a bit has a value of a '1', then the corresponding interrupt vector is masked, If a bit has a value of '0', then the corresponding interrupt vector is not masked.

3.1.5 Offset 14h: CC – Controller Configuration

This register modifies settings for the controller. Host software shall set the Arbitration Mechanism (CC.AMS), the Memory Page Size (CC.MPS), and the Command Set (CC.CSS) to valid values prior to enabling the controller by setting CC.EN to '1'. Attempting to create an I/O queue before initializing the I/O Completion Queue Entry Size (CC.IOCQES) and the I/O Submission Queue Entry Size (CC.IOSQES) should cause a controller to abort a Create I/O Completion Queue command or a Create I/O Submission Queue command~~s~~ with a status code of Invalid Queue Size.

Bit	Type	Reset	Description
31:24	RO	0	Reserved

Bit	Type	Reset	Description
23:20	RW	0	**I/O Completion Queue Entry Size (IOCQES):** This field defines the I/O Completion Queue entry size that is used for the selected I/O Command Set. The required and maximum values for this field are specified in the CQES field in the Identify Controller data structure in Figure 111 for each I/O Command Set. The value is in bytes and is specified as a power of two (2^*n*). If any I/O Completion Queues exist, then write operations that change the value in this field produce undefined results.
19:16	RW	0	**I/O Submission Queue Entry Size (IOSQES):** This field defines the I/O Submission Queue entry size that is used for the selected I/O Command Set. The required and maximum values for this field are specified in the SQES field in the Identify Controller data structure in Figure 111 for each I/O Command Set. The value is in bytes and is specified as a power of two (2^*n*). If any I/O Submission Queues exist, then write operations that change the value in this field produce undefined results.
…			
10:07	RW	0h	**Memory Page Size (MPS):** This field indicates the host memory page size. The memory page size is (2 ^ (12 + MPS)). Thus, the minimum host memory page size is 4 KiB and the maximum host memory page size is 128 MiB. The value set by host software shall be a supported value as indicated by the CAP.MPSMAX and CAP.MPSMIN fields. This field describes the value used for PRP entry size. This field shall only be modified when EN is cleared to '0'.
…			

3.1.6 Offset 1Ch: CSTS – Controller Status

…

Bit	Type	Reset	Description
…			
04	RW~~1~~C	HwInit	**NVM Subsystem Reset Occurred (NSSRO):** The initial value of this field is '1' if the last occurance of an NVM Subsystem Reset occured while power was applied to the NVM subsystem. The initial value of this field is '0' following an NVM Subsystem Reset due to application of power to the NVM subsystem. This field is only valid if the controller supports the NVM Subsystem Reset feature defined in section 7.3.1 as indicated by CAP.NSSRS set to '1'. The reset value of this field is '0' if an NVM Subsystem Reset causes activation of a new firmware image.
…			
00	RO	0	**Ready (RDY):** This field is set to '1' when the controller is ready to accept Submission Queue Tail doorbell writes after CC.EN is set to '1'. This field shall be cleared to '0' when CC.EN is cleared to '0' once the controller is ready to be re-enabled. Commands shall not be submitted to the controller until this field is set to '1' after the CC.EN bit is set to '1'. Failure to follow this requirement produces undefined results. Host software shall wait a minimum of CAP.TO seconds for this field to be set to '1' after setting CC.EN to '1' from a previous value of '0'.

…

3.1.11 Offset 38h: CMBLOC – Controller Memory Buffer Location

This optional register defines the location of the Controller Memory Buffer (refer to section 4.7). If CMBSZ (refer to section 3.1.12) is 0, then this register is reserved.

Bit	Type	Reset	Description
31:12	RO	Impl Spec	**Offset (OFST):** Indicates the offset of the Controller Memory Buffer in multiples of the Size Unit specified in CMBSZ. ~~This value shall be 4KB aligned.~~
11:03	RO	0h	Reserved
02:00	RO	Impl Spec	**Base Indicator Register (BIR):** Indicates the Base Address Register (BAR) that contains the Controller Memory Buffer. For a 64-bit BAR, the BAR for the lower 32-bits of the address is specified. Values 0h, 2h, 3h, 4h, and 5h are valid. The address specified by the bar shall be 4 KiB aligned.

3.1.12 Offset 3Ch: CMBSZ – Controller Memory Buffer Size

This optional register defines the size of the Controller Memory Buffer (refer to section 4.7). If the controller does not support the Controller Memory Buffer feature, then this register shall be cleared to 0h.

Bit	Type	Reset	Description
…			
11:08	RO	Impl Spec	**Size Units (SZU):** Indicates the granularity of the Size field. 0h: 4 KiB 1h: 64 KiB 2h: 1 MiB 3h: 16 MiB 4h: 256 MiB 5h: 4 GiB 6h: 64 GiB 7h – Fh: Reserved
…			

3.1.13 Offset 40h: BPINFO – Boot Partition Information

This optional register defines the characteristics of Boot Partitions (refer to section 8.13). If the controller does not support the Boot Partitions feature, then this register shall be cleared to 0h.

Bit	Type	Reset	Description
…			
14:00	RO	Impl Spec	**Boot Partition Size (BPSZ):** This field defines the size of each Boot Partition in multiples of 128 KiB. Both Boot Partitions are the same size.

3.1.14 Offset 44h: BPRSEL – Boot Partition Read Select

…

Bit	Type	Reset	Description
…			
29:10	RW	0h	**Boot Partition Read Offset (BPROF):** This field selects the offset into the Boot Partition, in 4 KiB units, that the controller copies into the Boot Partition Memory Buffer.
09:00	RW	0h	**Boot Partition Read Size (BPRSZ):** This field selects the read size in multiples of 4 KiB to copy into the Boot Partition Memory Buffer.

…

Modify a portion of section 3.2 (Index/Data Pair registers) as follows:

3.2 Index/Data Pair registers (Optional)

Index/Data Pair registers provide host software with a mechanism to access the NVM Express memory mapped registers using I/O space based registers. If supported, these registers are located in BAR2. On PC based platforms, host software (BIOS, Option ROMs, OSes) written to operate in 'real-mode' (8086 mode) are unable to access registers in a PCI Express function's address space, if the address space is memory mapped and mapped above 1 MiB.

The Index/Data Pair mechanism allows host software to access all of the memory mapped NVM Express registers using indirect I/O addressing in lieu of direct memory mapped access.

Note: UEFI drivers do not encounter the 1 MiB limitation, and thus when using EFI there is not a need for the Index/Data Pair mechanism. Thus, this feature is optional for the controller to support and may be obsoleted as UEFI becomes pervasive.

Modify a portion of section 4.1 (Submission Queue & Completion Queue Definition) as follows:

4.1 Submission Queue & Completion Queue Definition

Sections 4.1, 4.1.1 and 4.1.2 apply to NVMe over PCIe implementations only. For NVMe over Fabrics implementations, refer to section~~s~~ 2.4~~, 2.4.1 and 2.4.2~~ and the subsections of that section in the NVMe over Fabrics revision 1.0 specification.

…

Modify a portion of section 4.2 (Submission Queue Entry) as follows:

4.2 Submission Queue Entry – Command Format

…

Figure 11: Command Format – Admin and NVM Command Set

Bytes	Description
…	
23:16	**Metadata Pointer (MPTR):** This field is valid only if the command has metadata that is not interleaved with the logical block data, as specified in the Format NVM command. This is a reserved field in NVMe over Fabrics. If CDW0.PSDT (refer to Figure 10) is set to 00b, then this field shall contain the address of a contiguous physical buffer of metadata and that address shall be Dword aligned (i.e., bits 1:0 cleared to 00b). The controller is not required to check that bits 1:0 are cleared to 00b. The controller may report an error of Invalid Field in Command if bits 1:0 are not cleared to 00b. If the controller does not report an error of Invalid Field in Command, then the controller shall operate as if bits 1:0 are cleared to 00b. If CDW0.PSDT is set to 01b, then this field shall contain the address of a contiguous physical buffer of metadata and that address may be aligned on any byte boundary. If CDW0.PSDT is set to 10b, then this field shall contain the address of an SGL segment that contains exactly one SGL Descriptor. The address of that SGL segment shall be Qword aligned (i.e., bits 2:0 cleared to 000b). The SGL Descriptor contained in that SGL segment is the first SGL Descriptor of the metadata for the command. If the SGL Descriptor contained in that SGL segment is an SGL Data Block descriptor, then that SGL Data Block Descriptor is the only SGL Descriptor and therefore describes the entire metadata data transfer. Refer to section 4.4. The controller is not required to check that bits 2:0 are cleared to 000b. The controller may report an error of Invalid Field in Command if bits 2:0 are not cleared to 000b. If the controller does not report an error of Invalid Field in Command, then the controller shall operate as if bits 2:0 are cleared to 000b.
…	

…

Modify a portion of section 4.3 (Physical Region Page Entry and List) as follows:

4.3 Physical Region Page Entry and List

…

Figure 14: PRP Entry – Page Base Address and Offset

Bit	Description
63:00	**Page Base Address and Offset (PBAO):** This field indicates the 64-bit physical memory page address. The lower bits (*n*:0) of this field indicate the offset within the memory page~~.~~ ~~I~~ (e.g., if the memory page size is 4 KiB, then bits 11:00 form the Offset; if the memory page size is 8 KiB, then bits 12:00 form the Offset~~), etc~~. If this entry is not the first PRP entry in the command or a PRP List pointer in a command, then the Offset portion of this field shall be cleared to 0h. The Offset shall be Dword aligned, indicated by bits 1:0 being cleared to 00b. NOTE: The controller is not required to check that bits 1:0 are cleared to 00b. The controller may report an error of PRP Offset Invalid if bits 1:0 are not cleared to 00b. If the controller does not report an error of PRP Offset Invalid, then the controller shall operate as if bits 1:0 are cleared to 00b.

…

Modify a portion of section 4.4 (Scatter Gather List) as follows:

4.4 Scatter Gather List (SGL)

…

A Keyed SGL Data Block descriptor is a Data Block descriptor that includes a key that is used as part of the host memory access. The maximum length that may be specified in a Keyed SGL Data Block descriptor is (16 MiB – 1).

…

The SGL Descriptor Type field defined in Figure 18 specifies the SGL descriptor type. If the SGL Descriptor Type field is set to a reserved value or an unsupported value, then the SGL descriptor shall be processed as having an SGL Descriptor Type error. If the SGL Descriptor Sub Type field is set to a reserved value or an unsupported value, then the descriptor shall be processed as having an SGL Descriptor Type error.

…

Figure 20: SGL Data Block descriptor

Bytes	Description
07:00	**Address:** If the SGL Identifier Descriptor Sub Type field is cleared to 0h, then the Address field specifies the starting 64-bit memory byte address of the data block. If the SGL Identifier Descriptor Sub Type field is set to 1h, then the Address field contains an offset from the beginning of the location where data may be transferred. If the controller requires dword alignment and granularity as ~~specified~~ indicated in the SGL Support (SGLS) field of the Identify Controller data structure (refer to Figure 111), then the lower two bits shall be cleared to 00b. If dword alignment and granularity is required, the controller may report an error of Invalid Field in Command if bits 1:0 are not cleared to 00b. If the controller does not report an error of Invalid Field in Command, then the controller shall operate as if bits 1:0 are cleared to 00b.
11:08	**Length:** The Length field specifies the length in bytes of the data block. A Length field cleared to ~~0000000~~0h specifies that no data is transferred. An SGL Data Block descriptor specifying that no data is transferred is a valid SGL Data Block descriptor. If the controller requires dword alignment and granularity as specified in the SGL Support (SGLS) field of the Identify Controller data structure, then the lower two bits shall be cleared to 00b. If dword alignment and granularity is required, the controller may report an error of Invalid Field in Command if bits 1:0 are not cleared to 00b. If the controller does not report an error of Invalid Field in Command, then the controller shall operate as if bits 1:0 are cleared to 00b. If the value in the Address field plus the value in the Length field is greater than 1_00000000_00000000h, then the SGL Data Block descriptor shall be processed as having a Data SGL Length Invalid or Metadata SGL Length Invalid error.
…	

…

4.4.1 SGL Example

Figure 25 shows an example of a data read request using SGLs. In the example, the logical block size is 512B. The total length of the logical blocks accessed is 13 KiB, of which only 11 KiB is transferred to the host. The Number of Logical Blocks (NLB) field in the command shall specify 26, indicating the total length of the logical blocks accessed on the controller is 13 KiB. There are three SGL segments describing the locations in memory where the logical block data is transferred.

The three SGL segments contain a total of three Data Block descriptors with lengths of 3 KiB, 4 KiB and 4 KiB respectively. Segment 1 of the Destination SGL contains a Bit Bucket descriptor with a length of 2 KiB that specifies to not transfer (i.e., ignore) 2 KiB of logical block data from the NVM. Segment 1 of the destination SGL also contains a Last Segment descriptor specifying that the segment pointed to by the descriptor is the last SGL segment.

Modify a portion of section 4.6 (Completion Queue Entry) as follows:

4.6 Completion Queue Entry

…

4.6.1.2.1 Generic Command Status Definition

Completion queue entries with a Status Code type of Generic Command Status indicate a status value associated with the command that is generic across many different types of commands.

Figure 31: Status Code – Generic Command Status Values

Value	Description
…	
06h	**Internal Error:** The command was not completed successfully due to an internal error. Details on the internal device error ~~are available to~~ should be reported as an asynchronous event. Refer to Figure 47 for Internal Error Asynchronous Event Information.
07h	**Command Abort Requested:** The command was aborted due to a Command Abort command being received that specified the Submission Queue Identifier and Command Identifier of this command or as a result of a controller reset (refer to section 5.2).
…	
0Ah	**Command Aborted due to Missing Fused Command:** The fused command was aborted due to the adjacent submission queue entry not containing a fused command that is the other command in a supported fused operation (refer to section 6.2). ~~The Submission Queue does not contain the first command followed by the second command for a Fused Operation (refer to Figure 10).~~
…	
1Ah	**Keep Alive Timeout Invalid:** The Keep Alive Timeout value specified is invalid. This may be due to an attempt to specify a value of 0h on a transport that requires the Keep Alive feature to be enabled. This may be due to the value specified being too large for the associated NVMe Transport as defined in the NVMe Transport binding specification.
…	

…

4.6.1.2.2 Command Specific Errors Definition

…

Figure 33: Status Code – Command Specific Status Values

Value	Description	Commands Affected
…		
0Bh	Firmware Activation Requires Conventional Reset	Firmware Commit, Sanitize
…		
10h	Firmware Activation Requires NVM Subsystem Reset	Firmware Commit, Sanitize
11h	Firmware Activation Requires Controller Level Reset	Firmware Commit, Sanitize
…		

…

Modify a portion of section 4.7 (Controller Memory Buffer) as follows:

4.7 Controller Memory Buffer

…

The address region allocated for the CMB shall be 4 KiB aligned. It is recommended that a controller allocate the CMB on an 8 KiB boundary. The controller shall support burst transactions up to the maximum payload size, support byte enables, and arbitrary byte alignment.

…

Modify a portion of section 4.10 (Fused Operations) as follows:

4.10 Fused Operations

Fused operations enable a more complex command by "fusing" together two simpler commands. This feature is optional; support for this feature is indicated in FUSES field in the Identify Controller data structure in Figure 111. In a fused operation, the requirements are:

…

Whether a command is part of a fused operation is indicated in the Fused Operation (FUSE) field of Command Dword 0 shown in Figure 10. The ~~Fused Operation~~ FUSE field also indicates whether ~~this~~ each command is the first command in the fused operation or the second command in the fused operation. If the FUSE field is set to a non-zero value and the controller does not support the requested fused operation, then the controller should abort the command with a status of Invalid Field in Command.

Modify a portion of section 4.11 (Command Arbitration) as follows:

4.11 Command Arbitration

…

4.11.2 Weighted Round Robin with Urgent Priority Class Arbitration

…

The lowest strict priority class is the Weighted Round Robin class. This class consists of the three weighted round robin priority levels (High, Medium, and Low) that share the remaining bandwidth using weighted round robin arbitration. Host software controls the weights for the High, Medium, and Low service classes via the Set Features command. Round robin is used to arbitrate within multiple Submission Queues assigned to the same weighted round robin level. The number of candidate commands that may start processing from each Submission Queue per round is either the Arbitration Burst setting or the remaining weighted round robin credits, whichever is smaller.

…

Modify a portion of section 5 (Admin Command Set) as follows:

5 Admin Command Set

The Admin Command Set defines the commands that may be submitted to the Admin Submission Queue.

The Submission Queue Entry (SQE) structure and the fields that are common to all Admin commands are defined in section 4.2. The Completion Queue Entry (CQE) structure and the fields that are common to all Admin commands are defined in section 4.6. The command specific fields in the SQE and CQE structures (i.e., SQE Command Dwords 10-15 and CQE Dword 0) for the Admin Command Set are defined in this section.

~~For all Admin commands, Dword 14 and 15 are I/O Command Set specific.~~

Admin commands should not be impacted **…**

…

Figure 41: Opcodes for Admin Commands

Opcode by Field			Combined Opcode[2]	O/M[1]	Namespace Identifier Used[3]	Command
(07)	(06:02)	(01:00)				
Generic Command	Function	Data Transfer[4]				
…						
0b	000 01b	10b	06h	M	~~Yes~~ NOTE 8	Identify
…						

NOTES:
1. O/M definition: O = Optional, M = Mandatory.
2. Opcodes not listed are reserved.
3. A subset of commands use~~s~~ the Namespace Identifier (NSID) field ~~(CDW1.NSID)~~. If the Namespace Identifier field is used, then the value FFFFFFFFh is supported in this field unless otherwise indicated in footnote~~s 6~~ in this figure ~~indicates~~ that a specific command does not support that value or supports that value only under specific conditions. When this field is not used, the field is cleared to 0h as described in Figure 11.
4. Indicates the data transfer direction of the command. All options to the command shall transfer data as specified or transfer no data. All commands, including vendor specific commands, shall follow this convention: 00b = no data transfer; 01b = host to controller; 10b = controller to host; 11b = bidirectional.
5. For NVMe over PCIe implementations, the Keep Alive command is optional. For NVMe over Fabrics implementations, the associated NVMe Transport binding defines whether the Keep Alive command is optional or mandatory.
6. This command does not support the use of the Namespace Identifier (NSID) field ~~(CDW1.NSID)~~ set to FFFFFFFFh.
7. Support for the Namespace Identifier field set to FFFFFFFFh ~~is~~ depend~~sant~~ on the Directive Operation (refer to section 9).
8. Use of the Namespace Identifier field depends on the CNS value in the Identify Command as described in Figure 108.

Figure 42 defines Admin commands that are specific to the NVM Command Set.

Figure 42: Opcodes for Admin Commands – NVM Command Set Specific

Opcode (07)	Opcode (06:02)	Opcode (01:00)	Opcode[2]	O/M[1]	Namespace Identifier Used[3]	Command
Generic Command	Function	Data Transfer[4]				
…						

NOTES:
1. O/M definition: O = Optional, M = Mandatory.
2. Opcodes not listed are reserved.
3. A subset of commands use~~s~~ the Namespace Identifier (NSID) field ~~(CDW1.NSID)~~. If the Namespace Identifier field is used, then unless otherwise specified, the value FFFFFFFFh is supported in this field. When this field is not used, the field is cleared to 0h as described in Figure 11.
4. Indicates the data transfer direction of the command. All options to the command shall transfer data as specified or transfer no data. All commands, including vendor specific commands, shall follow this convention: 00b = no data transfer; 01b = host to controller; 10b = controller to host; 11b = bidirectional.
5. The use of the Namespace Identifier is Security Protocol specific.

Modify a portion of section 5.1 (Abort command) as follows:

5.1 Abort command

…

The Abort Command Limit field in the Identify Controller data structure (refer to Figure 111) indicates the controller limit on concurrent execution of Abort commands. A host should not allow the number of outstanding

Abort commands to exceed this value. The controller may complete any excess Abort commands with Abort Command Limit Exceeded status.

…

Modify a portion of section 5.2 (Asynchronous Event Request command) as follows:

5.2 Asynchronous Event Request command

…

The Asynchronous Event Request command is submitted by host software to enable the reporting of asynchronous events from the controller. This command has no timeout. The controller posts a completion queue entry for this command when there is an asynchronous event to report to the host. If Asynchronous Event Request commands are outstanding when the controller is reset, then each of those commands ~~are~~ is aborted and should not return a CQE.

…

The following event types are defined:

a) Error event: Indicates a general error that is not associated with a specific command (refer to Figure 47). To clear this event, host software reads the Error Information log (refer to section 5.14.1.1) using the Get Log Page command with the Retain Asynchronous Event field cleared to '0';

b) SMART / Health Status event: Indicates a SMART or health status event (refer to Figure 48). To clear this event, host software reads the SMART / Health Information log (refer to section 5.14.1.2) using the Get Log Page command with the Retain Asynchronous Event field cleared to '0'. The SMART / Health conditions that trigger asynchronous events may be configured in the Asynchronous Event Configuration feature using the Set Features command (see section 5.21);

c) Notice event: Indicates a general event (refer to Figure). To clear this event, host software reads the appropriate log page as described in Figure 49. The conditions that trigger asynchronous events may be configured in the Asynchronous Event Configuration feature using the Set Features command (see section 5.21.1.11). These notice events include:
   A. Namespace Attribute Changed;
   B. Firmware Activation Starting; and
   C. Telemetry Log Changed;

d) ~~I/O~~ NVM Command Set Specific events: Events that are defined by an I/O command set:
   ~~A. NVM Command Set Events:~~
   A. Reservation Log Page Available event: Indicates that one or more Reservation Notification log pages (refer to section 5.14.1.9.2) are available. To clear this event, host software reads the Reservation Notification log page using the Get Log Page command with the Retain Asynchronous Event field cleared to '0'; and
   B. Sanitize Operation Completed event: Indicates that a sanitize operation has completed and status is available in the Sanitize Status log page (refer to section 5.14.1.9.2). To clear this event, host software reads the Sanitize Status log page using the Get Log Page command with the Retain Asynchronous Event field cleared to '0';

and

e) Vendor Specific event: Indicates a vendor specific event. To clear this event, host software reads the indicated vendor specific log page using the Get Log Page command with the Retain Asynchronous Event field cleared to '0'.

Asynchronous events are reported due to a new entry being added to a log page (e.g., Error Information log) or a status update (e.g., status in the SMART / Health log). A status change may be permanent (e.g., the media has become read only) or transient (e.g., the temperature reached or exceeded a threshold for a period of time). Host software should modify the event threshold or mask the event for transient and permanent status changes before issuing another Asynchronous Event Request command to avoid repeated reporting of asynchronous events.

If ~~the controller needs to report~~ an event occurs for which reporting is enabled and there are no ~~outstanding~~ Asynchronous Event Request commands outstanding, the controller should retain the event information for ~~send a single notification of~~ that Asynchronous Event Type ~~when~~ and use that information as a response to the next Asynchronous Event Request command that is received. If a Get Log Page command clears the event prior to receiving the Asynchronous Event Request command or if a power off condition occurs, then a notification is not sent. If multiple events of the same type occur that have identical responses to the Asynchronous Event Request command, then those events may be reported as a single response to an Asynchronous Event Request command. If multiple events occur that are of different types, then the controller should retain a queue of those events for reporting in responses to subsequent Asynchronous Event Request commands.

5.2.1 Command Completion

…

Figure 46: Asynchronous Event Request – Completion Queue Entry Dword 0

Bit	Description
31:24	Reserved
23:16	**Log Page Identifier:** Indicates the log page associated with the asynchronous event. This log page needs to be read by the host to clear the event.
15:08	**Asynchronous Event Information:** Refer to Figure 47, Figure 48, Figure 49, and Figure 50 for detailed information regarding the asynchronous event.
07:03	Reserved
02:00	**Asynchronous Event Type:** Indicates the type of the asynchronous event. More specific information on the event is provided in the Asynchronous Event Information field.

Value	Definition	Reference
0h	Error status	Figure 47
1h	SMART / Health status	Figure 48
2h	Notice	Figure 49
3h ~~-~~ to 5h	Reserved	
6h	I/O Command Set specific status	Figure 50
7h	Vendor specific	

…

Figure 48: Asynchronous Event Information – SMART / Health Status

Value	Description
00h	**NVM subsystem Reliability:** NVM subsystem reliability has been compromised. This may be due to significant media errors, an internal error, the media being placed in read only mode, or a volatile memory backup device failing.
01h	**Temperature Threshold:** A temperature is ~~above~~ greater than or equal to an over temperature threshold or ~~below~~ less than or equal to an under temperature threshold (refer to section 5.21.1.4).
02h	**Spare Below Threshold:** Available spare capacity has fallen below the threshold.
03h - FFh	Reserved

Figure 49: Asynchronous Event Information – Notice

Value	Description
00h	**Namespace Attribute Changed:** The Identify Namespace data structure (refer to Figure 109) for one or more namespaces, as well as the Namespace List returned when the Identify command is issued with the CNS field set to 02h, have changed. Host software may use this event as an indication that it should read the Identify Namespace data structures for each namespace to determine what has changed. Alternatively, host software may request the Changed Namespace List (Log Identifier 04h) (refer to section 5.14.1.4) to determine which namespaces in this controller have changed information in the Identify Namespace ~~information~~ data structure since the last time the log page was read. A controller shall not send this event when Namespace Utilization has changed, as this is a frequent event that does not require action by the host. A controller shall only send this event for changes to the Format Progress Indicator field when bits 6:0 of that field transition from a non-zero value to zero, or from a zero value to a non-zero value.
01h	**Firmware Activation Starting:** The controller is starting a firmware activation process during which command processing is paused. Host software may use CSTS.PP to determine when command processing has resumed. To clear this event, host software reads the Firmware Slot Information log page.
02h	**Telemetry Log Changed:** The controller has saved the controller internal state in the Telemetry Controller-Initiated log page and set the Telemetry Controller-Initiated Data Available field to 1h in that log page. To clear this event, the host issues a Get Log Page command with Retain Asynchronous Event cleared to '0' for the Telemetry Controller-Initiated Log.
…	

…

Modify a portion of section 5.3 (Create I/O Completion Queue command) as follows:

5.3 Create I/O Completion Queue command

…

Figure 53: Create I/O Completion Queue – Command Dword 11

Bit	Description
31:16	**Interrupt Vector (IV):** This field indicates interrupt vector to use for this Completion Queue. This corresponds to the MSI-X or multiple message MSI vector to use. If using single message MSI or pin-based interrupts, then this field shall be cleared to 0h. In MSI-X, a maximum of ~~2K~~2,048 vectors are used. This value shall not be set to a value greater than the number of messages the controller supports (refer to MSICAP.MC.MME or MSIXCAP.MXC.TS). If the value is greater than the number of messages the controller supports the controller should return an error of Invalid Interrupt Vector.
…	

5.3.1 Command Completion

…

Figure 54: Create I/O Completion Queue – Command Specific Status Values

Value	Description
1h	**Invalid Queue Identifier:** The creation of the I/O Completion Queue failed due to an invalid queue identifier specified as part of the command. An invalid queue identifier is one that identifies the Admin Queue (i.e., 0h), is outside the range supported by the controller, or is a Completion Queue Identifier that is already in use.
2h	**Invalid Queue Size:** The host attempted to create an I/O Completion Queue: • with an invalid number of entries (e.g., a value of zero or a value which exceeds the maximum supported by the controller, specified in CAP.MQES); or • before initializing the CC.IOCQES field.
8h	**Invalid Interrupt Vector:** The creation of the I/O Completion Queue failed due to an invalid interrupt vector specified as part of the command.

Modify a portion of section 5.4 (Create I/O Submission Queue command) as follows:

5.4 Create I/O Submission Queue command

…

5.4.1 Command Completion

…

Figure 58: Create I/O Submission Queue – Command Specific Status Values

Value	Description
…	
2h	**Invalid Queue Size:** The host attempted to create an I/O ~~Completion~~ Submission Queue: • with an invalid number of entries (e.g., a value of zero or a value which exceeds the maximum supported by the controller, specified in CAP.MQES); or • before initializing the CC.IOSQES field.

Modify a portion of section 5.8 (Device Self-test command) as shown below:

5.8 Device Self-test command

…

The processing of a Device Self-test command and interactions with a device self-test operation already in progress is defined in Figure 68.

Figure 68: Device Self-test – Command Processing

Self-test in Progress[1]	Self-test Code value in new Drive Self-test command	Controller Action
Yes	1h – Short device self-test	Abort the new Device Self-test command with status Device Self-test in Progress.
	2h – Extended device self-test	
	Eh – Vendor specific	Vendor specific
	Fh – Abort device self-test	~~If bit 0 is in the Device Self-test Options (DSTO) of the Identify Controller data structure is:~~ ~~a) cleared to '0', or~~ ~~b) set to '1' and the new Device Self-test command was received on the same controller that that the self-test operation is already in progress on,~~ ~~then, the~~ The controller takes the following actions in order: 1. Abort device self-test operation in progress. 2. Create log entry in the Newest Self-test Result Data Structure in the Device Self-test Log. 3. Set the Current Device Self-test Status field in the Device Self-test Log to 0h. 4. Completes command successfully.
…		

NOTES:

1. If bit 0 is cleared to '0' in the Device Self-test Options (DSTO) of the Identify Controller data structure (refer to Figure 111), then the Self-test in Progress column represents that a device self-test operation is in progress on the controller that the new Device Self-test command was received on. If bit 0 is set to '1' in the Device Self-test Options (DSTO) of the Identify Controller data structure, then the Self-test in Progress column represents that a device self-test operation is in progress on the NVM subsystem.

Modify a portion of section 5.11 (Firmware Commit command) as follows:

5.11 Firmware Commit command

NOTE: This command was known in NVM Express revision 1.0 and 1.1 as "Firmware Activate."

The Firmware Commit command is used to modify the firmware image or Boot Partitions.

When modifying a firmware image, the Firmware Commit command verifies that a valid firmware image has been downloaded and commits that revision to a specific firmware slot. The host may select the firmware image to activate on the next Controller Level Reset as part of this command. The host may determine the currently executing firmware revision ~~may be determined from~~ by examining the Firmware Revision field ~~of~~ in the Identify Controller data structure in Figure 111. The host may determine the firmware revision to be executed on the next Controller Level Reset by examining ~~or as indicated in~~ the Firmware Slot Information log page. All controllers in the NVM subsystem share firmware ~~image~~ slots and the same firmware image is applied to all controllers.

When modifying Boot Partitions, the host may select the Boot Partition to mark as active or replace. A Boot Partition ~~may~~ is only able to be written when it is unlocked (refer to 8.13).

…

Figure 76: Firmware Commit – Command Dword 10

Bit	Description
…	
05:03	**Commit Action (CA):** This field specifies the action that is taken (refer to section 8.1) on the image downloaded with the Firmware Image Download command or on a previously downloaded and placed image. The actions are indicated in the following table. **Value** / **Definition** 000b: Downloaded image replaces the existing image, if any, in the specified Firmware Slot. The newly placed image is not activated. 001b: Downloaded image replaces the existing image, if any, in the specified Firmware Slot. The newly placed image is activated at the next Controller Level Rreset. 010b: The existing image in the specified Firmware Slot is activated at the next Controller Level Rreset. 011b: Downloaded image replaces the existing image, if any, in the specified Firmware Slot and is then activated immediately. If there is not a newly downloaded image, then the existing image in the specified firmware slot is activated immediately. 100-101b: Reserved 110b: Downloaded image replaces the Boot Partition specified by the Boot Partition ID field. 111b: Mark the Boot Partition specified in the BPID field as active and update BPINFO.ABPID.
02:00	**Firmware Slot (FS):** Specifies the firmware slot that shall be used for the Commit Action, if applicable. If the value specified is 0h, then the controller shall choose the firmware slot (i.e., slot 1 – to 7) to use for the operation.

5.11.1 Command Completion

Upon completion of the Firmware Commit command ~~When the command is completed~~, the controller posts a completion queue entry to the Admin Completion Queue indicating the status for the command.

For Firmware Commit commands ~~Requests~~ that specify activation of a new firmware image at the next Controller Level Rreset (i.e., the CA field was set to 001b or 010b) and ~~return~~ complete with a status code value of 00h (i.e., Success Completion), a~~ny~~ Controller Level Reset initiated by any of the methods defined in section 7.3.2 activates the specified firmware.

Firmware Commit command specific status values are defined in Figure 77.

Figure 77: Firmware Commit – Command Specific Status Values

Value	Description
…	
11h	**Firmware Activation Requires Controller Level Reset:** The firmware commit was successful; however, the image specified does not support being activated without a Controller Level Rreset. The image shall be activated at the next Controller Level Rreset. This status code should be returned only if the Commit Action field in the Firmware Commit command is set to 011b (i.e., activate immediately).
12h	**Firmware Activation Requires Maximum Time Violation:** The image specified if activated immediately, would exceed the Maximum Time for Firmware Activation (MTFA) value reported in the Identify Controller data structure (refer to Figure 111). To activate the firmware, the Firmware Commit command needs to be re-issued and the image activated using a reset.
…	

…

Modify a portion of section 5.13 (Get Features command) as follows:

5.13 Get Features command

…

Figure 84 describes the Feature Identifiers whose attributes may be retrieved using the Get Features command. The definition of the attributes returned and the associated format is specified in the section indicated.

…

Modify a portion of section 5.14 (Get Log Page command) as follows:

5.14 Get Log Page command

…

Figure 88: Get Log Page – Command Dword 12

Bit	Description
31:00	**Log Page Offset Lower (LPOL):** The log page offset specifies the location within a log page to start returning data from. This field specifies the lower 32 bits of the log page offset. The offset shall be dword aligned, indicated by bits 1:0 being cleared to 00b. The controller is not required to check that bits 1:0 are cleared to 00b. The controller may report an error of Invalid Field in Command if bits 1:0 are not cleared to 00b. If the controller does not report an error of Invalid Field in Command, then the controller shall operate as if bits 1:0 are cleared to 00b. If the host specifies an offset (i.e., LPOL and LPOU) that is greater than the size of the log page requested (e.g., a log page containing 100 bytes is requested starting at offset 200), then the controller shall abort the command with a status of Invalid Field in Command.

…

5.14.1.1 Error Information (Log Identifier 01h)

…

Each entry in the log page returned is defined in Figure 92. The log page is a set of 64-byte entries; the maximum number of entries supported is indicated in the ELPE field in the Identify Controller data structure ~~in~~ (refer to Figure 111). If the log page is full when a new entry is generated, the controller should insert the new entry into the log and discard the oldest entry.

…

5.14.1.2 SMART / Health Information (Log Identifier 02h)

…

Figure 93: Get Log Page – SMART / Health Information Log

<table>
<tr><td>00</td><td>Critical Warning: This field indicates critical warnings for the state of the controller. Each bit corresponds to a critical warning type; multiple bits may be set. If a bit is cleared to ‘0’, then that critical warning does not apply. Critical warnings may result in an asynchronous event notification to the host. Bits in this field represent the current associated state and are not persistent.

<table>
<tr><th>Bit</th><th>Definition</th></tr>
<tr><td>0</td><td>If set to ‘1’, then the available spare capacity has fallen below the threshold.</td></tr>
<tr><td>1</td><td>If set to ‘1’, then a temperature is:
a) greater than or equal to <s>above</s> an over temperature threshold; or
b) less than or equal to <s>below</s> an under temperature threshold,
(refer to section 5.21.1.4).</td></tr>
<tr><td>2</td><td>If set to ‘1’, then the NVM subsystem reliability has been degraded due to significant media related errors or any internal error that degrades NVM subsystem reliability.</td></tr>
<tr><td>3</td><td>If set to ‘1’, then the media has been placed in read only mode.</td></tr>
<tr><td>4</td><td>If set to ‘1’, then the volatile memory backup device has failed. This field is only valid if the controller has a volatile memory backup solution.</td></tr>
<tr><td>7:5</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>…</td><td></td></tr>
<tr><td>47:32</td><td>Data Units Read: Contains the number of 512 byte data units the host has read from the controller; this value does not include metadata. This value is reported in thousands (i.e., a value of 1 corresponds to 1000 units of 512 bytes read) and is rounded up (e.g, one indicates the that number of 512 byte data units read is from 1 to 1000, three indicates that the number of 512 byte data units read is from 2001 to 3000). When the LBA size is a value other than 512 bytes, the controller shall convert the amount of data read to 512 byte units.

For the NVM command set, logical blocks read as part of Compare operations and Read operations shall be included in this value.

A value of 0h in this field indicates that the number of Data Units Read is not reported.</td></tr>
<tr><td>63:48</td><td>Data Units Written: Contains the number of 512 byte data units the host has written to the controller; this value does not include metadata. This value is reported in thousands (i.e., a value of 1 corresponds to 1000 units of 512 bytes written) and is rounded up (e.g,one indicates that the number of 512 byte data units written is from 1 to 1000, three indicates that the number of 512 byte data units written is from 2001 to 3000). When the LBA size is a value other than 512 bytes, the controller shall convert the amount of data written to 512 byte units.

For the NVM command set, logical blocks written as part of Write operations shall be included in this value. Write Uncorrectable commands shall not impact this value.

A value of 0h in this field indicates that the number of Data Units Written is not reported.</td></tr>
<tr><td>79:64</td><td>Host Read Commands: Contains the number of read commands completed by the controller.

For the NVM command set, this value is the sum of the number of Compare commands and the number of Read commands.</td></tr>
<tr><td>…</td><td></td></tr>
<tr><td>175:160</td><td>Media and Data Integrity Errors: Contains the number of occurrences where the controller detected an unrecovered data integrity error. Errors such as uncorrectable ECC, CRC checksum failure, or LBA tag mismatch are included in this field. Errors introduced as a result of a Write Uncorrectable command (refer to section 6.15) may or may not be included in this field.</td></tr>
<tr><td>…</td><td></td></tr>
</table>

…

5.14.1.4 Changed Namespace List (Log Identifier 04h)

This log page is used to describe namespaces attached to this controller that have:

a) changed information in their Identify Namespace ~~information~~ data structure since the last time the log page was read;
b) been added; and
c) been deleted.

…

5.14.1.7 Telemetry Host-Initiated (Log Identifier 07h)

This log consists of a header describing the log and zero or more Telemetry Data Blocks (refer to section 8.14). All Telemetry Data Blocks are 512 bytes in size. The controller shall initiate a capture of the controller's internal controller state to this log when the controller processes a Get Log Page command for this log with the Create Telemetry Host-Initiated Data bit set to '1' in the Log Specific Field. If the host specifies a Log Page Offset Lower value that is not a multiple of 512 bytes in the Get Log Page command for this log, then the controller shall return an error of Invalid Field in Command. This log page is global to the controller.

Figure 100: Command Dword 10 – Log Specific Field

Bit	Description
11:09	Reserved
08	**Create Telemetry Host-Initiated Data:** If set to '1' then the controller shall capture the Telemetry Host-Initiated Data representing the internal state of the controller at the time the associated Get Log Page command is processed. If cleared to '0' then the controller shall not update the Telemetry Host Initiated Data. The Host-Initiated Data shall not change until the controller processes: a) a subsequent Telemetry Host-Initiated Log with this bit set to '1'~~,~~; b) a Firmware Commit command~~,~~; or c) a power on reset.

…

5.14.1.8 Telemetry Controller-Initiated (Log Identifier 08h)

…

Figure 102: Get Log Page – Telemetry Controller-Initiated Log (Log Identifier 08h)

Bytes	Description
…	
382	**Telemetry Controller-Initiated Data Available:** If this field is cleared to 0h, the log does not contain saved internal controller state. If this field is set to 1h, the log contains saved internal controller state. If this field is set to 1h, it shall not be cleared to 0h until a Get Log Page command with Retain Asynchronous Event cleared to '0' for the Telemetry Controller-Initiated Log completes successfully. This value is persistent across power states and reset. Other values are reserved.
…	

…

Modify a portion of section 5.15 (Identify command) as follows:

5.15 Identify command

5.15.1 Identify command overview

…

Figure 107: Identify – Command Dword 10

Bit	Description
31:16	**Controller Identifier (CNTID):** This field specifies the controller identifier used as part of some Identify operations. Whether the CNTID field is used for a particular Identify operation is indicated in Figure 108. If ~~the~~ this field is not used as part of the Identify operation, then: • host software shall clear this field to 0h for backwards compatibility (0h is a valid controller identifier); and • the controller shall ignore this field. Controllers that support the Namespace Management capability (refer to section 8.12) shall support this field.
15:08	Reserved
07:00	**Controller or Namespace Structure (CNS):** This field specifies the information to be returned to the host. Refer to Figure 108.

…

Figure 108: Identify – CNS Values

CNS Value	O/M [1]	Definition	NSID[2]	CNTID[3]	Reference Section
…					
01h	M	Identify Controller data structure for the controller processing the command.	N	N	5.15.3
…					
12h	O [4]	Controller ~~identifier l~~List of controllers attached to the specified NSID.	Y	Y	5.15.8
13h	O [4]	Controller ~~identifier l~~List of controllers that exist in the NVM subsystem.	N	Y	5.15.9
…					

NOTES:
1. O/M definition: O = Optional, M = Mandatory.
2. The ~~CDW1.~~NSID field is used: Y = Yes, N = No.
3. The CDW10.CNTID field is used: Y = Yes, N = No.
4. Mandatory for controllers that support the Namespace Management capability (refer to section 8.12).
5. Mandatory for controllers that support Virtualization Enhancements (refer to section 8.5).

5.15.2 Identify Namespace data structure (CNS 00h)

The Identify Namespace data structure (refer to Figure 109) is returned to the host for the namespace specified in the Namespace Identifier (~~CDW1.~~NSID) field if it is an active NSID. If the specified namespace is ~~not~~ an inactive NSID, then the controller returns a zero filled data structure.

If the controller supports the Namespace Management capability (refer to section 8.12) and ~~CDW1.~~ the NSID field is set to FFFFFFFFh, then the controller returns an Identify Namespace data structure that specifies capabilities that are common across namespaces for this controller. If the controller does not support the Namespace Management capability and ~~CDW1.~~ the NSID field is set to FFFFFFFFh, then the controller shall fail the command with a status code of Invalid Namespace or Format.

Figure 109: Identify – Identify Namespace Data Structure, NVM Command Set Specific

Bytes	O/M [1]	Description
…		
24	M	**Namespace Features (NSFEAT):** This field defines features of the namespace. Bits 7:4 are reserved. Bit 3 if set to '1' indicates that the ~~non-zero~~ value in the NGUID field for this namespace, if non-zero, is never reused by the controller and ~~non-zero~~ that the value in the EUI64 field~~s~~ for this namespace, if non-zero, is ~~are~~ never reused by the controller. If cleared to '0', then the NGUID value may be reused and the EUI64 value~~s~~ 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 7.11. …
25	M	**Number of LBA Formats (NLBAF):** This field defines the number of supported LBA … It is recommended that software and controllers transition to an LBA size that is 4 KiB or larger for ECC efficiency at the controller. If providing metadata, it is recommended that at least 8 bytes are provided per logical block to enable use with end-to-end data protection, refer to section 8.2.
…		
30	O	**Namespace Multi-path I/O and Namespace Sharing Capabilities (NMIC):** This field specifies multi-path I/O and namespace sharing capabilities of the namespace. Bits 7:1 are reserved. Bit 0: If set to '1', then the namespace may be attached to two or more controllers in the NVM subsystem concurrently (i.e., may be a shared namespace). If cleared to '0', then the namespace is a private namespace and ~~may only~~ is able to be attached to only one controller at a time.
…		
33	O	**Deallocate Logical Block Features (DLFEAT):** This field indicates information about features that affect deallocating logical blocks for this namespace. … Bits 2:0 indicate deallocated logical block read behavior. For a logical block that is deallocated, this field indicates the values read from ~~a~~ that deallocated logical block and its metadata (excluding protection information). The values for this field have the following meanings: **Value** / **Definition** 000b: The read behavior is ~~N~~not reported 001b: A deallocated logical block returns all bytes cleared to 00h 010b: A deallocated logical block returns all bytes set to FFh 011b to 111b: Reserved
…		

The LBA format data structure is described in Figure 110.

Figure 110: Identify – LBA Format Data Structure, NVM Command Set Specific

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>31:26</td><td>Reserved</td></tr>
<tr><td>25:24</td><td>Relative Performance (RP): This field indicates the relative performance of the LBA format indicated relative to other LBA formats supported by the controller. Depending on the size of the LBA and associated metadata, there may be performance implications. The performance analysis is based on better performance on a queue depth 32 with 4 KiB read workload. The meanings of the values indicated are included in the following table.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>00b</td><td>Best performance</td></tr>
<tr><td>01b</td><td>Better performance</td></tr>
<tr><td>10b</td><td>Good performance</td></tr>
<tr><td>11b</td><td>Degraded performance</td></tr>
</table></td></tr>
<tr><td>...</td><td></td></tr>
</table>

5.15.3 Identify Controller data structure (CNS 01h)

The Identify Controller data structure (refer to Figure 111) is returned to the host for this controller.

Figure 111: Identify – Identify Controller Data Structure

Bytes	O/M [1]	Description
...		
77	M	**Maximum Data Transfer Size (MDTS):** This field indicates the maximum data transfer size for a command that transfers data between memory accessible by the host (e.g., host memory, Controller Memory Buffer (refer to section 4.7)) and the controller. The host should not submit a command that exceeds this maximum data transfer size. If a command is submitted that exceeds ~~the~~ this transfer size, then the command is aborted with a status of Invalid Field in Command. The value is in units of the minimum memory page size (CAP.MPSMIN) and is reported as a power of two (2^*n*). A value of 0h indicates that there is no ~~restrictions on~~ maximum data transfer size. Thi**s** ~~e restriction~~ field includes the length of metadata, if ~~it~~ metadata is interleaved with the logical block data. ~~The restriction~~ This field does not apply to commands that do not transfer data between memory accessible by the host and the controller (e.g., the Write Uncorrectable command ~~or~~ and the Write Zeroes command); there is no maximum data transfer size for those commands. If SGL Bit Bucket descriptors are supported, their lengths shall be included in determining if a command exceeds the Maximum Data Transfer Size for destination data buffers. Their length in a source data buffer is not included for a Maximum Data Transfer Size calculation.
...		
260	M	**Firmware Updates (FRMW):** This field indicates capabilities regarding firmware updates. Refer to section 8.1 for more information on the firmware update process. Bits 7:5 are reserved. Bit 4 if set to '1' indicates that the controller supports firmware activation without a reset. If cleared to '0', then the controller requires a reset for firmware to be activated. Bits 3:1 indicate the number of firmware slots that the controller supports. This field shall specify a value ~~between~~ from one ~~and~~ to seven, indicating that at least one firmware slot is supported and up to seven maximum. This corresponds to firmware slots 1 through 7. Bit 0 if set to '1' indicates that the first firmware slot (i.e., slot 1) is read only. If cleared to '0', then the first firmware slot (i.e., slot 1) is read/write. Implementations may choose to have a baseline read only firmware image.

Bytes	O/M [1]	Description
261	M	**Log Page Attributes (LPA):** This field indicates optional attributes for log pages that are accessed via the Get Log Page command. Bits 7:4 are reserved. Bit 3 if set to '1', then the controller supports the Telemetry Host-Initiated and Telemetry Controller-Initiated log pages and sending Telemetry Log Notices. If cleared to '0', then the controller does not support the Telemetry Host-Initiated and Telemetry Controller-Initiated log pages and Telemetry Log Notice events. Bit 2 if set to '1', then the controller supports extended data for the Get Log Page command (including extended Number of Dwords and Log Page Offset fields). Bit 2 if cleared to '0', then the controller does not support extended data for the Get Log Page command. Bit 1 if set to '1', then the controller supports the Commands Supported and Effects log page. Bit 1 if cleared to '0', then the controller does not support the Commands Supported and Effects log page. Bit 0 if set to '1', then the controller supports the SMART / Health information log page on a per namespace basis. If cleared to '0', then the controller does not support the SMART / Health information log page on a per namespace basis.
...		
267:266	M	**Warning Composite Temperature Threshold (WCTEMP):** This field indicates the minimum Composite Temperature field value (reported in the SMART / Health Information log in Figure 93) that indicates an overheating condition during which controller operation continues. Immediate remediation is recommended (e.g., additional cooling or workload reduction). The platform should strive to maintain a composite temperature ~~below~~ less than this value. A value of 0h in this field indicates that no warning temperature threshold value is reported by the controller. Implementations compliant to revision 1.2 or later of this specification shall report a non-zero value in this field. It is recommended that implementations report a value of 0157h in this field.
269:268	M	**Critical Composite Temperature Threshold (CCTEMP):** This field indicates the minimum Composite Temperature field value (reported in the SMART / Health Information log in Figure) that indicates a critical overheating condition (e.g., may prevent continued normal operation, possibility of data loss, automatic device shutdown, extreme performance throttling, or permanent damage). A value of 0h in this field indicates that no critical temperature threshold value is reported by the controller. Implementations compliant to revision 1.2 or later of this specification shall report a non-zero value in this field.
...		
275:272	O	**Host Memory Buffer Preferred Size (HMPRE):** This field indicates the preferred size that the host is requested to allocate for the Host Memory Buffer feature in 4 KiB units. This value shall be ~~larger~~ greater than or equal to the Host Memory Buffer Minimum Size. If this field is non-zero, then the Host Memory Buffer feature is supported. If this field is cleared to 0h, then the Host Memory Buffer feature is not supported.
279:276	O	**Host Memory Buffer Minimum Size (HMMIN):** This field indicates the minimum size that the host is requested to allocate for the Host Memory Buffer feature in 4 KiB units. If this field is cleared to 0h, then the host is requested to allocate any amount of host memory possible up to the HMPRE value.
...		

<table>
<tr><th>Bytes</th><th>O/M [1]</th><th>Description</th></tr>
<tr><td>315:312</td><td>O</td><td>Replay Protected Memory Block Support (RPMBS): This field indicates if the controller supports one or more Replay Protected Memory Blocks (RPMBs) and the capabilities. Refer to section 8.10.

<table>
<tr><th>Bits</th><th>Description</th></tr>
<tr><td>…</td><td></td></tr>
<tr><td>23:16</td><td>Total Size: If the Number of RPMB Units field is non-zero, then this field indicates the number of 128 KiB units of data in each RPMB supported in the controller. This is a 0’s based value. A value of 0h indicates support for one unit of 128 KiB of data.

If the Number of RPMB Units field is 0h, then this field shall be ignored.</td></tr>
<tr><td>…</td><td></td></tr>
</table>
</td></tr>
<tr><td>…</td><td></td><td></td></tr>
<tr><td>319</td><td>M</td><td>Firmware Update Granularity (FWUG): This field indicates the granularity and

…

The value is reported in 4 KiB units (e.g., 1h corresponds to 4 KiB, 2h corresponds to 8 KiB). A value of 0h indicates that no information on granularity is provided. A value of FFh indicates there is no restriction (i.e., any granularity and alignment in Dwords is allowed).</td></tr>
<tr><td>321:320</td><td>M</td><td>Keep Alive Support (KAS): This field indicates the granularity of the Keep Alive Timer in 100 ms units (refer to section 7.12). If this field is cleared to 0h, then the Keep Alive feature is not supported. The Keep Alive feature shall be supported for NVMe over Fabrics implementations as described in section 7.12.</td></tr>
<tr><td>…</td><td></td><td></td></tr>
<tr><td>512</td><td>M</td><td>Submission Queue Entry Size (SQES): This field defines the required and maximum Submission Queue entry size when using the NVM Command Set.

Bits 7:4 define the maximum Submission Queue entry size when using the NVM Command Set. This value is <s>larger</s> greater than or equal to the required SQ entry size (i.e., bits 3:0 in this field). The value is in bytes and is reported as a power of two (2^n). The recommended value is 6, corresponding to a standard NVM Command Set SQ entry size of 64 bytes. Controllers that implement proprietary extensions may support a larger value.

Bits 3:0 define the required (i.e., minimum) Submission Queue Entry size when using the NVM Command Set. This is the minimum entry size that may be used. The value is in bytes and is reported as a power of two (2^n). The required value shall be 6, corresponding to 64.</td></tr>
<tr><td>513</td><td>M</td><td>Completion Queue Entry Size (CQES): This field defines the required and maximum Completion Queue entry size when using the NVM Command Set.

Bits 7:4 define the maximum Completion Queue entry size when using the NVM Command Set. This value is <s>larger</s> greater than or equal to the required CQ entry size (i.e., bits 3:0 in this field). The value is in bytes and is reported as a power of two (2^n). The recommended value is 4, corresponding to a standard NVM Command Set CQ entry size of 16 bytes. Controllers that implement proprietary extensions may support a larger value.

Bits 3:0 define the required (i.e., minimum) Completion Queue entry size when using the NVM Command Set. This is the minimum entry size that may be used. The value is in bytes and is reported as a power of two (2^n). The required value shall be 4, corresponding to 16.</td></tr>
<tr><td>…</td><td></td><td></td></tr>
</table>

Bytes	O/M [1]	Description
525	M	**Volatile Write Cache (VWC):** This field indicates attributes related to the presence of a volatile write cache in the controller. Bits 7:1 are reserved. Bit 0 if set to '1' indicates that a volatile write cache is present. If cleared to '0', a volatile write cache is not present. If a volatile write cache is present, then the host controls whether the volatile write cache is enabled with a Set Features command specifying the Volatile Write Cache feature identifier (refer to section 5.21.1.6). The Flush command (refer to section 6.8) is used to request that the contents of a volatile write cache be made non-volatile.
527:526	M	**Atomic Write Unit Normal (AWUN):** This field indicates the size of the write **…** A value of FFFFh indicates all commands are atomic as this is the largest command size. It is recommended that implementations support a minimum of 128 KiB (appropriately scaled based on LBA size).
…		
533:532	O	**Atomic Compare & Write Unit (ACWU):** This field indicates the size of the write operation guaranteed to be written atomically to the NVM across all namespaces with any supported namespace format for a Compare and Write fused operation. If a specific namespace guarantees a larger size than is reported in this field, then ~~this~~ the Atomic Compare & Write Unit size for that namespace ~~specific size~~ is reported in the NACWU field in the Identify Namespace data structure. Refer to section 6.4. This field shall be supported if the Compare and Write fused command is supported. This field is specified in logical blocks and is a 0's based value. If a Compare and Write is submitted that requests a transfer size larger than this value, then the controller may fail the command with a status code of Invalid Field in Command. If Compare and Write is not a supported fused command, then this field shall be 0h.
…		

…

Figure 112: Identify – Power State Descriptor Data Structure

…	
149:144	Reserved
143:128	**Idle Power (IDLP):** This field indicates the typical power consumed by the NVM subsystem over 30 seconds in this power state when idle (i.e., there are no pending commands, register accesses, background processes, sanitize operation, nor device self-test operations). The measurement starts after the NVM subsystem has been idle for 10 seconds. The power in Watts is equal to the value in this field multiplied by the scale indicated in the Idle Power Scale field. A value of 0000h indicates Idle Power is not reported. Refer to section 8.4. Note: This value may be used by hosts to manage power versus resume latency. Platform and form factor specifications may have additional power measurement and reporting requirements that are outside the scope of this specification.
127:125	Reserved
…	
23:16	Reserved

15:00	**Maximum Power (MP):** This field indicates the sustained maximum power consumed by the NVM subsystem in this power state. The power in Watts is equal to the value in this field multiplied by the scale specified in the Max Power Scale field. A value of 0h indicates Maximum Power is not reported. Refer to section 8.4. Note: This value is intended to provide an approximate guideline for hosts to manage power versus performance. Platform and form factor specifications may have additional power measurement and reporting requirements that are outside the scope of this specification.

…

5.15.4 Active Namespace ID list (CNS 02h)

A list of 1,024 namespace IDs is returned to the host containing active NSIDs in increasing order that are greater than the value specified in the Namespace Identifier (~~CDW1.~~NSID) field of the command. The controller should abort the command with status code Invalid Namespace or Format if ~~CDW1.~~ the NSID field is set to FFFFFFFEh or FFFFFFFFh. The ~~CDW1.~~ NSID field may be cleared to 0h to retrieve a Namespace List including the namespace starting with NSID of 1h. The data structure returned is a Namespace List (refer to section 4.8).

5.15.5 Namespace Identification Descriptor list (CNS 03h)

A list of Namespace Identification Descriptor structures (refer to Figure 113) is returned to the host for the namespace specified in the Namespace Identifier (~~CDW1.~~NSID) field if it is an active NSID. If the NSID field does not specify an active NSID, then refer to section 6.1.5 for the status code to return.

The controller may return any number of variable length Namespace Identification Descriptor structures that fit into the 4096 byte Identify payload.

…

5.15.6 Allocated Namespace ID list (CNS 10h)

A list of up to 1,024 namespace IDs is returned to the host containing allocated NSIDs in increasing order that are greater than the value specified in the Namespace Identifier (~~CDW1.~~NSID) field of the Identify command.

The controller should abort the command with status code Invalid Namespace or Format if ~~CDW1.~~ the NSID field is set to FFFFFFFEh or FFFFFFFFh. The ~~CDW1.~~NSID field may be cleared to 0h to retrieve a Namespace List including the namespace starting with NSID of 1h. The data structure returned is a Namespace List (refer to section 4.8).

5.15.7 Identify Allocated Namespace data structure (CNS 11h)

The Identify Namespace data structure (refer to Figure 109) is returned to the host for the namespace specified in the Namespace Identifier (~~CDW1.~~NSID) field if it is an allocated NSID. If the specified namespace is an unallocated NSID, then the controller returns a zero filled data structure.

If the specified namespace is an invalid NSID, then the controller shall fail the command with a status code of Invalid Namespace or Format. If ~~CDW1.~~ the NSID field is set to FFFFFFFFh, then the controller should fail the command with a status code of Invalid Namespace or Format.

5.15.8 Namespace Attached Controller list (CNS 12h)

A Controller List (refer to section 4.9) of up to 2047 controller identifiers is returned containing a controller identifier greater than or equal to the value specified in the Controller Identifier (CDW10.CNTID) field. The list contains controller identifiers that are attached to the namespace specified in the Namespace Identifier (~~CDW1.~~NSID) field. If the NSID field is set to FFFFFFFFh, then the controller should fail the command with a status code of Invalid Field in Command.

5.15.9 Controller list (CNS 13h)

A Controller List (refer to section 4.9) of up to 2047 controller identifiers is returned containing a controller identifier greater than or equal to the value specified in the Controller Identifier (CDW10.CNTID) field. The list contains controller identifiers in the NVM subsystem that may or may not be attached to namespace(s).

…

Modify a portion of section 5.16 (Keep Alive command) as shown below:

5.16 Keep Alive command

…

5.16.1 Command Completion

Upon completion of the Keep Alive command, ~~If the command is completed, then~~ the controller shall post a completion queue entry to the Admin Completion Queue indicating the status for the command.

Modify a portion of section 5.20 (Namespace Management command) as shown below:

5.20 Namespace Management command

…

The Namespace Management command uses the Data Pointer and Dword 10 fields. All other command specific fields are reserved.

The Namespace Identifier (~~CDW1.~~NSID) field is used as follows for create and delete operations:

- Create: The ~~CDW1.~~NSID field is reserved for this operation; host software clears this field to a value of 0h. The controller shall select an available Namespace Identifier to use for the operation; or

…

Modify a portion of section 5.21 (Set Features command) as shown below:

5.21 Set Features command

…

5.21.1 Feature Specific Information

Figure 128 defines the Features that may be configured with a Set Features command and retrieved with a Get Features command. Figure 129 defines Features that are specific to the NVM Command Set. Some Features utilize a memory buffer to configure or return attributes for a Feature, whereas others only utilize a Dword in the command or completion queue entry. Feature values that are not persistent across power cycles and resets are restored to their default values as part of a controller reset operation. ~~The default value for each Feature is vendor specific and set by the manufacturer unless otherwise specified; it is not changeable.~~ For more information on Features, including default value definitions, saveable value definitions, and current value definitions, refer to section 7.8.

There may be commands in execution when a Feature is changed. The new settings may or may not apply to commands already submitted for execution when the Feature is changed. Any commands submitted to a Submission Queue after a Set Features command is successfully completed shall utilize the new settings for the associated Feature. To ensure that a Features values apply~~ies~~ to all subsequent commands, the host should allow commands being processed ~~should be~~ to complete~~d~~ prior to issuing the Set Features command.

If the controller does not support a changeable value for a Feature (e.g., the Feature is not changeable), and a Set Feature command for that Feature is processed, then if that command specifies a Feature value that:

- is not the same as the existing value for that Feature, then the controller shall abort that command with a status code of Feature Not Changeable; and
- is the same as the existing value for that Feature, then the controller may:
  - complete that command successfully; or
  - abort that command with a status code of Feature Not Changeable.

Figure 128: Set Features – Feature Identifiers

Feature Identifier	O/M[6]	Current Setting Persist~~sent~~ Across Power Cycle and Reset[2]	Uses Memory Buffer for Attributes	Description
…				
05h	M	No	No	Error Recovery
…				

NOTES:
1. The behavior of a controller in response to an inactive namespace ID to a vendor specific Feature Identifier is vendor specific.
2. This column is only valid if the feature is not saveable (refer to section 7.8). If the feature is saveable, then this column is not used. ~~and any feature may be configured to be saved across power cycles and reset.~~
3. The controller does not save settings for the Host Memory Buffer feature across power states and reset events, however, host software may restore the previous values. Refer to section 8.9.
4. The feature does not use a memory buffer for Set Features commands, but it does use a memory buffer for Get Features commands. Refer to section 8.9.
5. The feature is mandatory for NVMe over PCIe implementations. This feature is not supported for NVMe over Fabrics implementations.
6. O/M: O = Optional, M = Mandatory.

Figure 129: Set Features, NVM Command Set Specific – Feature Identifiers

Feature Identifier	O/M[4]	Current Setting Persist~~sent~~ Across Power Cycle and Reset[1]	Uses Memory Buffer for Attributes	Description
80h	O	Yes	No	Software Progress Marker
81h	O[2]	No	Yes	Host Identifier
82h	O[3]	No	No	Reservation Notification Mask
83h	O[3]	Yes	No	Reservation Persistance
84h – BFh				Reserved

NOTES:
1. This column is only valid if the feature is not saveable (refer to section 7.8). If the feature is saveable, then this column is not used. ~~and any feature may be configured to be saved across power cycles and reset.~~
2. Mandatory if reservations are supported as indicated in the Identify Controller data structure.
3. Mandatory if reservations are supported by the namespace as indicated by a non-zero value in the Reservation Capabilities (RESCAP) field in the Identify Namespace data structure.
4. O/M: O = Optional, M = Mandatory.

…

5.21.1.2 Power Management (Feature Identifier 02h)

This Feature allows the host to configure the power state. The attributes are ~~indicated~~ specified in Command Dword 11 (refer to Figure 131).

~~After a~~ Upon successful completion of a Set Features command for this feature, the controller shall be in the Power State specified. For a transition to a non-operational power state, the device may exceed the power indicated for that non-operational power state as defined in section 8.4.1 (e.g., while completing this command). If enabled, autonomous power state transitions continue to occur from the new state.

If a Get Features command is submitted for this Feature, the attributes ~~specified~~ described in Figure 131.a are returned in Dword 0 of the completion queue entry for that command.

Figure 131: Power Management – Command Dword 11

Bit	Description
31:08	Reserved
07:05	**Workload Hint (WH):** This field indicates the type of workload expected. This hint may be used by the NVM subsystem to optimize performance. Refer to section 8.4.3 for more details.
04:00	**Power State (PS):** This field indicates the new power state into which the controller is requested to ~~should~~ transition. This power state shall be one supported by the controller as indicated in the Number of Power States Supported (NPSS) field in the Identify Controller data structure. If the power state specified is not supported, the controller shall abort the command and should return an error of Invalid Field in Command.

Figure 131.a: Power Management – Completion Queue Entry Dword 0

Bit	Description
31:08	Reserved
07:05	**Workload Hint (WH):** This field indicates the type of workload. Refer to section 8.4.3 for more details.
04:00	**Power State (PS):** This field indicates the current power state of the controller, or the power state into which the controller is transitioning.

5.21.1.3 LBA Range Type (Feature Identifier 03h), (Optional)

This feature indicates the type and attributes of LBA ranges that are part of the specified namespace. If multiple Set Features commands for this feature are processed, only information from the most recent successful command is retained (i.e., subsequent commands replace information provided by previous commands).

A Set Features command with the Feature Identifier set to 03h and the NSID field set to FFFFFFFFh shall be aborted with a status of Invalid Field in Command.

…

Figure 133: LBA Range Type – ~~Dword 0 of command completion queue entry~~ Completion Queue Entry Dword 0

Bit	Description
31:06	Reserved
05:00	**Number of LBA Ranges (NUM):** This field indicates the number of valid LBA ranges returned in the data buffer for the command (refer to Figure 134). This is a 0's based value.

Each entry in the LBA Range Type data structure is defined in Figure 134. The LBA Range feature is a set of 64 byte entries; the number of entries is indicated as a command parameter, the maximum number of entries is 64. The controller is not required to perform validation checks on any of the fields in this data structure. The LBA ranges ~~shall~~ should not overlap and may be listed in any order (e.g., ordering by LBA is not required). If

the controller checks for LBA range~~s~~ overlap and the controller detects an LBA range overlap, then the controller should return an error of Overlapping Range.

For a Get Features command, the controller ~~shall~~ may clear to zero all unused entries in the LBA Range Type data structure. For a Set Features command, the controller shall ignore all unused entries in the LBA Range Type data structure.

…

5.21.1.4 Temperature Threshold (Feature Identifier 04h)

A controller may report up to nine temperature values in the SMART / Health Information log (i.e., the Composite Temperature and Temperature Sensor 1 through Temperature Sensor 8; refer to Figure 93). Associated with each implemented temperature sensor is an over temperature threshold and an under temperature threshold. When a temperature is greater than or equal to its corresponding over temperature threshold or less than or equal to its corresponding under temperature threshold, then bit one of the Critical Warning field in the SMART / Health Information Log (refer to section 5.14.1.2) is set to one. This may trigger an asynchronous event.

The over temperature threshold feature shall be implemented for Composite Temperature. The under temperature threshold Feature shall be implemented for Composite Temperature if a non-zero Warning Composite Temperature Threshold (WCTEMP) field value is reported in the Identify Controller data structure (~~in~~ refer to Figure 111). The over temperature threshold and under temperature threshold features shall be implemented for all implemented temperature sensors (i.e., all Temperature Sensor fields that report a non-zero value).

The default value of the over temperature threshold feature for Composite Temperature is the value in the Warning Composite Temperature Threshold (WCTEMP) field in the Identify Controller data structure if WCTEMP is non-zero; otherwise, ~~it~~ the default value is implementation specific. The default value of the under temperature threshold feature for Composite Temperature is implementation specific. The default value of the over temperature threshold for all implemented temperature sensors is FFFFh. The default value of the under temperature threshold for all implemented temperature sensors is 0h.

If a Get Features command is submitted for this feature, the temperature threshold selected by Command Dword 11 is returned in Dword 0 of the completion queue entry for that command.

…

5.21.1.5 Error Recovery (Feature Identifier 05h)

This Feature controls the error recovery attributes for the specified namespace. The attributes are indicated in Command Dword 11.

If a Get Features command is submitted for this Feature, the attributes ~~specified~~ described in Figure 136 are returned in Dword 0 of the completion queue entry for that command.

Figure 136: Error Recovery – Command Dword 11

Bit	Description
31:17	Reserved
16	**Deallocated or Unwritten Logical Block Error Enable (DULBE):** If set to '1', then the Deallocated or Unwritten Logical Block error is enabled for the namespace specified in ~~CDW1.~~ ~~the~~ NSID field. If cleared to '0', then the Deallocated or Unwritten Logical Block error is disabled for the namespace specified in ~~CDW1.~~ the NSID field. Host software shall only enable this error if it is supported for this namespace as indicated in the Namespace Features field of the Identify Namespace data structure. The default value for this field shall be '0'. Refer to section 6.7.1.1.
15:00	**Time Limited Error Recovery (TLER):** Indicates a limited retry timeout value in 100 millisecond units. This limit applies to I/O commands that support the Limited Retry bit and that are sent to the namespace for which this Feature has been set. The timeout starts when error recovery actions have started while processing the command. A value of 0h indicates that there is no timeout. Note: This mechanism is primarily intended for use by host software that may have alternate means of recovering the data.

5.21.1.6 Volatile Write Cache (Feature Identifier 06h), (Optional)

…

If a volatile write cache is not present, then a Set Features command specifying the Volatile Write Cache feature identifier shall fail with Invalid Field in Command status, and a Get Features command specifying the Volatile Write Cache feature identifier should fail with Invalid Field in Command status.

…

5.21.1.7 Number of Queues (Feature Identifier 07h)

This Feature indicates the number of queues that the host requests for this controller. This feature shall only be issued during initialization prior to creation of any I/O Submission and/or Completion Queues. If a Set Features command is issued for this feature after creation of any I/O Submission and/or I/O Completion Queues, then the Set Features command shall fail with status code of Command Sequence Error. The controller ~~value allocated~~ shall not change the value allocated between resets. For a Set Features command, the attributes are indicated in Command Dword 11(refer to Figure 138). For a Get Features command, Dword 11 is ignored.

…

Figure 139: Number of Queues – ~~Dword 0 of command completion queue entry~~ Completion Queue Entry Dword 0

Bit	Description
31:16	**Number of I/O Completion Queues Allocated (NCQA):** Indicates the number of I/O Completion Queues allocated by the controller. A minimum of one queue shall be allocated, reflecting that the minimum support is for one I/O Completion Queue. The value may not match the number requested by host software. This is a 0's based value.
15:00	**Number of I/O Submission Queues Allocated (NSQA):** Indicates the number of I/O Submission Queues allocated by the controller. A minimum of one queue shall be allocated, reflecting that the minimum support is for one I/O Submission Queue. The value may not match the number requested by host software. This is a 0's based value.

5.21.1.12 Autonomous Power State Transition (Feature Identifier 0Ch), (Optional)

…

Figure 145: Autonomous Power State Transition – Data Structure Entry

Bit	Description
63:32	Reserved
31:08	**Idle Time Prior to Transition (ITPT):** This field specifies the amount of idle time that occurs in this power state prior to transitioning to the Idle Transition Power State. The time is specified in milliseconds. A value of 0h disables the autonomous power state transition feature for this power state.
07:03	**Idle Transition Power State (ITPS):** This field specifies the power state ~~for~~ to which the controller ~~to~~ autonomously transitions, ~~to~~ after there is a continuous period of idle time in the current power state that exceeds the time specified in the Idle Time Prior to Transition (ITPT) field. If t~~T~~he ITPT field is set to a non-zero value, then the state specified ~~is required to~~ in this field shall be a non-operational state as described in Figure 112. This field should not specify a power state with higher reported idle power than the current power state. If the ITPT field is cleared to 0h, then this field should be cleared to 0h.
02:00	Reserved

…

5.21.1.13 Host Memory Buffer (Feature Identifier 0Dh), (Optional)

This Feature controls the Host Memory Buffer. The attributes are indicated in Command Dword 11, Command Dword 12, Command Dword 13, Command Dword 14, and Command Dword 15.

The Host Memory Buffer feature provides a mechanism for the host to allocate a portion of host memory for the exclusive use of the controller. After a successful completion of a Set Features command enabling the host memory buffer, the host shall not write to:

…

Figure 147: Host Memory Buffer – Command Dword 11

Bit	Description
31:02	Reserved
01	**Memory Return (MR):** If set to '1', then the host is returning memory previously allocated ~~memory~~ to the controller for use as the host memory buffer (HMB). That memory may have been in use~~d~~ for the HMB prior to a reset or entering the Runtime D3 state (e.g., prior to the HMB being disabled). A returned host memory buffer shall have the exact same size, descriptor list address, descriptor list contents, and host memory buffer contents as last seen by the controller before the host memory buffer was disabled (i.e., a Set Features command with the EHM bit cleared to '0' was processed). If cleared to '0', then the host is allocating host memory resources with undefined content.
00	**Enable Host Memory (EHM):** If set to '1', then the host memory buffer shall be enabled and the controller may use the host memory buffer. If cleared to '0', then the host memory buffer shall be disabled, and the controller shall not use the host memory buffer. If a Set Features command is processed with this bit cleared to '0', then the controller shall ignore Command Dword 12, Command Dword 13, Command Dword 14, and Command Dword 15.

…

Figure 153: Host Memory Buffer – Host Memory Buffer Descriptor Entry

Bit	Description
127:96	Reserved
95:64	**Buffer Size (BSIZE):** Indicates the number of contiguous memory page size (CC.MPS) units for this descriptor.
63:00	**Buffer Address (BADD):** Indicates the host memory address for this descriptor aligned to the memory page size (CC.MPS). The lower bits (*n*:0) of this field indicate the offset within the memory page is 0h~~.~~ ~~I~~ (e.g., if the memory page size is 4 KiB, then bits 11:00 shall be zero; if the memory page size is 8 KiB, then bits 12:00 shall be zero~~), etc~~.

…

Figure 154: Host Memory Buffer – ~~Dword 0 of command completion queue entry~~ Completion Queue Entry Dword 0

Bit	Description
31:01	Reserved
00	**Enable Host Memory (EHM):** If set to '1', then the host memory buffer is enabled and the controller may use the host memory buffer. If cleared to '0', then the host memory buffer is disabled, and the controller is not using the host memory buffer.

…

5.21.1.14 Timestamp (Feature Identifier 0Eh), (Optional)

The Timestamp feature enables the host to set a timestamp value in the controller. A controller indicates support for the Timestamp feature through the Optional NVM Command Support (ONCS) field in the Identify Controller data structure. The Timestamp value (refer to Figure 157) in a Set Features command sets a timestamp value in the controller. After the current value for this feature is set, the controller updates that value as time passes. A Get Features command that requests the current value reports the timestamp value in the controller at the time the Get Features command is processed (e.g., the value set with a Set Features command for the current value plus the elapsed time since being set).

Note: If the Timestamp feature supports a saveable value and the host sets a saveable value, then the timestamp value restored after a subsequent power on or reset event is the value that was saved (refer to section 7.8). As a result, it may appear as if the timestamp moves backwards in time.

The accuracy of Timestamp values after initialization may be affected by vendor specific factors, such as whether the controller continuously counts after the timestamp is initialized, or whether it stops counting during certain intervals (~~such as~~ e.g., non-operational power states). If the controller stops counting during such intervals, then the Synch bit in the Timestamp – Data Structure for Get Features (refer to Figure 157) shall be set to '1'.

If the controller maintains (i.e., continues to update) the timestamp value across any type of Controller Level Reset (e.g., across a Controller Reset), then the controller shall also preserve the Timestamp Origin field (refer to Figure 157) across that type of Controller Level Reset.

Timestamp values should not be used for security applications. The use of the Timestamp is ~~beyond~~ outside the scope of this specification.

If a Set Features command is issued for this Feature, the data structure specified in Figure 156 is transferred in the data buffer for that command, specifying the Timestamp value

…

5.21.1.17 Non-Operational Power State Config (Feature Identifier 11h), (Optional)

…

Figure 160: Non-Operational Power State Config – Command Dword 11

Bit	Description
31:01	Reserved

Figure 160: Non-Operational Power State Config – Command Dword 11

Bit	Description
00	**Non-Operational Power State Permissive Mode Enable (NOPPME):** If NOPPME is set to '1', then 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 (i.e., Non-Operational Power State Permissive Mode is enabled). If NOPPME is cleared to '0', then the controller shall not exceed the limits of any non-operational state while running controller initiated background operations in that state (i.e., Non-Operational Power State Permissive Mode is disabled). If Non-Operational Power State Permissive Mode is disabled, then: a) thermal management that requires power (e.g., cooling fans) may be disabled; and b) performance after resuming from the non-operational power state may be degraded until background activity that was not allowed while in that non-operational power state has completed. If the host attempts to set this bit to '1' and the controller does not support Non-Operational Power State Permissive Mode as indicated in the Controller Attributes (CTRATT) field of the Identify Controller data structure, then the controller shall abort the command ~~fails~~ with a status of Invalid Field in Command.

…

5.21.1.19.1 NVMe over PCIe Implementations

The Host Identifier is an optional feature in NVMe over PCIe implementations. The controller may support a 64-bit Host Identifier and/or an extended 128-bit Host Identifier. It is recommended that implementations support the extended 128-bit Host Identifier as indicated in the Controller Attributes field in the Identify Controller data structure. The Host Identifier may be modified at any time using a Set Features command causing the controller to be logically remapped from the original host associated with the old Host Identifier to a new host associated with the new Host Identifier.

…

5.21.1.20 Reservation Notification Mask (Feature Identifier 82h), (Optional[1])

…

A Set Features command that uses a namespace ID other than FFFFFFFFh modifies the reservation notification mask for the corresponding namespace only. A Set Features command that uses a namespace ID of FFFFFFFFh modifies the reservation notification mask of all namespaces that are attached to the controller and that support reservations. A Get Features command that uses a namespace ID other than FFFFFFFFh returns the reservation notification mask for the corresponding namespace. A Get Features command that uses a namespace ID of FFFFFFFFh ~~is~~ should be aborted with status Invalid Field in Command. If a Set Features command or a Get Features command attempts to access the Reservation Notification Mask on a namespace that does not support reservations or is invalid, then ~~the~~ that command is aborted with status Invalid Field in Command.

…

[1] Mandatory if reservations are supported by the namespace as indicated by a non-zero value in the Reservation Capabilities (RESCAP) field in the Identify Namespace data structure.

5.21.1.21 Reservation Persistence (Feature Identifier 83h), (Optional[2])

Each namespace that supports reservations has a Persist Through Power Loss (PTPL) state that may be modified using either a Set Features command or a Reservation Register command (refer to section 6.11). The Reservation Persistence feature attributes are indicated in Command Dword 11.

The PTPL state is contained in the Reservation Persistence Feature that is namespace specific. A Set Features command that uses the namespace ID FFFFFFFFh modifies the PTPL state associated with all namespaces that are attached to the controller and that support PTPL (i.e., support reservations). A Set Features command that uses a valid namespace ID other than FFFFFFFFh and corresponds to a namespace that supports reservations, modifies the PTPL state for that namespace. A Get Features command that uses a namespace ID of FFFFFFFFh ~~is~~ should be aborted with status Invalid Field in Command. A Get Features command that uses a valid namespace ID other than FFFFFFFFh and corresponds to a namespace that supports PTPL, returns the PTPL state for that namespace. If a Set Features command or a Get Features command using a namespace ID other than FFFFFFFFh attempts to access the PTPL state for a namespace that does not support this Feature Identifier, then the command is aborted with status Invalid Field in Command.

This Feature should not support a saveable value. If a saveable value is supported for this Feature, then the host should set the current value and the saveable value to the same value.

If a Get Features command successfully completes for this Feature Identifier, the attributes specified in Figure 165 are returned in Dword 0 of the completion queue entry for that command

…

5.21.2 Command Completion

Upon completion of the Set Features command, the controller posts a completion queue entry to the Admin Completion Queue. If a status of Successful Completion is returned, the completion queue entry shall not be posted until ~~A completion queue entry is posted to the Admin Completion Queue when~~ the controller has completed setting attributes associated with the Feature. Set Features command specific status values are defined in Figure 166.

Figure 166: Set Features – Command Specific Status Values

Value	Description
0Dh	**Feature Identifier Not Saveable:** The Feature Identifier specified does not support a saveable value.
0Eh	**Feature Not Changeable:** The Feature Identifier ~~is not able to be changed~~ specified does not support a changeable value.
0Fh	**Feature Not Namespace Specific:** The Feature Identifier specified is not namespace specific. The Feature Identifier settings apply across all namespaces.
14h	**Overlapping Range:** This error is indicated if the LBA Range Type data structure has overlapping ranges.

…

Modify a portion of section 5.23 (Format NVM command) as shown below:

5.23 Format NVM command – NVM Command Set Specific

…

As part of the Format NVM command, the host requests a format operation and may request a secure erase of the contents of the NVM (refer to the SES field in Figure 173). There are two types of secure erase. The User Data Erase erases all user content present in the NVM subsystem. The Cryptographic Erase erases all user

content present in the NVM subsystem by deleting the encryption key with which the user data was previously encrypted.

The scope of the format operation and the scope of the format with secure erase depend on the attributes that the controller supports for the Format NVM command and the Namespace Identifier specified in the command~~. The scope for the format operation is defined~~ as described in Figure 171. The ~~scope for~~ type of secure erase, if applicable, is based on the setting of the Secure Erase Settings field in Command Dword 10 ~~is~~ as defined in Figure ~~172~~173.

Figure 171: Format NVM ~~– Format~~ Operation Scope

FNA[1] ~~Bit 0~~[1]	NSID	Format Operation
0b	FFFFFFFFh	All namespaces attached to the controller. Other namespaces are not affected.
0b	Any valid value (refer to section 6.1.2)	Particular namespace specified. Other namespaces are not affected.
1b	Any valid value (refer to section 6.1.2) or FFFFFFFFh	All namespaces in the NVM subsystem
NOTES: 1. For a Format NVM command with Secure Erase, this column refers to bit 1 in the FNA field in the Identify Controller data structure (refer to Figure 111) and bit 0 in the FNA field is ignored. For a Format NVM command without Secure Erase, this column refers to bit 0 in the FNA field, and bit 1 in the FNA field is ignored.~~FNA is the Format NVM Attributes field in the Identify Controller data structure.~~		

~~**Figure 172: Format NVM – Secure Erase Scope**~~

~~FNA[1] Bit 1[2]~~	~~NSID~~	~~Secure Erase~~
~~0b~~	~~FFFFFFFFh~~	~~All namespaces attached to the controller~~
~~0b~~	~~Any valid value (refer to section 6.1.2)~~	~~Particular namespace specified~~
~~1b~~	~~Any valid value (refer to section 6.1.2) or FFFFFFFFh~~	~~All namespaces in the NVM subsystem~~
~~NOTES:~~ ~~1. FNA is the Format NVM Attributes field in the Identify Controller data structure.~~		

The Format NVM command shall fail if the controller is in an invalid security state (refer to the appropriate security specification, e.g., TCG Storage Interface Interactions Specification). The Format NVM command may fail if there are outstanding I/O commands to the namespace specified to be formatted. I/O commands for a namespace that has a Format NVM command in progress may ~~fail.~~ be aborted and if aborted, the controller should return a status code of Format in Progress.

For a Format command with ~~an~~ the NSID field set to **…**

…

After successful completion of a Format NVM command, the ~~The~~ settings specified in the Format NVM command (e.g., PI, MSET, LBAF) are reported as part of the Identify Namespace data structure. If the Format NVM command results in a change of the logical block size for the namespace, then the resulting namespace size (i.e., NSZE) (refer to Figure 109) and the namespace capacity (i.e., NCAP) (refer to Figure 109) may differ from the values indicated prior to the processing of the Format NVM command.

The Format NVM command uses the Command Dword 10 field. All other command specific fields are reserved.

Figure 173: Format NVM – Command Dword 10

<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>31:12</td><td>Reserved</td></tr>
<tr><td>11:09</td><td>Secure Erase Settings (SES): This field specifies whether a secure erase should be performed as part of the format and the type of the secure erase operation. The erase applies to all user data, regardless of location (e.g., within an exposed LBA, within a cache, within deallocated LBAs, etc.).

<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>000b</td><td>No secure erase operation requested</td></tr>
<tr><td>001b</td><td>User Data Erase: All user data shall be erased, contents of the user data after the erase is indeterminate (e.g., the user data may be zero filled, one filled, etc.). The controller may perform a cryptographic erase when a User Data Erase is requested if all user data is encrypted.</td></tr>
<tr><td>010b</td><td>Cryptographic Erase: All user data shall be erased cryptographically. This is accomplished by deleting the encryption key.</td></tr>
<tr><td>011b to 111b</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td>…</td><td></td></tr>
</table>

…

Modify a portion of section 5.24 (Sanitize command) as shown below:

5.24 Sanitize command – NVM Command Set Specific

…

If a firmware activation with reset is pending, then the controller shall abort any Sanitize command. ~~with a status of~~

If the Firmware Commit command that established the pending firmware activation with reset condition returned a status code of:

a) Firmware Activation Requires Controller Level Reset;
b) Firmware Activation Requires Conventional Reset; or
c) Firmware Activation Requires NVM Subsystem Reset,

then the controller should abort the Sanitize command with that same status code.

If the Firmware Commit command that established the pending firmware activation with reset condition completed successfully or returned a status code other than:

a) Firmware Activation Requires Controller Level Reset;
b) Firmware Activation Requires Conventional Reset; or
c) Firmware Activation Requires NVM Subsystem Reset,

then the controller should abort the Sanitize command with a status code of Firmware Activation Requires Controller Level Reset.

Activation of new firmware is prohibited during a sanitize operation (refer to section 8.15.1).

…

Modify a portion of section 5.25 (Security Receive command) as shown below:

5.25 Security Receive command – NVM Command Set Specific

…

Figure 179: Security Receive – Command Dword 10

Bit	Description
31:24	**Security Protocol (SECP):** This field specifies the security protocol as defined in SFSC. The controller shall ~~fail~~ abort the command with ~~Invalid Parameter indicated~~ status of Invalid Field in Command if an unsupported value of the Security Protocol is specified.
…	

…

Modify a portion of section 5.26 (Security Send command) as shown below:

5.26 Security Send command – NVM Command Set Specific

…

Figure 183: Security Send – Command Dword 10

Bit	Description
31:24	**Security Protocol (SECP):** This field specifies the security protocol as defined in SFSC. The controller shall ~~fail the command with Invalid Parameter indicated~~ abort the command with status Invalid Field in Command if a reserved value of the Security Protocol is specified.
…	

Modify a portion of section 6 (NVM Command Set) as shown below:

6 NVM Command Set

An NVM subsystem is comprised of some number of controllers, where each controller may access some number of namespaces, where each namespace is comprised of some number of logical blocks. A logical block is the smallest unit of data that may be read or written from the controller. The logical block data size, reported in bytes, is always a power of two. Logical block sizes may be 512 bytes, 1 KiB, 2 KiB, 4 KiB, 8 KiB, etc. Supported logical block sizes are reported in the Identify Namespace data structure.

…

Figure 185: Opcodes for NVM Commands

Opcode by Field			Combined Opcode[2]	O/M[1]	Command[3]	Reference Section
(07)	(06:02)	(01:00)				
Standard Command	Function	Data Transfer[5]				
…						

NOTES:
1. O/M definition: O = Optional, M = Mandatory.
2. Opcodes not listed are reserved.
3. All NVM commands use the Namespace Identifier ~~field~~ (~~CDW1.~~NSID) field.
4. Mandatory if reservations are supported as indicated in the Identify Controller data structure.
5. Indicates the data transfer direction of the command. All options to the command shall transfer data as specified or transfer no data. All commands, including vendor specific commands, shall follow this convention: 00b = no data transfer; 01b = host to controller; 10b = controller to host; 11b = bidirectional.

…

Modify a portion of section 6.1 (Namespaces) as shown below:

6.1 Namespaces

…

6.1.5 NSID and Namespace Relationships

Unless otherwise noted, specifying an inactive NSID in a command that uses the Namespace Identifier ~~field~~ (~~CDW1.~~NSID) field shall cause the controller to abort the command with status Invalid Field in Command. Specifying an invalid NSID in a command that uses the NSID field shall cause the controller to abort the command with status Invalid Namespace or Format.

…

6.1.6 NSID and Namespace Usage

…

To determine the active NSIDs for a particular controller, the host may follow either of the following methods:

1. Issue an Identify command with the CNS field cleared to 00h for each valid NSID (based on the Number of Namespaces value (i.e., NN field) in the Identify Controller data structure). If a non-zero data structure is returned for a particular NSID, then that is an active NSID; or
2. Issue an Identify command with a CNS field set to 02h to retrieve a list of up to 1,024 active NSIDs. If there are more than 1,024 active NSIDs, continue to issue Identify commands with a CNS field set to 02h until all active NSIDs are retrieved.

…

Modify a portion of section 6.4 (Atomic Operations) as shown below:

6.4 Atomic Operations

…

Figure 188: Atomicity Parameters

	Parameter Name	Value[1]
Controller Atomic Parameters (refer to the Identify Controller data structure in Figure 111)	Atomic Write Unit Normal (AWUN)	
	Atomic Write Unit Power Fail (AWUPF)	≤ AWUN
	Atomic Compare and Write Unit (ACWU)	
Namespace Atomic Parameters (refer to the Identify Namespace data structure in Figure 109)	Namespace Atomic Write Unit Normal (NAWUN)	≥ AWUN
	Namespace Atomic Write Unit Power Fail (NAWUPF)	≥ AWUPF ≤ NAWUN
	Namespace Atomic Compare and Write Unit (NACWU)	≥ ACWU
Namespace Atomic Boundary Parameters (refer to the Identify Namespace data structure in Figure 109)	Namespace Atomic Boundary Size Normal (NABSN)	≥ NAWUN
	Namespace Atomic Boundary Offset (NABO)	≤ NABSN ≤ NABSPF
	Namespace Atomic Boundary Size Power Fail (NABSPF)	≥ NAWUPF

NOTES:
1. When the parameter is supported, the value shall meet the listed condition(s).

…

An NVM subsystem may report per namespace values for these fields that are specific to the namespace format and are indicated in the Identify Namespace data structure (refer to Figure 109). If an NVM subsystem reports a per namespace value, it shall be greater than or equal to the corresponding baseline value indicated in the Identify Controller data structure (refer to Figure 111).

The values are reported in the fields (Namespace) Atomic Write Unit Normal, (Namespace) Atomic Write Unit Power Fail, and (Namespace) Atomic Compare & Write Unit in the Identify Controller data structure or the Identify Namespace data structure depending on whether the values are the baseline or namespace specific.

…

Modify a portion of section 6.6 (Compare Command) as shown below:

6.6 Compare command

The Compare command reads the logical blocks specified by the command from the medium and compares the data read to a comparison data buffer transferred as part of the command. If the data read from the controller and the comparison data buffer are equivalent with no miscompares, then the command completes successfully. If there is any miscompare, the command completes with an error of Compare Failure.

If metadata is provided, then a comparison is also performed for the metadata, excluding protection information. The command may specify protection information to be checked as described in ~~Refer to~~ section ~~8.3~~ 8.3.1.4.

…

Figure 197: Compare – Command Dword 12

Bit	Description
…	
30	**Force Unit Access (FUA):** ~~This field specifies that~~ If set to '1', then for data and metadata, if any, associated with logical blocks specified by the Compare command, the controller shall: 1) commit that data and metadata, if any, to non-volatile media; and 2) read the data and metadata, if any, ~~read shall be read~~ from non-volatile media. If cleared to '0', then this bit has no effect.
…	

Modify a portion of section 6.7 (Dataset Management command) as shown below:

6.7 Dataset Management command

…

The data that the Dataset Management command provides is a list of ranges with context attributes. Each range consists of a starting LBA, a length of logical blocks that the range consists of, and the context attributes to be applied to that range. The length in logical blocks field is a 1-based value. The definition of the Dataset Management command Range field is specified in Figure 204. The maximum case of 256 ranges is shown.

Figure 204: Dataset Management – Range Definition

Range	Byte	Field
Range 0	03:00	Context Attributes
	07:04	Length in logical blocks
	15:08	Starting LBA
Range 1	19:16	Context Attributes
	23:20	Length in logical blocks
	31:24	Starting LBA
…		
Range 255	4083:4080	Context Attributes
	4087:4084	Length in logical blocks
	4095:4088	Starting LBA

…

6.7.1.1 Deallocate

A logical block that has been deallocated using the Dataset Management command is no longer deallocated when the logical block is written. Read operations do not affect the deallocation status of a logical block. The value read from a deallocated logical block shall be deterministic; specifically, the value returned by subsequent reads of that logical block shall be the same until a write operation occurs to that logical block.

The values read from a deallocated logical block and its metadata (excluding protection information) shall be all bytes set to 00h (e.g., bits 2:0 in the DLFEAT field are set to 001b), all bytes set to FFh (e.g., bits 2:0 in the DLFEAT field are set to 010b), or the last data written to the associated logical block and its metadata, except that access is prohibited to all data and metadata values written before the most recent successful sanitize operation, if any. The Deallocate Logical Block Features (DLFEAT) field in the Identify Namespace data structure (refer to Figure 109) may report the values read from a deallocated logical block and its metadata.

The values read from a deallocated or unwritten logical block's protection information field shall:

- have the Guard field value set to FFFFh or set to the CRC for the value read from the deallocated logical block and its metadata (excluding protection information) (e.g., set to 0000h if the value read is all bytes set to 00h); and
- have the Application Tag field value set to FFFFh and the Reference Tag field value set to FFFFFFFFh (indicating the protection information shall not be checked).

…

Modify a portion of section 6.9 (Read command) as shown below:

6.9 Read command

…

Figure 210: Read – Command Dword 12

Bit	Description
…	
30	**Force Unit Access (FUA):** ~~This field indicates that~~ If set to '1', then for data and metadata, if any, associated with logical blocks specified by the Read command, the controller shall: 1) commit that data and metadata, if any, to non-volatile media; and 2) return the data, and metadata, if any, that are read ~~shall be returned~~ from non-volatile media. There is no implied ordering with other commands. If cleared to '0', then this bit has no effect.
…	

…

Modify a portion of section 6.11 (Reservation Register command) as shown below:

6.11 Reservation Register command

The Reservation Register command is used to register, unregister, or replace a reservation key.

The command uses Command Dword 10 and a Reservation Register data structure in memory (refer to Figure 221). If the command uses PRPs for the data transfer, then PRP Entry 1 and PRP Entry 2 fields are used. If the command uses SGLs for the data transfer, then the SGL Entry 1 field is used. All other command specific fields are reserved.

Figure 219: Reservation Register – Data Pointer

Bit	Description
127:00	**Data Pointer (DPTR):** This field specifies the location of a data buffer where data is transferred from. Refer to Figure 11 for the definition of this field.

Figure 220: Reservation Register – Command Dword 10

<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>31:30</td><td>Change Persist Through Power Loss State (CPTPL): This field allows the Persist Through Power Loss (PTPL) state associated with the namespace to be modified as a side effect of processing this command. If a saveable value is supported for the Reservation Persistence Feature (refer to section 5.21.1.21), then any change to the PTPL state as a result of processing this command shall be applied to both the current value and the saveable value of that feature.

<table>
<tr><th>CPTPL Value</th><th>Description</th></tr>
<tr><td>00b</td><td>No change to PTPL state</td></tr>
<tr><td>01b</td><td>Reserved</td></tr>
<tr><td>10b</td><td>Set PTPL state to ‘0’. Reservations are released and registrants are cleared on a power on.</td></tr>
<tr><td>11b</td><td>Set PTPL state to ‘1’. Reservations and registrants persist across a power loss.</td></tr>
</table>
</td></tr>
<tr><td>…</td><td></td></tr>
</table>

…

Modify a portion of section 6.14 (Write command) as shown below:

6.14 Write command

…

Figure 235: Write – Command Dword 12

Bit	Description
…	
30	**Force Unit Access (FUA):** ~~This field indicates that~~ If set to ‘1’, then for data and medadata, if any, associated with logical blocks specified by the Write command, the controller shall write that data and metadata, if any, ~~the data shall be written~~ to non-volatile media before indicating command completion. There is no implied ordering with other commands. If cleared to ‘0’, then this bit has no effect.
…	

…

Modify a portion of section 6.16 (Write Zeroes command) as shown below (changes shown here are based on changes published in ECN-003):

6.16 Write Zeroes command

The Write Zeroes command is used to set a range of logical blocks to zero. Non-PI related metadata for this command, if any, shall be all bytes set to 00h. The protection information for logical blocks written to the media is updated based on CDW12.PRINFO. If the Protection Information Action field (PRACT) is cleared to ‘0’, then the protection information for this command shall be all zeroes. If the Protection Information Action field (PRACT) is set to ‘1’, then the protection information shall be based on the End-to-end Data Protection Type Settings (DPS) field in the Identify Namespace data structure (refer to Figure 109) and the CDW15.EILBRT, CDW15.ELBATM, and CDW15.ELBAT fields in the Write Zeroes command.

After successful completion of this command, the value returned by subsequent reads of logical blocks in this range shall be all bytes set to 00h until a write occurs to this LBA range.

If the Deallocate bit (CDW12.DEAC) is set to '1' in a Write Zeroes command, and the namespace supports ~~setting~~ clearing all bytes to 00h in the values read (e.g., bits 2:0 in the DLFEAT field are set to 001b) from a deallocated logical block and its metadata (excluding protection information), then for each specified logical block, the controller:

- should deallocate ~~all~~ that logical block~~s in the range specified by that command~~;
- shall return all bytes cleared to 00h in the values read from:
  - that ~~the deallocated~~ logical block~~s;~~ and
  - ~~its~~ that logical blocks metadata (excluding protection information); and
- shall return the protection information in that ~~the deallocated~~ logical block~~s~~ as specified in section 6.7.1.1

If the Deallocate bit is cleared to '0' in a Write Zeroes command, and the namespace supports ~~setting~~ clearing all bytes to 00h in the values read (e.g., bits 2:0 in the DLFEAT field are set to 001b) from a deallocated logical block and its metadata (excluding protection information), then, for each specified logical block, the controller:

- may deallocate ~~any~~ that logical block~~s in the range specified by that command~~;
- shall return all bytes cleared to 00h in the values read from:
  - that ~~the deallocated~~ logical block~~s;~~ and
  - ~~its~~ that logical blocks metadata (excluding protection information); and
- shall return the protection information in that ~~the deallocated~~ logical block~~s~~ based on CDW12.PRINFO in that Write Zeroes command.

~~If the namespace does not support setting all bytes to 00h in the values read from a deallocated logical block and its metadata (excluding protection information), then the controller shall not deallocate any logical blocks in the range specified by a Write Zeroes command.~~

For each logical block in the range specified by a Write Zeroes command, if the namespace does not support that logical block clearing all bytes to 00h in the values read from that logical block and its metadata (excluding the protection information) read, then the controller shall not deallocate that logical block.

…

Figure 244: Write Zeroes – Command Dword 12

Bit	Description
…	
30	**Force Unit Access (FUA):** ~~This field indicates that~~ If set to '1', then the controller shall write the data, and metadata, if any, ~~shall be written~~ to non-volatile media before indicating command completion. There is no implied ordering with other commands. If cleared to '0', then this bit has no effect.
…	
25	**Deallocate (DEAC):** If set to '1', then the host is requesting that the controller ~~should~~ deallocate the specified logical blocks ~~and may write all bytes set to 00h~~. If cleared to '0', then ~~the controller may write all bytes set to 00h or may~~ the host is not requesting that the controller deallocate the specified logical blocks.
…	

…

6.16.1 Command Completion

Upon completion of the Write Zeroes command ~~When the command is completed with success or failure~~, the controller shall post a completion queue entry to the associated I/O Completion Queue indicating the status for the command.

…

Modify a portion of section 7.2 (Command Submission and Completion Mechanism) as shown below:

7.2 Command Submission and Completion Mechanism (Informative)

…

7.2.2 Basic Steps when Building a Command

…

d. The Namespace Identifier, ~~CDW1.~~NSID field, is set to the namespace the command applies to;

…

7.2.5 Command Examples

7.2.5.1 Creating an I/O Submission Queue

…

- ~~CDW1.~~ The NSID field is set to 0h; Submission Queues are not specific to a namespace;

…

7.2.5.2 Executing a Fused Operation

This example describes how host software creates and executes a fused command, specifically Compare and Write for a total of 16 KiB of data. In this case, there are two commands that are created. The first command is the Compare, referred to as CMD0. The second command is the Write, referred to as CMD1. In this case, end-to-end data protection is not enabled and the size of each logical block is 4 KiB.

…

The attributes of the Compare command are:

…

- CMD0.~~CDW1.~~NSID is set to identify the appropriate namespace;

…

- CMD0.CDW12.FUA is cleared to '0', indicating that the data may be read from any location, including a ~~DRAM~~ volatile cache, in the NVM subsystem;
- CMD0.CDW12.PRINFO is cleared to 0h since end-to-end protection is not enabled;
- CMD0.CDW12.NLB is set to 3h, indicating that four logical blocks of a size of 4 KiB each are to be compared against;

…

The attributes of the Write command are:

…

- CMD1.~~CDW1.~~NSID is set to identify the appropriate namespace. This value shall be the same as CMD0.~~CDW1.~~NSID;

…

- CMD1.CDW12.FUA is cleared to '0', indicating that the data may be written to any location, including a ~~DRAM~~ volatile cache, in the NVM subsystem;
- CMD1.CDW12.PRINFO is cleared to 0h since end-to-end protection is not enabled;
- CMD1.CDW12.NLB is set to 3h, indicating that four logical blocks of a size of 4 KiB each are to be compared against. This value shall be the same as CMD0.CDW12.NLB;

…

Modify a portion of section 7.3 (Resets) as shown below:

7.3 Resets

7.3.1 NVM Subsystem Reset

An NVM Subsystem Reset is initiated when:

- Main ~~P~~power is applied to the NVM subsystem;
- A value of 4E564D65h ("NVMe") is written to the NSSR.NSSRC field;
- Requested using a method defined in the NVMe Management Interface specification; or
- A vendor specific event occurs.

…

The ability for host software to initiate an NVM Subsystem Reset by writing to the NSSR.NSSRC field is an optional capability of a controller indicated by the state of the CAP.NSSRS field. An implementation may protect the NVM subsystem from an inadvertent NVM Subsystem Reset by not providing this capability to one or more controllers that make up the NVM subsystem.

The occurrence of a vendor specific event that results in an NVM Subsystem Reset is intended to allow implementations to recover from a severe NVM subsystem internal error that prevents continued normal operation (e.g., fatal hardware or firmware error).

7.3.2 Controller Level Reset

There are five ~~primary~~ methods to initiate a Controller Level Reset ~~mechanisms~~:

- NVM Subsystem Reset;
- Conventional Reset (i.e., PCI Express Hot, Warm, or Cold reset);
- PCI Express transaction layer Data Link Down status;
- Function Level Reset (i.e., PCI reset); and
- Controller Reset (i.e., CC.EN transitions from '1' to '0').

A Controller Level Reset consists of ~~When any of the above resets occur,~~ the following actions ~~are performed~~:

- The controller stops processing any outstanding Admin or I/O commands;
- All I/O Submission Queues are deleted;
- All I/O Completion Queues are deleted;
- The controller is brought to an Idle state. When this is complete, CSTS.RDY is cleared to '0'; and
- The Admin Queue registers (AQA, ASQ, or ACQ) are not reset as part of a C~~c~~ontroller R~~r~~eset. All other controller registers defined in section 3 and internal controller state are reset.

In all Controller Level Reset cases except a Controller Reset, the PCI register space is reset as defined by the PCI Express base specification. Refer to the PCI Express specification for further details.

To continue after a Controller Level R~~r~~eset, the host shall:

- Update register state as appropriate;
- Set CC.EN to '1';
- Wait for CSTS.RDY to be set to '1';
- Configure the controller using Admin commands as needed;
- Create I/O Completion Queues and I/O Submission Queues as needed; and
- Proceed with normal I/O operations.

Note that all Controller Level Reset cases except a Controller Reset result in the controller immediately losing communication with the host. In these cases, the controller is unable to indicate any aborts or update any completion queue entries.

…

Modify a portion of section 7.4 (Queue Management) as shown below:

7.4 Queue Management

7.4.1 Queue Setup and Initialization

To setup and initialize I/O Submission Queues and I/O Completion Queues for use, host software follows these steps:

1. Configures the Admin Submission and Completion Queues by initializing the Admin Queue Attributes (AQA), Admin Submission Queue Base Address (ASQ), and Admin Completion Queue Base Address (ACQ) registers appropriately;
2. Configure the size of the I/O Submission Queues (CC.IOSQES) and I/O Completion Queues (CC.IOCQES);
3. Submits a Set Features command with the Number of Queues attribute to request the desired number of I/O Submission Queues and I/O Completion Queues. The completion queue entry for this Set Features command indicates the number of I/O Submission Queues and I/O Completion Queues allocated by the controller;
4. Determines the maximum number of entries supported per queue (CAP.MQES) and whether the queues are required to be physically contiguous (CAP.CQR);
5. Creates the desired I/O Completion Queues within the limitations of the number allocated by the controller and the queue attributes supported (maximum entries and physically contiguous requirements) by using the Create I/O Completion Queue command; and
6. Creates the desired I/O Submission Queues within the limitations of the number allocated by the controller and the queue attributes supported (maximum entries and physically contiguous requirements) by using the Create I/O Submission Queue command.

At the end of this process, the desired I/O Submission Queues and I/O Completion Queues have been setup and initialized and may be used to complete I/O commands.

…

Modify a portion of section 7.8 (Feature Values) as shown below:

7.8 Feature Values

The Get Features command~~, defined in~~ (refer to section 5.13), and Set Features command~~, defined in~~ (refer to section 5.21), may be used to read and modify operating parameters of the controller. The operating parameters

are grouped and identified by Feature Identifiers. Each Feature Identifier contains one or more attributes that may affect the behavior of the Feature.

If bit 4 is set to '1' in the Optional NVM Command Support (ONCS) field of the Identify Controller data structure in Figure 111, then for each Feature, there are three settings: default, saveable, and current. If bit 4 is cleared to '0' in the Optional NVM Command Support field of the Identify Controller data structure ~~in Figure 111~~ then the controller only supports a current and default value for each Feature. In this case, the current value may be persistent across power states based on the information specified in Figure 128 and Figure 129.

The default value for each Feature is vendor specific and set by the manufacturer; unless otherwise specified; it is not changeable. The saveable value is the value that the Feature has after a power on or reset event. The controller may not support a saveable value for a Feature; this is discovered by using the 'supported capabilities' value in the Select field in Get Features. If the controller does not support a saveable value for a Feature, then the default value is used after a power on or reset event. The current value for a Feature is the value ~~actively~~ in active use by the controller for ~~a~~ that Feature ~~after a Set Features command completes~~.

A Set Features command may be used to modify the saveable value, if supported, and the current value for a Feature. A Get Features command may be used to read the default value, the saveable value, if supported, and the current value for a Feature. If the controller does not support a saveable value for a Feature, then the default value is returned for the saveable value in for a Get Features command.

Feature settings may apply to:

a) the controller (i.e., the feature is not namespace specific); or
b) a namespace (i.e., the feature is namespace specific).

For feature values that apply to the controller:

a) if the ~~CDW1.~~NSID field is set to 0h or FFFFFFFFh, then:
   - the Set Features command shall set the specified feature value for the controller; and
   - the Get Features command shall return the current setting of the requested feature value for the controller;

   and
b) if the ~~CDW1.~~NSID field is set to a valid namespace identifier (refer to section 6.1), then:
   - the Set Features command shall fail with a status code of Feature Not Namespace Specific; and
   - the Get Features command shall return the current setting of the requested feature value for the controller.

For feature values that apply to a namespace:

a) if the ~~CDW1.~~NSID field is set to an active namespace identifier (refer to section 6.1), then:
   - the Set Features command shall set the specified feature value of the specified namespace; and
   - the Get Features command shall return the current setting of the requested feature value for the specified namespace;
b) if the ~~CDW1.~~NSID field is set to FFFFFFFFh, then:
   - the Set Features command shall, unless otherwise specified, set the specified feature value for all namespaces attached to the controller processing the command; and
   - the Get Features command shall, unless otherwise specified in section 5.21.1, fail with a status code of Invalid Namespace or Format;

   and
c) if the ~~CDW1.~~NSID field is set to any other value, then the Set Features command and the Get Features command shall fail as described in the rules for namespace identifier usage in Figure 11.

…

Modify a portion of 7.9 (NVMe Qualified names) as shown below:

7.9 NVMe Qualified Names

…

There are two supported NQN formats. The first format may be used by any organization that owns a domain name. This naming format may be used to create a human ~~readable~~ interpretable string to describe the host or NVM subsystem. This format consists of:

- The string “nqn.”;
- A date code, in “yyyy-mm~~.~~” format. This date shall be during a time interval when the naming authority owned the domain name used in this format. The date code uses the Gregorian calendar. All digits and the dash shall be included;
- The string “.” (i.e., the ASCII period character);
- The reverse domain name of the naming authority that is creating the NQN; and
- A colon (:) prefixed string that the owner of the domain name assigns that does not exceed the maximum length. The naming authority is responsible to ensure that the NQN is worldwide unique.

The following are examples of NVMe Qualified Names that may be generated by “Example NVMe, Inc.”

- nqn.2014-08.com.example:nvme:nvm-subsystem-sn-d78432
- nqn.2014-08.com.example:nvme.host.sys.xyz

The second format may be used to create a unique identifier when there is not a naming authority or there is not a desire for a human ~~readable~~ interpretable string. This format consists of:

- The string “nqn.”;
- The string “2014-08.org.nvmexpress:uuid:”; and
- A 128-bit UUID based on the definition in RFC 4122 represented as a string formatted as “*11111111-2222-3333-4444-555555555555*”.

The following is an example of an NVMe Qualified Name using the UUID-based format:

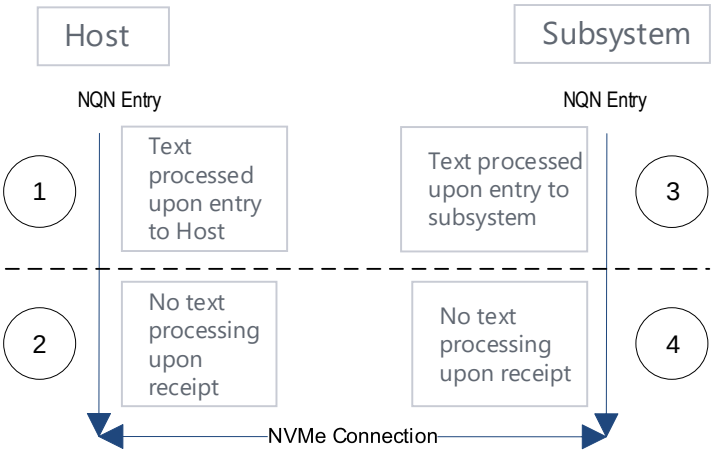
- nqn.2014-08.org.nvmexpress:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6

NVMe hosts, controllers and NVM subsystems compare (e.g., for equality) NVMe Qualified Names used by NVMe as binary strings without any text processing or text comparison logic that is specific to the Unicode character set or locale (e.g., case folding or conversion to lower case, Unicode normalization). Any such text processing:

a) may occur as part of entry of NVMe Qualified Names into NVMe hosts and NVM subsystems; and
b) should not occur as part of receiving NVMe Qualified Names via an NVMe connection, as shown in Figure 250.a.

Upon entry (e.g., at point 1 in Figure 250.a, described as “input” in RFC4122), NVMe host software may process an NVMe Qualified Name (e.g., for conversion to lower case based on the Unicode locale). Upon entry (e.g., at point 3 in Figure 250.a, described as “input” in RFC4122), a controller may process an NVMe Qualified Name (e.g., for conversion to lower case based on the Unicode locale). Upon receipt by the host (e.g., at point 2 in Figure 250.a) of an NVMe Qualified Name from the controller, no text process (e.g., no case folding) should occur. Upon receipt by the controller (e.g., at point 4 in Figure 250.a) of an NVMe Qualified Name from the host, no text processing (e.g., no case folding) should occur.

Figure 250.a NQN Processing



Modify a portion of section 7.11 (Unique Identifier) as shown below:

7.11 Unique Identifier

The NVM Subsystem NVMe Qualified Name specified in the Identify Controller data structure (refer to Figure 111) should be used (e.g., by host software) as the unique identifier for the NVM subsystem. If the controller complies with an older version of this specification that does not include the NVM Subsystem NQN, then the PCI Vendor ID, Serial Number, and Model Number fields in the Identify Controller data structure and the NQN Starting String "nqn.2014.08.org.nvmexpress:" may be combined by the host to form a globally unique value that identifies the NVM subsystem (e.g., for host software that uses NQNs). The method shown in Figure 251 should be used by the host to construct an NVM Subsystem NQN for older NVM subsystems that do not provide an NQN in the Identify Controller data structure. The mechanism used by the vendor to assign Serial Number and Model Number values to ensure uniqueness is outside the scope of this specification.

…

Modify a portion of section 7.12 (Keep Alive) as shown below:

7.12 Keep Alive

The Keep Alive feature (refer to section 5.21.1.15) is used by the host to determine that the controller is operational and used by the controller to determine that the host is operational. The host and controller are operational when each is accessible and able to issue or execute commands. The controller indicates the granularity of the Keep Alive Timer in the Identify Controller data structure.

The Keep Alive timer is a watchdog timer intended to detect a malfunctioning connection, controller, or host. The Keep Alive Timeout is the maximum time a connection remains established without processing a Keep Alive command. The Keep Alive timer in the controller expires when a Keep Alive command is not received within the Keep Alive Timeout interval.

The Keep Alive timer is active ~~only for an enabled controller, i.e., the Keep Alive timer is active~~ if:

- CC.EN is set to '1'; ~~and~~
- CSTS.RDY is set to '1'; ~~and~~
- CC.SHN is cleared to '0'; ~~and~~
- CSTS.SHST is cleared to '0'; and
- the Keep Alive Timer feature has been enabled with a KATO field (refer to section 5.21.1.15 or the Fabric Connect command in the NVMe-oF specification) set to a non-zero value,~~.~~

~~O~~otherwise, the Keep Alive timer is inactive and a Keep Alive Timeout shall not occur.

Activating an inactive Keep Alive timer (e.g., enabling a controller with the Keep Alive feature in use, enabling an NVMe-oF controller where the Fabric Connect command specified a non-zero Keep Alive Timeout (refer to the NVMe-oF specification)) shall initialize the Keep Alive timer to the Keep Alive Timeout value.

…

The NVMe Transport binding specification defines for the associated NVMe Transport:

- the minimum Keep Alive Timeout value;
- the maximum Keep Alive Timeout value; and
- if support for the Keep Alive feature is required.

NVMe Transports that do not detect a connection loss in a timely manner shall require that the Keep Alive feature be enabled. If a command attempts to disable the Keep Alive timer by setting the Keep Alive T~~t~~imeout value to 0h or to a value that exceeds the maximum allowed by the associated NVMe Transport binding specification, a status value of Keep Alive Timeout Invalid shall be returned. If a command sets the Keep Alive T~~t~~imeout value to a value that is ~~smaller~~ less than the minimum supported by the NVMe Transport or less than the minimum supported by the specific implementation, then the controller ~~rounds up~~ sets the Keep Alive T~~t~~imeout value to ~~the~~ that minimum value.

7.12.1 NVMe over PCIe Implementations

The Keep Alive feature is not required for NVMe over PCIe implementations. The PCIe Transport does not impose any limitations on the minimum and maximum Keep Alive Timeout value.

…

Modify a portion of section 8.1 (Firmware Update Process) as shown below:

8 Features

8.1 Firmware Update Process

…

The process for a firmware update to be activated without a reset is:

…

3. The controller completes the Firmware Commit command. The following actions are taken in certain error scenarios:

   a. If the firmware image is invalid, then the controller reports the appropriate error (e.g., Invalid Firmware Image);
   b. If the firmware activation was not successful because a Controller Level R~~r~~eset is required to activate this firmware, then the controller reports an error of Firmware Activation Requires Controller Level Reset and the image is applied at the next Controller Level R~~r~~eset;
   c. If the firmware activation was not successful because an NVM Subsystem Reset is required to activate this firmware, then the controller reports an error of Firmware Activation Requires NVM Subsystem Reset and the image is applied at the next NVM Subsystem Reset;
   d. If the firmware activation was not successful because a Conventional Reset is required to activate this firmware, then the controller reports an error of Firmware Activation Requires Conventional Reset and the image is applied at the next Conventional Reset; and
   e. If the firmware activation was not successful because the firmware activation time would exceed the MTFA value reported in the Identify Controller data structure, then the controller reports an error of Firmware Activation Requires Maximum Time Violation. In this case, to activate the firmware, the Firmware Commit command needs to be re-issued and the image activated using a reset.

…

If ~~a~~ the controller transitions to the $D3_{cold}$ ~~condition occurs~~ state (refer to the PCI Express Base Specification) after the submission of a Firmware Commit command that attempts to activate a firmware image and before the completion of that command then ~~during the firmware activation process~~, the controller may resume operation with either the ~~old~~ firmware image active at the time the Firmware Commit command was submitted or ~~new~~ the firmware image that was activated by that command.

If the firmware is not able to be successfully loaded, then the controller shall revert to the ~~previously active~~ firmware image present in the most recently activated firmware slot or the baseline read-only firmware image, if available, and indicate the failure as an asynchronous event with a Firmware Image Load Error.

If a host overwrites (i.e., updates) the firmware in the active firmware slot, then the previously active firmware image may no longer be available. As a result, any action (e.g., power cycling the controller) that requires the use of that firmware slot may instead use the firmware image that is currently in that firmware slot.

Host software should not …

…

Modify a portion of section 8.3 (End to end Data Protection) as shown below:

8.3 End-to-end Data Protection (Optional)

To provide robust data protection from the application to the NVM media and back to the application itself, end-to-end data protection may be used. If this optional mechanism is enabled, then additional protection information (e.g. CRC) is added to the logical block that may be evaluated by the controller and/or host software to determine the integrity of the logical block. This additional protection information, if present, is either the first eight bytes of metadata or the last eight bytes of metadata, based on the format of the namespace. For metadata formats with more than eight bytes, if the protection information is contained within the first eight bytes of metadata, then the CRC does not cover any metadata bytes. For metadata formats with more than eight bytes, if the protection information is contained within the last eight bytes of metadata, then the CRC covers all metadata bytes up to but excluding these last eight bytes. As described in section 8.2, metadata and hence this protection information may be configured to be contiguous with the logical block data or stored in a separate buffer.

The most commonly used data protection mechanisms in Enterprise implementations are SCSI Protection Information, commonly known as Data Integrity Field (DIF), and the Data Integrity Extension (DIX). The primary difference between these two mechanisms is the location of the protection information. In DIF, the protection information is contiguous with the logical block data and creates an extended logical block, while in DIX, the protection information is stored in a separate buffer. The end-to-end data protection mechanism defined by this specification is functionally compatible with both DIF and DIX. DIF functionality is achieved by configuring the metadata to be contiguous with logical block data (as shown in Figure 252), while DIX functionality is achieved by configuring the metadata and data to be in separate buffers (as shown in Figure 253).

The NVM Express interface supports the same end-to-end protection types ~~as DIF~~ defined in the SCSI Protection information model specified in SBC-3. The type of end-to-end data protection (i.e., Type 1, Type 2, or Type 3) is selected when a namespace is formatted and is reported in the Identify Namespace data structure (refer to Figure 109).

The Protection Information format is shown in Figure 254 and is contained in the metadata associated with each logical block. The Guard field contains a CRC-16 computed over the logical block data. The formula used to calculate the CRC-16 is defined in SBC-3. In addition to a CRC-16, DIX also specifies an optional IP checksum that is not supported by the NVM Express interface. The Application Tag is an opaque data field not interpreted by the controller and that may be used to disable checking of protection information. The Reference Tag associates logical block data with an address and protects against misdirected or out-of-order logical block transfer. Like the Application Tag, the Reference Tag may also be used to disable checking of protection information.

…

8.3.1.1 Protection Information and Write Commands

Figure 255 provides some examples of the protection information processing that may occur as a side effect of a Write command.

If the namespace is not formatted with end-to-end data protection, then logical block data and metadata is transferred from the host to the NVM with no protection information related processing by the controller.

If the namespace is formatted with protection information and the PRACT bit is cleared to ‘0’, then logical block data and metadata, which contains the protection information and may contain additional metadata, are transferred from the host buffer to NVM (i.e., the metadata field remains the same size in the NVM and the host buffer). As the logical block data and metadata passes through the controller, the protection information is checked. If a protection information check error is detected, the command completes with the status code of the error detected (i.e., End-to-end Guard Check Error, End-to-end Application Tag Check Error, or End-to-end Reference Tag Check Error).

If the namespace is formatted with protection information and the PRACT bit is set to ‘1’, then:

…

8.3.1.2 ~~The PRACT Bit~~ Protection Information and Read Commands

Figure 256 provides some examples of the protection information processing that may occur as a side effect of Read command processing.

If the namespace is formatted with protection information and the PRACT bit is cleared to ‘0’, then the logical block data and metadata, which in this case contains the protection information and possibly additional host metadata, is transferred by the controller from the NVM to the host buffer (i.e., the metadata field remains the same size in the NVM and the host buffer). As the logical block data and metadata pass through the controller, the protection information within the metadata is checked. If a protection information check error is detected, the command completes with the status code of the error detected (i.e., End-to-end Guard Check Error, End-to-end Application Tag Check Error, or End-to-end Reference Tag Check Error).

If the namespace is formatted with protection information and the PRACT bit is set to '1', then:

a) if the namespace is formatted with Metadata Size equal to 8 (refer to Figure 110), the logical block data and metadata (which in this case is, by definition, the protection information)~~,~~ is read from the NVM by the controller. As the logical block and metadata pass through the controller, the protection information is checked. If a protection information check error is detected, the command completes with the status code of the error detected (i.e., End-to-end Guard Check Error, End-to-end Application Tag Check Error, or End-to-end Reference Tag Check Error). After processing the protection information, the controller strips it and returns the logical block data to the host (i.e., the metadata is not resident within the host buffer); and

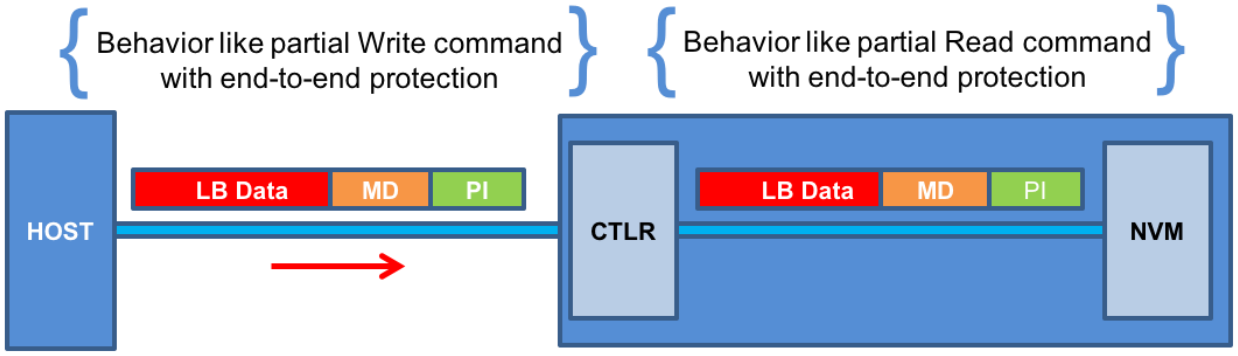
b) if the namespace is formatted with Metadata Size greater than 8, the logical block data and the metadata, which in this case contains the protection information and additional host formatted metadata, is read from the NVM by the controller. As the logical block and metadata pass through the controller, the protection information embedded within the metadata is checked. If a protection information check error is detected, the command completes with the status code of the error detected (i.e., End-to-end Guard Check Error, End-to-end Application Tag Check Error, or End-to-end Reference Tag Check Error). After processing the protection information, the controller passes the logical block data and metadata, with the embedded protection information unchanged, to the host (i.e., the metadata field remains the same size in the NVM as within the host buffer).

…

8.3.1.4 Protection ~~Checking with the~~ Information and Compare commands

Figure 257 illustrates the protection information processing that may occur as a side effect of Compare command processing. Compare command processing parallels both Write and Read commands. ~~The controller checks the protection information contained in the command and the protection information read from the NVM.~~ For the portion of the Compare command that transfers data and protection information from the host to the controller, the protection information checks performed by the controller parallels the Write command protection information checks (refer to section 8.3.1.1). For the portion of the Compare command that transfers data and protection information from the NVM media to the controller, the protection information checks performed by the controller parallels the Read command protection information checks (refer to section 8.3.1.2).

Figure 257: Protection Information Processing for Compare



Protection Information with PRACT bit set to '0' (i.e., pass)

Modify a portion of section 8.4 (Power Management) as shown below:

8.4 Power Management

…

Associated with each power state is a Power State Descriptor in the Identify Controller data structure (refer to Figure 113). The descriptors for all implemented power states may be viewed as forming a table as shown in the example in Figure 259 for a controller with seven implemented power states. Note that Figure 259 is illustrative and does not include all fields in the power state descriptor. The Maximum Power (MP) field indicates the sustained maximum power that may be consumed in that state, where power measurement methods are outside the scope of this specification. ~~R~~ (e.g., refer to the appropriate form factor specification for power measurement methodologies for that form factor). The controller may employ autonomous power management techniques to reduce power consumption below this level, but under no circumstances is power allowed to exceed this level except for non-operational power states as described in section 8.4.1.

…

8.4.3 NVM Subsystem Workloads

…

Figure 260: Workload Hints

Value	Description
…	
001b	**Workload #1:** Extended Idle Period with a Burst of Random Writes. Workload #1 consists of five (5) minutes of idle followed by thirty-two (32) random write commands of size 1 MiB submitted to a single controller while all other controllers in the NVM subsystem are idle, and then thirty (30) seconds of idle.
010b	**Workload #2:** Heavy Sequential Writes. Workload #2 consists of 80,000 sequential write commands of size 128 KiB submitted to a single controller while all other controllers in the NVM subsystem are idle. The submission queue(s) should be sufficiently large allowing the host to ensure there are multiple commands pending at all times during the workload.
011b – 111b	Reserved

8.4.4 Runtime D3 Transitions

…

The latency reported is based on a normal shutdown with optimal controller settings preceding the RTD3 resume. The latency reported assumes that host software enables and initializes the controller and sends a 4 KiB read operation.

The RTD3 Entry Latency is the expected elapsed time from the time CC.SHN is set to 01b by host software until CSTS.SHST is set to 10b by the controller. When CSTS.SHST is set to 10b, it is safe for host software to remove power from the controller.

In this specification, RTD3 refers to the $D3_{cold}$ power state described in the PCI Express specification. RTD3 does not include the PCI Express $D3_{hot}$ power state because main power is not removed from the controller in the $D3_{hot}$ power state. Refer to the PCI Express Base Specification for details on the $D3_{hot}$ power state and the $D3_{cold}$ power state.

…

Modify a portion of section 8.5.3 (Secondary Controller States and Resource Configuration) as shown below:

8.5.3 Secondary Controller States and Resource Configuration

…

A primary controller or secondary controller is enabled when CC.EN and CSTS.RDY are both set to '1' for that controller. A secondary controller ~~may only~~ is able to be enabled only when it is in the Online state. If the primary controller associated with a secondary controller is disabled or undergoes a Controller Level Reset, then the secondary controller shall implicitly transition to the Offline state ~~implicitly~~.

Modify a portion of section 8.8 (Reservations) as shown below:

8.8 Reservations (Optional)

NVM Express reservations provide capabilities that may be utilized by two or more hosts to coordinate access to a shared namespace. The protocol and manner in which these capabilities are used is outside the scope of this specification. Incorrect application of these capabilities may corrupt data and/or otherwise impair system operation.

A reservation on a namespace restricts hosts access to that namespace. If a host submits a command to a namespace in the presence of a reservation and lacks sufficient rights, then the command is aborted by the controller with a status of Reservation Conflict. If a host submits a command with the NSID set to FFFFFFFFh in the presence of a reservation on any of the namespaces impacted by that command and that host lacks sufficient rights on all the impacted namespaces, then the command is aborted by the controller with a status of Reservation Conflict. Capabilities are provided that allow recovery from a reservation on a namespace held by a failing or uncooperative host.

…

A host may be associated with multiple controllers. In Figure 263 host A is associated with two controllers while hosts B and C are each associated with a single controller. A host registers a Host Identifier (~~Host Identifier~~ refer to section 5.21.1.24) with each controller with which it is associated using a Set Features command (refer to section 5.21) prior to performing any operations associated with reservations.

…

Controllers that make up an NVM subsystem shall all have the same support for reservations. Although strongly encouraged, namespaces that make up an NVM subsystem are not all required to have the same support for reservations. For example, some namespaces within a single controller may support reservations while others do not, or the supported reservation types may differ among namespaces. If a controller supports reservations, then the controller shall:

- Indicate support for reservations by returning a '1' in bit 5 of the Optional NVM Command Support (ONCS) field in the Identify Controller data structure;
- Support the Reservation Report command (refer to section 6.13), Reservation Register command (refer to section 6.11), Reservation Acquire command (refer to section 6.10), and Reservation Release command (refer to section 6.12);

…

NOTE: The behavior of Ignore Existing Key has been changed to improve compatibility with SCSI based implementations. Conformance to the modified behavior is indicated in the Reservation Capabilities (RESCAP) field of the Identify Namespace data structure. For the previous definition of Ignore Existing Key behavior, refer to revision 1.2.1.

8.8.1 Reservation Notifications

There are three types of reservation notifications: registration preempted, reservation released, and reservation preempted. Conditions that cause a reservation notification to occur are described in the following sections. A Reservation Notification log page is created whenever an unmasked reservation notification occurs on a namespace associated with the controller (refer to section 5.14.1.9.1). Reservation notifications may be masked from generating a reservation log page on a per reservation notification type and per namespace ID basis through the Reservation Notification Mask feature (refer to section 5.21.1.20). A host may use the Asynchronous Event Request command (refer to section 5.2) to be notified of the presence of one or more available Reservation Notification log pages (refer to section ~~5.2~~ 5.14.1.9.1).

…

8.8.2 Registering

…

A host registers a reservation key by executing a Reservation Register command (refer to section 6.11) on the namespace with the Reservation Register Action (RREGA) field set to 000b (i.e., Register Reservation Key) and supplying a reservation key in the New Reservation Key (NRKEY) field.

…

8.8.3 Reservation Types

…

Reservations and registrations persist across all Controller Level Resets and all NVM Subsystem Resets except reset due to power loss. A reservation may be optionally configured to be retained across a reset due to power loss using the Persist Through Power Loss State (PTPLS). A Persist Through Power Loss State (PTPLS) is associated with each namespace that supports reservations and may be modified as a side effect of a Reservation Register command (refer to section 6.11) or a Set Features command (refer to section 5.21).

…

8.8.4 Unregistering

A host that is a registrant of a namespace may unregister with the namespace by executing a Reservation Register command (refer to section 6.11) on the namespace with the RREGA field set to 001b (i.e., Unregister Reservation Key) and supplying its current reservation key in the CRKEY field.

…

8.8.5 Acquiring a Reservation

In order for a host to obtain a reservation on a namespace, it shall be a registrant of that namespace. A registrant obtains a reservation by executing a Reservation Acquire command (refer to section 6.10), setting the Reservation Acquire Action (RACQA) field to 000b (Acquire), and supplying the current reservation key associated with the host in the Current Reservation Key (CRKEY) field. The CRKEY value shall match that used by the registrant to register with the namespace. If the ~~key~~ CRKEY value does not match, then the command is aborted with status Reservation Conflict. If the host is not a registrant, then the command is aborted with a status of Reservation Conflict.

…

8.8.6 Releasing a Reservation

Only a reservation holder may release in an orderly manner a reservation held on a namespace. A host releases a reservation by executing a Reservation Release command (refer to section 6.12), setting the Reservation Release Action (RRELA) field to 000b (i.e., Release), setting the Reservation Type (RTYPE) field to the type of reservation being released, and supplying the current reservation key associated with the host in the Current Reservation Key (CRKEY) field.

…

8.8.7 Preempting a Reservation or Registration

A host that is a registrant may preempt a reservation and/or registration by executing a Reservation Acquire command (refer to section 6.10), setting the Reservation Acquire Action (RACQA) field to 001b (Preempt), and supplying the current reservation key associated with the host in the Current Reservation Key (CRKEY) field. The CRKEY value shall match that used by the registrant to register with the namespace. If the CRKEY value does not match, then the command is aborted with status Reservation Conflict. The preempt actions that occur are dependent on the type of reservation held on the namespace, if any, and the value of the Preempt Reservation Key (PRKEY) field in the command. If the host is not a registrant, then the command is aborted with a status of Reservation Conflict. The remainder of this section assumes that the host is a registrant.

If the existing reservation type is not Write Exclusive - All Registrants and not Exclusive Access - All Registrants, then the actions performed by the command depend on the value of the PRKEY field as follows. If the PRKEY field value matches the reservation key of the current reservation holder, then the following occur as an atomic operation:

- ~~the reservation holder is~~ all registrants with a matching registration key other than the host that issued the command are unregistered~~,~~;
- the reservation is released~~,~~; and
- a new reservation is created of the type specified by the Reservation Type (RTYPE) field in the command for the host that issued the command as the reservation key holder.

If the PRKEY field value does not match that of the current reservation holder and is not equal to zero, then registrants whose reservation key matches the value of the PRKEY field are unregistered. If the PRKEY field value does not match that of the current reservation holder and is equal to zero, then the command is aborted with status Invalid Field in Command.

If the existing reservation type is Write Exclusive - All Registrants or Exclusive Access - All Registrants, then the actions performed by the command depend on the value of the PRKEY field as follows. If the PRKEY field value is zero, then the following occurs as an atomic operation:

- all registrants other than the host that issued the command are unregistered~~,~~;
- the reservation is released~~,~~; and
- a new reservation is created ~~for the host~~ of the type specified by the Reservation Type (RTYPE) field in the command for the host that issued the command as the reservation key holder.

If the PRKEY value is non-zero, then registrants whose reservation key matches the value of the PRKEY field are unregistered. If the PRKEY value is non-zero and there are no registrants whose reservation key matches the value of the PRKEY field, the controller should return an error of Reservation Conflict.

If there is no reservation held on the namespace, then execution of the command causes registrants whose reservation key match the value of the PRKEY field to be unregistered.

If the existing reservation type is not Write Exclusive - All Registrants and not Exclusive Access - All Registrants, then a ~~A~~ reservation holder may preempt itself using the above mechanism. When a host preempts itself the following occurs as an atomic operation:

- registration of the host is maintained~~,~~;
- the reservation is released~~,~~; and
- a new reservation is created for the host of the type specified by the RTYPE field.

A host may abort commands as a side effect of preempting a reservation by executing a Reservation Acquire command (refer to section 6.10) and setting the RACQA field to 010b (Preempt and Abort). The behavior of such a command is exactly the same as that described above with the RACQA field set to 001b (Preempt), with two exceptions:

- After the atomic operation changes namespace reservation and registration state, all controllers associated with any host whose reservation or registration is preempted by that atomic operation are requested to abort all commands being processed that ~~target~~ were addressed to the namespace

specified in the Namespace Identifier field (i.e., ~~CDW1.~~ the NSID field in ~~of~~ the Reservation Acquire command) (refer to section 4.11 for the definition of "being processed"); and

…

As with the Abort Admin command (refer to section 5.1), abort as a side effect of preempting a reservation is best effort; as a command that is requested to be aborted may currently be at a point in execution where it can no longer be aborted or may have already completed, when a Reservation Acquire or Abort Admin command is submitted.

…

8.8.8 Clearing a Reservation

A host that is a registrant may clear a reservation (i.e., force the release of a reservation held on the namespace and unregister all registrants) by executing a Reservation Release command (refer to section 6.12), setting the Reservation Release Action (RRELA) field to 001b (i.e., Clear), and supplying the current reservation key associated with the host in the Current Reservation Key (CRKEY) field.

…

8.8.9 Reporting Reservation Status

A host may determine the current reservation status associated with a namespace by executing a Reservation Report command (refer to section 6.13).

…

Modify Section 8.10 (Replay Protected Memory Block) as shown:

8.10 Replay Protected Memory Block (Optional)

…

The controller may support multiple RPMB targets. RPMB targets are not contained within a namespace. Controllers in the NVM subsystem may share the same RPMB targets. Security Send and Security Receive commands for RPMB do not use the namespace ID field; NSID shall be cleared to 0h. Each RPMB target operates independently – there may be requests outstanding to multiple RPMB targets at once (where the requests may be interleaved between RPMB targets). In order to guarantee ordering the host should issue and wait for completion for one Security Send or Security Receive command at a time. Each RPMB target requires individual authentication and key programming. Each RPMB target may have its own unique Authentication Key.

…

Figure 269: RPMB Contents

Content	Type	Size	Description
…			
RPMB Data Area	Readable and writable, not erasable	Size is reported in Identify Controller data structure (128 KiB minimum, 32 MiB maximum)	Data ~~which may only be~~ that is able to be read and written only via successfully authenticated read/write access.

…

8.10.2.1 Authentication Key Programming

…

Figure 271: RPMB – Authentication Key Data Flow

Command	Bytes in Command	Field Name	Value	Objective
...				
Security Receive 1	**Data populated by the controller and returned to the host**			Retrieve the Key Programming Result
	222-*N*:00	Stuff Bytes	0…00h	
	222:222-(*N*-1)	MAC/Key	0…00h	
	223	RPMB Target	*RPMB target ~~to access~~ response was sent from*	
	239:224	Nonce	0…00h	
	...			

8.10.3 Authenticated Device Configuration Block Write

…

When the host receives a successful completion of the Security Send command from the controller, it should send a Security Receive command to the controller to retrieve the data. The controller returns an RPMB Data Frame with Response Message Type (0600h), the incremented counter value, the MAC, and the Result. All other fields are cleared to 0h.

…

Figure 275: RPMB – Authenticated Device Configuration Block Write Flow

Command	Bytes in Command	Field Name	Value	Objective
Security Send 1	**Data populated by the host and sent to the controller**			Request Device Configuration Block Write
	222-*N*:00	Stuff Bytes	0…00h	
	222:222-(*N*-1)	MAC/Key	*MAC generated by the host*	
	223	RPMB Target	00h	
	239:224	Nonce	0…00h	
	243:240	Write Counter	*Current Write Counter value*	
	247:244	Address	0000 0000h	
	251:248	Sector Count	0000 0001h	
	253:252	Result	0000h	
	255:254	Request/Response	0006h *(Request)*	
	767:256	Data	*RPMB Device Configuration Block data structure*	
Security Send 2	**Data populated by the host and sent to the controller**			Request Result of data programming
	222-*N*:00	Stuff Bytes	0…00h	
	222:222-(*N*-1)	MAC/Key	0…00h	
	223	RPMB Target	*RPMB target to access*	
	239:224	Nonce	0…00h	
	243:240	Write Counter	0000 0000h	
	247:244	Address	0000 0000h	
	251:248	Sector Count	0000 0000h	
	253:252	Result	0000h	
	255:254	Request/Response	0005h *(Request)*	
Security Receive 1	**Data populated by the controller and returned to the host**			Retrieve Device Configuration Block Write Result
	222-*N*:00	Stuff Bytes	0…00h	
	222:222-(*N*-1)	MAC/Key	*MAC generated by the controller*	
	223	RPMB Target	00h	
	239:224	Nonce	0…00h	
	243:240	Write Counter	*Incremented Write Counter value*	
	247:244	Address	0000 0000h	
	251:248	Sector Count	0000 0000h	
	253:252	Result	*Result Code*	
	255:254	Request/Response	0600h *(Response)*	

Modify a porting of section 8.11 (Device Self-test Operations) as shown below:

8.11 Device Self-test Operations (Optional)

…

8.11.1 Short Device Self-Test Operation

…

A short device self-test operation:

a) shall be aborted by any Controller Level Reset that affects the controller on which the device self-test is being performed;
b) shall be aborted by a Format NVM command as described in Figure 277.a;~~, if the Namespace Identifier field specified in the Format NVM command is the same as the Device Self-test command that invoked the device self-test operation;~~
c) shall be aborted if a Device Self-test command with the Self-Test Code field set to Fh is processed; and
d) may be aborted if the specified namespace is removed from the namespace inventory.

Figure 277.a: Format NVM command Aborting a Device Self-Test Operation

FNA bit[1]	NSID in Format NVM command	NSID in Device Self-test Command	Abort Device Self-Test operation?
0	Any valid NSID value (refer to section 6.1)	Any valid NSID value (refer to section 6.1)	Yes, if the NSID values are the same
0	FFFFFFFFh	Any valid NSID value (refer to section 6.1)	Yes
0	Any valid NSID value (refer to section 6.1)	FFFFFFFFh	Optional
0	FFFFFFFFh	FFFFFFFFh	Yes
1	Ignored	Ignored	Yes

Key:
Optional = The device self-test operation is not required to be aborted but may be aborted.

NOTES:
1. For a Format NVM command with Secure Erase, this column refers to Bit 1 in the FNA field in the Identify Controller data structure (refer to Figure 111) and bit 0 in the FNA field is ignored. For a Format NVM command without Secure Erase, this column refers to bit 0 in the FNA field, and bit 1 in the FNA field is ignored.

8.11.2 Extended Device Self-Test Operation

…

An extended device self-test operation:

a) shall be aborted by a Format NVM command as described in Figure 277.a;~~, if the Namespace Identifier field specified in the Format NVM command is the same as Device Self-test command the invoked the device self-test operation;~~
b) shall be aborted if a Device Self-test command with the Self-Test Code field set to Fh is processed; and
c) may be aborted if the specified namespace is removed from the namespace inventory.

Modify a portion of section 8.12 (Namespace Management) as shown below:

8.12 Namespace Management (Optional)

…

The total and unallocated NVM capacity for the NVM subsystem is reported in the Identify Controller data structure. For each namespace, the NVM capacity used for that namespace is reported in the Identify

Namespace data structure. The controller may allocate NVM capacity in units such that the requested size for a namespace may be rounded up to the next unit boundary. For example, if host software requests a namespace of 32 logical blocks with a logical block size of 4 KiB for a total size of 128 KiB and the allocation unit for the implementation is 1 MiB then the NVM capacity consumed may be rounded up to 1 MiB. The NVM capacity fields may not correspond to the logical block size multiplied by the total number of logical blocks.

To create a namespace, host software performs the following actions:

1. Host software requests the Identify Namespace data structure that specifies common namespace capabilities (i.e., using an Identify command with ~~a setting of CDW1.~~ the NSID field set to FFFFFFFFh and the CNS field cleared to 0h);

…

Modify a portion of section 8.13 (Boot Partitions) as shown below:

8.13 Boot Partitions (Optional)

…

An NVMe controller that supports Boot Partitions has two Boot Partitions of equal size using Boot Partition identifiers 0h and 1h. The two Boot Partitions allow the host to update one and verify the contents before marking the Boot Partition active. Controllers in the NVM subsystem may share the same Boot Partitions.

The contents of Boot Partitions are **…**

…

8.13.3 Boot Partition Protection

…

All Boot Partitions remain unlocked until Boot Partition Protection is enabled by host software. Host software enables Boot Partition Protection by setting the Boot Partition Protection Enable bit in the RPMB Device Configuration Block data structure (refer to section 8.10). Once Boot Partition Protection is enabled, the controller shall reject Authenticated Device Configuration Block Writes that disable Boot Partition Protection (i.e., enabling Boot Partition Protection is permanent). Once Boot Partition Protection is enabled, Boot Partitions ~~may only~~ are able to be modified only after unlocking the Boot Partition using RPMB.

After activating Boot Partition Protection, the default state for all Boot Partitions is the “Locked” state. In this state, host software may read a Boot Partition. In this state, the controller rejects attempts to write to a Boot Partition using the Firmware Commit command.

Modify a portion of section 9 (Directives) as shown below:

9 Directives

…

9.2.2.1 Enable Directive (Directive Operation 01h)

The Enable Directive operation is used to enable a specific Directive for use within a namespace by all controllers that are associated with the same Host Identifier. The DSPEC field in command Dword 11 is not used for this operation. The Identify Directive is always enabled. The enable state of each Directive on each

shared namespace attached to enabled controllers associated with the same non-zero Host Identifier is the same. If an NSID value of FFFFFFFFh is specified, then the Enable Directive operation applies to the NVM subsystem (i.e., all namespaces and all controllers associated with the NVM subsystem). On an NVM Subsystem Reset, all Directives other than the Identify Directive are disabled for the entire NVM subsystem.

On a~~ny other type of~~ Controller Level Reset:

- all Directives other than the Identify Directive are disabled for that controller; and
- if there is an enabled controller associated with the Host Identifier for the controller that was reset, then for namespaces attached to enabled controllers associated with that Host Identifier, Directives are not disabled.

…

For all controllers in an NVM subsystem that have the same non-zero Host Identifier, if a host changes the enable state of any Directive for a shared namespace attached to a controller by a means other than a Controller Level Reset, then that change shall be made to the enable state of that Directive for that namespace attached to any other controller associated with that Host Identifier.

…

Modify a portion of section 9.3 (Streams) as shown below:

9.3 Streams (Directive Type 01h, Optional)

…

If the Streams Directive becomes disabled for use by a host within a namespace, then all stream resources and stream identifiers ~~are~~ shall be released for ~~the~~ that host ~~in that~~ for the affected namespace. If the host issues a Format NVM command, ~~or deletes a namespace,~~ then all stream identifiers for all open streams for affected namespaces ~~are~~ shall be released. If the host deletes a namespace, then all stream resources and all stream identifiers for that namespace shall be released.

…

9.3.1.3 Allocate Resources (Directive Operation 03h)

…

Figure 296: Allocate Resources – ~~Dword 0 of command completion queue entry~~ Completion Queue Entry Dword 0

Bit	Description
31:16	Reserved
15:00	**Namespace Streams Allocated (NSA):** This field indicates the number of streams resources that have been allocated for exclusive use by the namespace specified. The allocated resources are available to all controllers associated with that host.

…

9.3.2.1 Release Identifier (Directive Operation 01h)

The Release Identifier operation specifies that the stream identifier specified in the DSPEC field in command Dword 11 is no longer in use by the host. Specifically, if the host uses the stream identifier in a future operation then it is referring to a different stream. If the specified identifier does not correspond to an open stream for the specified namespace, then the command completes successfully. If there are stream resources allocated for the exclusive use of the specified namespace, then ~~the~~ those exclusive stream resources remain allocated for this namespace, and may be re-used in a subsequent write command. If there are no stream resources allocated for the exclusive use of the specified namespace, then the stream resources are returned to the NVM subsystem stream resources for future use by a namespace without exclusive allocated stream resources. If

an NSID value of FFFFFFFFh is specified, then the controller shall abort the command with a status of Invalid Field in Command.

No data transfer occurs.

9.3.2.2 Release Resources (Directive Operation 02h)

The Release Resources operation is used to release all streams resources allocated for the exclusive use of the namespace attached to all controllers associated with the same non-zero Host Identifier of the controller that processed the operation. On successful completion of this command, the exclusive allocated stream resources are released and the Namespace Streams Allocated (refer to Figure 293) field is cleared to 0h for the specified namespace. If this command is issued when no streams resources are allocated for the exclusive use of the namespace, the command shall complete successfully.

No data transfer occurs.
…

Modify a portion of section 10.3 (Memory Error Handling) as shown below:

10.3 Memory Error Handling

For PCI Express implementations, memory ~~Memory~~ errors such as target abort, master abort, and parity may cause the controller to stop processing the currently executing command. These are serious errors that cannot be recovered from without host software intervention.

A master/target abort error occurs when host software has ~~given a pointer to the host controller that does not exist in memory~~ provided, to the controller, the address of memory that does not exist. When this occurs, the ~~host~~ controller aborts the command with a Data Transfer Error status code.
…

Modify a portion of section 10.5 (Controller Fatal Status Condition) as shown below:

10.5 Controller Fatal Status Condition

If the controller has a serious error condition and is unable to communicate with host software via completion queue entries in the Admin Completion Queue or I/O Completion Queues, then the controller may set the Controller Fatal Status (CSTS.CFS) field to '1' (refer to section 3.1.6). This indicates to host software that a serious error condition has occurred. When this condition occurs, host software should attempt to reset and then re-initialize the controller.

The Controller Fatal Status condition is not indicated with an interrupt. If host software experiences timeout conditions and/or repeated errors, then host software should consult the Controller Fatal Status (CSTS.CFS) field to determine if a more serious error has occurred.

If the Controller Fatal Status (CSTS.CFS) field is set to '1' on any controller in the NVM subsystem, the host should issue a Controller Reset to that controller.

If that Controller Reset does not clear the Controller Fatal Status condition, the host should initiate an NVM Subsystem Reset (refer to section 7.3.1), if supported.

Performing an NVM Subsystem Reset (NSSR) may cause PCI Express links to go down as part of resetting the NVM Subsystem. Host software may have undesirable effects related to PCI Express links going down (e.g., some host operating systems or hypervisors may crash).

NVM Subsystem Reset should not be used if the host software has undesirable effects related to PCI Express links going down. This host software includes, but is not limited to, operating systems using Firmware First Error Handling (refer to the ACPI specification). Such operating systems should not use NSSR for recovery from CFS conditions.