



LEGAL NOTICE:

© Copyright 2008 to 2023 NVM Express, Inc. ALL RIGHTS RESERVED.

This erratum 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, 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: “© 2008 to 2023 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.

The NVM Express® design mark is a registered trademark of NVM Express, Inc. PCI-SIG®, PCI Express®, and PCIe® are registered trademarks of PCI-SIG. InfiniBand™ is a trademark and servicemark of the InfiniBand Trade Association.

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

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant’s agreement. Copyright © 2008-2023 NVM Express, Inc.

NVM Express® Technical Errata

Errata ID	113
Revision Date	08/23/2023
Affected Spec Ver.	NVM Express® Base Specification Revision 2.0c NVM Express® NVM Command Set Specification Revision 1.0c NVM Express® Zoned Namespace Command Set Specification Revision 1.1c NVM Express® Management Interface Specification Revision 1.2c
Corrected Spec Ver.	

Errata Author(s)

Name	Company
Judy Brock, Mike Allison, Bill Martin,	Samsung
Fred Knight	NetApp
David Black, Austin Bolen	DellEMC
Paul Suhler	Kioxia
Andres Baez	Solidigm
Yoni Shternhell	Western Digital

Errata Overview

This ECN updates and clarifies various text within the NVM Express Base Specification Revision 2.0c, the NVM Express NVM Command Set Specification Revision 1.0c, the NVM Express Zoned Namespace Command Set Specification 1.1c, and the NVM Express Management Interface Specification 1.2c.

Revision History

Revision Date	Change Description
6/1/2022	Initial creation
9/7/2022	Incorporated ECN114 as of 7/13/2022.
10/17/2022	Aligned to the new minor revisions of the specifications
1/24/2023	Closed all updates and prepared for Technical WG review.
1/26/2023	Made editorial changes as part of the TWG review.
1/30/2023	Added correcting the NVMe-MI Receive command to NVM Admin command mapping title name change.
2/16/2023	Corrected reference figures to Figure 98 as opposed to Figure 101
4/26/2023	Editorial changes during review in Technical WG.
5/9/2023	Editorial changes for the paragraph explaining overlaps between the destination LBA range and the Source Range entries for the Copy command.

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2023 NVM Express, Inc.

5/10/2023	Simplified the changes to the Number of LBA Formats field.
5/25/2023	Ready for member review.
8/23/2023	Integrated

Description of Changes

NVM Express Base Specification 2.0c:

Backward Incompatible Changes:

- Support for the NVMe-MI Commands Supported and Effects log page is optional if the NVMe-MI Send command and NVMe-MI Receive command is not supported.
- While a recommendation in the ECN, for the next major release of the NVM Express Base Specification, a controller shall abort a Set Features command for the Host Memory Buffer feature if the Host Memory Descriptor List Entry Count field is cleared to 0h.

Editorial Changes:

- Using the same numbering when referencing number of I/O queues.
- Clarified the Storage Entities section to specify storage entities and use lists.
- Clarified the CDPMLS field to indicate memory containing PRPS list or SGL.
- Clarified the reset restrictions of the CMBMSC property.
- Added the Get Log Page command as an Admin command that is NVM Set Aware.
- Clarified the use of the Property Get command and the Property Set command on a message-based Transport controller initialization.
- Clarified the commands types in the description of the Controller Ready Independent of Media section.
- Clarified the restrictions on the resetting of the CMBMSC property.
- Clarified that the reset after the host issues a Firmware Commit command is not the last step in the firmware update process.
- Clarified in the Asynchronous Event Request command definition that the Retain Asynchronous Event bit is cleared to '0' in a Get Log Page command to clear the event.
- Clarified that any Controller Level Reset that occurs between a firmware download and completion of the Firmware Commit command the downloaded images are discarded.
- Fixed spelling
- Move phrases in text for clarify.
- Individually identified each command in list of commands.
- Clarified that a controller ignores a Host Memory Buffer Descriptor if the Buffer Size field is cleared to 0h.
- Clarified the definition of the Generate Default Host Metadata field and the requirements of a Get Features command for the Host Metadata feature.
- Clarified that Controller Memory Buffer (CMB) support for PRP Lists and SGL about those being contained in the CMB.
- Corrected the RPMB Operation Result value for an Authenticated Data Write when the Write Counter has expired.
- Replaced "registration key value" for "reservation key value" as it is reservations that have key values.
- Clarified that a Management Endpoint can be used to access and change the contents of a Telemetry Host-Initiated log page.
- Clarified the step ordering when describing the basic steps in building a command.

NVM Express NVM Command Set Specification 1.0c:

Backward Incompatible Changes:

- A Copy command that specifies a destination LBA range that overlaps in any of the Source Range entries has undefined results (i.e., vendor specific results).
- The LBA field in the Error Information Log was described as containing the “first” LBA that experienced the error. The meaning of “first” was not defined (first in time vs. first in LBA ordering). This was clarified to indicate that the “lowest numbered” LBA is to be reported, rather than the LBA where the error was detected first (based on time of detection). This may be considered by some to be an incompatible change.

Editorial Changes:

- Formally defined logical block size and logical block data size to clarify the fields used to make calculations that use the format size of logical blocks of a namespace.
- Removed any reference to the first bytes of metadata being used for protection information since that was prohibited in the initial version of the NVM Express NVM Command Set specification.
- Clarified that an aborted Copy command may or may not have copied some of the data.
- Clarified the name of the Command Size Limit Exceeded status code for the Copy command.
- Clarified that the LBA reported in the Error Information Log Entry data structure is the lowest-numbered LBA that experienced the error condition.
- Clarified that Controller Memory Buffer (CMB) support for PRP Lists and SGL about those being contained in the CMB.
- Clarified that the Namespace Atomic Write Unit Normal (NAUWN) field, Namespace Atomic Write Unit Power Fail (NAWUPF) field, Namespace Atomic Compare & Write Unit (NACWU), Namespace Atomic Boundary Size Normal (NABSN) field, and Namespace Atomic Boundary Size Power Fail (NABSPF) field are 0's based except for the value of 0h.
- Changed the use of “host software” to just “host” as to not imply a host implementation.
- Clarified that the Protection information Checking is based on a variable sized Logical Block Reference Tag as opposed to a fixed 4 bytes.
- Clarified proper references to the LBA Format Data Structure figure.

NVM Express Zoned Namespace Command Set Specification 1.1c:

Editorial Changes:

- Made sure nesting levels use a different style of numbering for clarity.
- Added host responsibility for setting the ILBRT field on commands to zoned namespaces when the PIREMAP bit is set to ‘1’.

NVM Express Management Interface Specification 1.2c:

Editorial Changes:

- Renamed “Status Field field” to “Status field”.
- Clarified that status codes are for the Status Code field contained in the Status field.
- Clarified heading titles to cover all NVMe-MI Send/Receive command to Admin command mapping.
- Corrected the opcodes for the NVMe-MI Receive command and NVMe-MI Send command.
- Corrected the PEC values for Basic Management commands in Example 1 and Example 4.

Note:

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2023 NVM Express, Inc.

BLACK text indicates unchanged text. **BLUE** text indicates newly inserted text. **RED stricken** text indicates deleted text. **ORANGE** text indicates changes from another ECN. **Purple** text indicates the destination of moved text without changes. **Purple stricken** text indicates the source of moved text without changes. **GREEN** text indicates editor notes.

Description of NVM Express Base Specification 2.0c changes

Modify a portion of section 2 as shown below:

2 Theory of Operation

The NVM Express scalable interface is designed to address the needs of storage 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 65,535 ~~64 Ki-1~~ outstanding commands per I/O Queue. Additionally, support has been added for many Enterprise capabilities like end-to-end data protection (compatible with SCSI Protection Information, commonly known as T10 DIF, and SNIA DIX standards), enhanced error reporting, and virtualization.

...

2.3 NVM Storage Model

2.3.1 Storage Entities

The NVM storage model includes the following entities:

- NVM subsystems (refer to 1.5.40);
- Domains (refer to section 3.2.4);
- Endurance Groups (refer to section 3.2.3);
- NVM Sets (refer to section 3.2.2); and
- Namespaces (refer to section 3.2.1). <add a period>

As illustrated below, each domain is contained in a single NVM subsystem, each Endurance Group is contained in a single domain, each NVM Set is contained in a single Endurance Group, and each namespace is contained in a single NVM Set. Each Media Unit is contained in a single Endurance Group.

Each Endurance Group is composed of storage media, which are termed Media Units (refer to section 8.3.2). For clarity, Media Units are not shown in the examples that follow.

Figure 11 shows the hierarchical relationships of these entities within a simple NVM subsystem, which has:

- one domain;
- one Endurance Group;
- one NVM Set; and
- one namespace.

Figure 11: NVM Storage Hierarchy

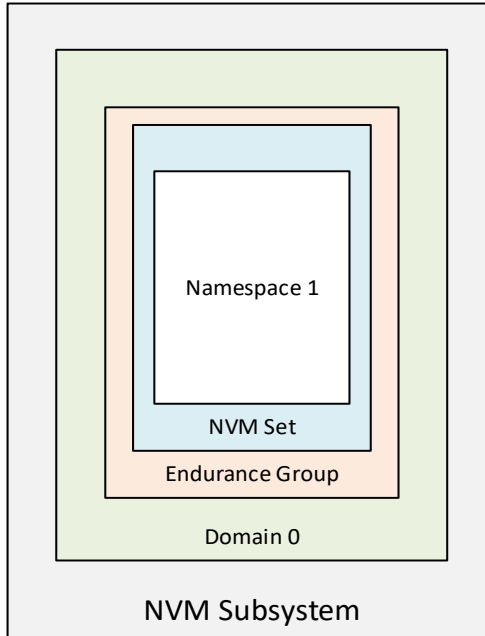
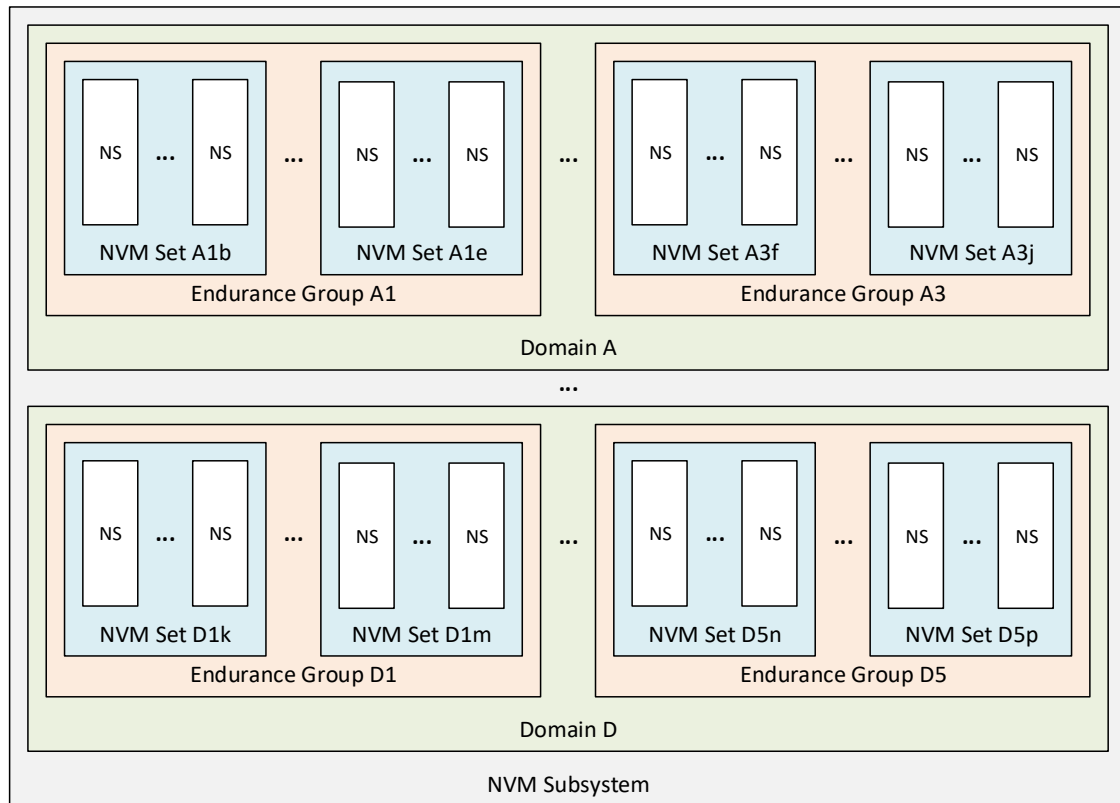


Figure 12 shows the relationships of these entities in a complex NVM subsystem, which has:

- multiple domains;
- multiple Endurance Groups per domain;
- multiple NVM Sets per Endurance Group; and
- multiple namespaces per NVM Set.

Figure 12: Complex NVM Storage Hierarchy



...

Modify a portion of section 3 as shown below:

3 NVM Express Architecture

...

3.1 NVM Controller Architecture

...

3.1.2 Controller Types

...

3.1.2.1 I/O Controller

...

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2023 NVM Express, Inc.

3.1.2.1.2 Log Page Support

...

Figure 24: I/O Controller – Log Page Support

Log Page Name	Log Page Support Requirements ¹
...	
Feature Identifiers Supported and Effects	M ³
NVMe-MI Commands Supported and Effects	M ³ , TBD
Command and Feature Lockdown	O
...	
Notes:	
1. O/M/P definition: O = Optional, M = Mandatory, P = Prohibited	
2. Mandatory for controllers that support Fixed Capacity Management (refer to section 8.3.2).	
3. Optional for NVM Express revision 1.4 and earlier.	
TBD. Optional if the NVMe-MI Send command and the NVMe-MI Receive command are not supported (refer to Figure 22).	

...

3.1.3 Controller Properties

...

3.1.3.11 Offset 38h: CMBLOC – Controller Memory Buffer Location

...

Figure 52: Offset 38h: CMBLOC – Controller Memory Buffer Location

Bits	Type	Reset	Description
...			
05	RO	Impl Spec	CMB Data Pointer Mixed Locations Support (CDPMLS): If this bit is set to '1', then the restriction that for a particular PRP List or SGL associated with a single command, all memory that contains is associated with that particular PRP List or SGL shall reside in either the Controller Memory Buffer or outside the Controller Memory Buffer, is not enforced (refer to section 8.5). If this bit is cleared to '0', then that restriction is enforced.
...			

...

3.1.3.16 Offset 50h: CMBMSC – Controller Memory Buffer Memory Space Control

This optional property specifies how the controller references the Controller Memory Buffer with host-supplied addresses. If the controller supports the Controller Memory Buffer (CAP.CMBS), this property is mandatory. Otherwise, this property is reserved.

This property shall be reset by ~~neither Controller Reset nor Function Level Reset, but it shall be reset by all other~~ Controller Level Resets ~~other than Controller Level Resets caused by:~~

- a Controller Reset; and
- a Function Level Reset.

...

3.2 NVM Subsystem Entities

...

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2023 NVM Express, Inc.

3.2.2 NVM Sets

...

There is a subset of Admin commands that are NVM Set aware as described in Figure 73.

Figure 73: NVM Set Aware Admin Commands

Admin Command	Details
Identify	<ul style="list-style-type: none">The Identify Namespace data structure includes the associated NVM Set Identifier.The NVM Set List data structure includes attributes for each NVM Set.
Capacity Management	<ul style="list-style-type: none">The Create NVM Set action returns the NVM Set Identifier of the NVM Set that is created.The Delete NVM Set action includes the NVM Set Identifier of the NVM Set that is to be deleted.
Namespace Management	<ul style="list-style-type: none">The create action includes the NVM Set Identifier as a host specified field.
Get Features and Set Features	<ul style="list-style-type: none">The Read Recovery Level Feature specifies the associated NVM Set Identifier.The Predictable Latency Mode Config Feature specifies the associated NVM Set Identifier.The Predictable Latency Mode Window Feature specifies the associated NVM Set Identifier.
Create I/O Submission Queue	<ul style="list-style-type: none">The Create I/O Submission Queue command includes the associated NVM Set Identifier.
Get Log Page	<ul style="list-style-type: none">The Predictable Latency Per NVM Set log page specifies the associated NVM Set Identifier.

...

3.3 NVM Queue Models

...

3.3.3 Queueing Data Structures

...

3.3.3.2 Common Completion Queue Entry

...

Figure 91: Completion Queue Entry: DW 3

Bits	Description
31:17	Status : Indicates the status for the command that is being completed. Refer to section 3.3.3.2.1.
16	Phase Tag (P) : Identifies whether a completion queue entry is new. Refer to section 3.3.3.2.2. This is a reserved bit in NVMe over Fabrics implementations.
15:00	Command Identifier (CID) : Indicates the identifier of the command that is being completed. This identifier is assigned by host software when the command is submitted to the Submission Queue. The combination of the SQ Identifier and Command Identifier uniquely identifies the command that is being completed. The maximum number of requests outstanding for a Submission Queue at one time is 65,535 64 Ki .

...

3.3.3.3 Queue Size

The Queue Size is indicated in a 16-bit 0's based field that indicates the number of slots in the queue. The minimum size for a queue is two slots. The maximum size for either an I/O Submission Queue or an I/O Completion Queue is defined as 65,536 ~~64 Ki~~ slots, limited by the maximum queue size supported by the controller that is reported in the CAP.MQES field. The maximum size for the Admin Submission and Admin

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2023 NVM Express, Inc.

Completion Queue is defined as 4,096 ~~4-Ki~~ slots. One slot in each queue is not available for use due to Head and Tail entry pointer definition.

...

3.5 Controller Initialization

...

3.5.2 Message-based Transport Controller Initialization

...

The controller initialization steps after an association is established are described below. For determining capabilities or configuring properties, the host uses the Property Get **command** and Property Set **command**s, respectively.

...

3.5.3 Controller Ready Modes During Initialization

There are two controller ready modes:

- **Controller Ready With Media:** By the time the controller becomes ready (i.e., by the time that CSTS.RDY transitions from '0' to '1') after the controller is enabled (i.e., CC.EN transitions from '0' to '1'), then:
 - a) the controller shall be able to process all commands without error as described in section 3.5.4.1; and
 - b) all namespaces attached to the controller and all media required to process Admin commands shall be ready (i.e., commands are not permitted to be aborted with a status code of Namespace Not Ready with the Do Not Retry bit cleared to '0' or Admin Command Media Not Ready with the Do Not Retry bit cleared to '0').
- **Controller Ready Independent of Media:** After the controller is enabled, all namespaces attached to the controller and media required to process Admin commands may or may not become ready by the time the controller becomes ready. Any **Admin NVM**-command or **I/O command** that specifies one or more namespaces attached to the controller is permitted to be aborted with a status code of Namespace Not Ready with the Do Not Retry bit cleared to '0' until CRYPTO.CRWMT amount of time after the controller is enabled.

Admin commands that require access to the media are permitted to be aborted with a status code of Admin Command Media Not Ready with the Do Not Retry bit cleared to '0' until CRYPTO.CRWMT amount of time after the controller is enabled. Refer to Figure 104 for a list of Admin commands that are permitted to be aborted with a status code of Admin Command Media Not Ready.

The controller shall be able to process without error as described in section 3.5.4.1:

- a) all Admin commands not listed in Figure 104 by the time the controller is ready;
- b) all Admin commands listed in Figure 104 no later than CRYPTO.CRWMT amount of time after the controller is enabled; and
- c) all **I/O NVM** commands no later than CRYPTO.CRWMT amount of time after the controller is enabled.

...

3.11 Firmware Update Process

The process for a firmware update to be activated in a domain (refer to section 3.2.4) by a reset is:

1. The host issues a Firmware Image Download command to download the firmware image to a controller. There may be multiple portions of the firmware image to download, thus the offset for each portion of the firmware image being downloaded on that controller is specified in the Firmware Image Download command. The data provided in the Firmware Image Download command should

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2023 NVM Express, Inc.

- conform to the Firmware Update Granularity indicated in the Identify Controller data structure or the firmware update may fail;
2. After the firmware is downloaded to that controller, the next step is for the host to submit a Firmware Commit command to that controller. The Firmware Commit command verifies that the last firmware image downloaded is valid and commits that firmware image to the firmware slot indicated for future use. A firmware image that does not start at offset zero, contains gaps, or contains overlapping regions is considered invalid. A controller may employ additional vendor specific means (e.g., checksum, CRC, cryptographic hash, or a digital signature) to determine the validity of a firmware image:
 - a. The Firmware Commit command may also be used to activate a firmware image associated with a previously committed firmware slot;
 3. The ~~host last step is to~~ performs a reset ~~on that controller to that then~~ causes the firmware image specified in the Firmware Slot field in the Firmware Commit command to be activated. The reset may be an NVM Subsystem Reset, Conventional Reset, Function Level Reset, or Controller Reset (CC.EN transitions from '1' to '0'):
 - a. In some cases a Conventional Reset or NVM Subsystem Reset is required to activate a firmware image. This requirement is indicated by Firmware Commit command specific status (refer to section 5.12.1);

and
 4. After the reset has completed, host software re-initializes the controller. This includes re-allocating I/O Submission and Completion Queues. Refer to sections 3.5.1 and 3.5.2.

...

Modify a portion of section 5 as shown below:

5 Admin Command Set

...

5.2 Asynchronous Event Request command

...

Asynchronous events are grouped into event types. The event type is indicated in the Asynchronous Event Type field in Dword 0 of the completion queue entry for the Asynchronous Event Request command. When the controller posts a completion queue entry for an outstanding Asynchronous Event Request command and thus reports an asynchronous event, subsequent events of that event type are automatically masked by the controller until the host clears that event. Unless otherwise stated, an event is cleared by reading the log page associated with that event ~~using the Get Log Page command~~ (refer to section 5.16). If that log page is not accessible because media is not ready (i.e., the controller aborts the Get Log Page command with a status code of Admin Command Media Not Ready), then the controller shall not post a completion queue entry for that asynchronous event until the controller is able to successfully return the log page that is required to be read to clear the asynchronous event.

...

5.13 Firmware Image Download command

...

After downloading an image, host software issues a Firmware Commit command before downloading another image. Processing of the first Firmware Image Download command after completion of a Firmware Commit command shall cause the controller to discard all remaining portion(s), if any, of downloaded images. If a ~~Controller Level Reset~~ ~~reset~~ occurs between a firmware download and completion of the Firmware Commit command, then the controller shall discard all portion(s), if any, of downloaded images.

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2023 NVM Express, Inc.

...

5.16 Get Log Page command

...

5.16.1 Log Specific Information

...

5.16.1.12 Predictable Latency Event Aggregate (Log Identifier 0Bh)

...

If there is an enabled Predictable Latency Event pending for an NVM Set, then the Predictable Latency Event Aggregate log page includes an entry for that NVM Set. The log page is an ordered list by NVM Set Identifier. For example, if Predictable Latency Events are pending for NVM Set 27, 13, and 17, then the log page shall have entries in numerical order of 13, 17, and 27. A particular NVM Set is removed from this log page after the Get Log Page command ~~is completed successfully~~ with the Retain Asynchronous Event bit cleared to '0' for the Predictable Latency Per NVM Set log page for that NVM Set ~~is completed successfully~~.

...

5.16.1.19 NVMe-MI Commands Supported and Effects (Log Identifier 13h)

...

The ~~NVMe-MI~~ ~~NMVe-MI~~ Commands Supported and Effects data structure describes the overall possible effect of a Management Interface command using the using the NVMe-MI Send command, including any optional features of the command.

...

5.16.1.25 Sanitize Status (Log Identifier 81h)

...

Figure 267: Sanitize Status Log Page

Bytes	Description
...	
03:02	Sanitize Status (SSTAT): This field indicates the status associated with the most recent sanitize operation. ...
...	

...

5.17 Identify command

...

5.17.2 Identify Data Structures

...

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2023 NVM Express, Inc.

5.17.2.1 Identify Controller Data Structure (CNS 01h)

...

Figure 275: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O ¹	Admin ¹	Disc ¹	Description
Controller Capabilities and Features				
...				
Admin Command Set Attributes & Optional Controller Capabilities				
257:256	M	M	R	<p>Optional Admin Command Support (OACS): This field indicates the optional Admin commands and features supported by the controller. Refer to section 3.1.2.</p> <p>...</p> <p>Bit 6 if set to '1', then the controller supports the NVMe-MI Send command and NVMe-MI Receive commands. If cleared to '0', then the controller does not support the NVMe-MI Send command and NVMe-MI Receive commands.</p> <p>...</p> <p>Bit 2 if set to '1', then the controller supports the Firmware Commit command and Firmware Image Download commands. If cleared to '0', then the controller does not support the Firmware Commit command and Firmware Image Download commands.</p> <p>...</p> <p>Bit 0 if set to '1', then the controller supports the Security Send command and Security Receive commands. If cleared to '0', then the controller does not support the Security Send command and Security Receive commands.</p>
...				
521:520	M	M	R	<p>Optional NVM Command Support (ONCS): This field indicates the optional I/O commands and features supported by the controller. Refer to section 3.1.2.</p> <p>If a controller supports more than one I/O Command Set, then the bits in the field associated with command support reflect the aggregate support across all of the I/O Command Sets supported or I/O Command Sets inherited from I/O Command Sets (e.g., Zoned Namespace Command Set). The Commands Supported and Effects log page (refer to section 5.16.1.6) may be used to determine the support of commands for a specific I/O Command Set.</p> <p>Bits 15:9 are reserved.</p> <p>...</p>
...				

...

5.27 Set Features command

...

5.27.1 Feature Specific Information

...

5.27.1.10 Host Memory Buffer (Feature Identifier 0Dh)

...

Figure 334: Host Memory Buffer – Command Dword 15

Bits	Description
31:00	Host Memory Descriptor List Entry Count (HMDLEC): This field specifies the number of entries provided in the Host Memory Descriptor List.

If the host specifies the Host Memory Descriptor List Entry Count field cleared to 0h, then the controller should <Editor: shall in the next major release> abort the command with a status code of Invalid Field in Command.

...

Figure 336: Host Memory Buffer – Host Memory Buffer Descriptor Entry

Bits	Description
127:96	Reserved
95:64	Buffer Size (BSIZE): Indicates the number of contiguous memory page size (CC.MPS) units for this descriptor. <i>If this field is cleared to 0h, then the controller shall ignore this descriptor.</i>
63:00	Buffer Address (BADD): Indicates the host memory address for this descriptor aligned to the memory page size (CC.MPS). The least significant bits (<i>n</i> :0) of this field indicate the offset within the memory page is 0h (e.g., if the memory page size is 4 KiB, then bits 11:00 shall be 0h; if the memory page size is 8 KiB, then bits 12:00 shall be 0h).

...

5.27.1.23 Host Metadata (Feature Identifier 7Dh), (Feature Identifier 7Eh), (Feature Identifier 7Fh)

...

Figure 358: Get Features – Command Dword 11

Bits	Description
31:01	Reserved

Figure 358: Get Features – Command Dword 11

Bits	Description
00	<p>Generate Default Host Metadata (GDHM): If this bit is set to '1', then the controller shall modify the default value of the specified Host Metadata feature by creating and returning generate a number of vendor specific strings for the Element Types of the specified Host Metadata feature value. The number of vendor specific strings created and returned is implementation specific. The controller is allowed to return:</p> <ul style="list-style-type: none"> no vendor specific strings; vendor specific strings created for a subset of the defined Element Types of the specified Host Metadata feature; or vendor specific strings created for each of the defined Element Types of the specified Host Metadata feature. <p>These vendor specific strings replace the Metadata Element Descriptors in the default Host Metadata data structure returned by the specified Host Metadata feature, when a Get Features command for a Host Metadata feature with the SEL field set to 001b (i.e., Default) is submitted, until a Controller Level Reset occurs (i.e., the replacement default values are not persistent across a Controller Level Reset).</p> <p>If the generated vendor specific string's Metadata Element Descriptor does not exist for the Host Metadata Data Structure that contains the default value of the specified Host Metadata Feature value, then the controller shall create the Metadata Element Descriptor in the Host Metadata Data Structure that contains the default value with the generated vendor specific string.</p> <p>If the generated vendor specific string's Metadata Element Descriptor does exist for the Host Metadata Data Structure that contains the default value of the specified Host Metadata Feature value, then the controller shall replace the Metadata Element Descriptor with the generated vendor specific string.</p> <p>If the number of vendor specific strings generated is 0h, then the default value for the Number of Metadata Element Descriptors for the specified Host Metadata feature shall be 0h. If the number of vendor specific strings generated is not 0h, then the Host Metadata Data Structure that contains the default value for the Number of Metadata Element Descriptors of the specified Host Metadata Feature value shall be the number of vendor specific strings created.</p> <p>If this bit is cleared to '0', then the controller shall not modify the default value generate any vendor specific strings for the Element Types of the specified Host Metadata feature.</p> <p>If this bit is cleared to '0' and a Get Features command for a Host Metadata feature with the SEL field set to 001b (i.e., Default) is submitted, then the controller shall return the currently existing modified default value, if any, for that Host Metadata feature (i.e., the updated default value that was created by the last Get Features command, with the GDHM bit set to '1', that completed successfully since the last Controller Level Reset).</p>

The host issues a Set Features command specifying one of the Host Metadata features containing a Host Metadata data structure (refer to Figure 360). The host receives a Host Metadata data structure via the Get Features command. The content of the strings in the Host Metadata data structure are vendor specific.

If any Get Features command specifying the GDHM bit set to '1' returned a status code of Successful Completion since the last Controller Level Reset, then for any subsequent Get Features command that specifies a SEL field set to 001b (i.e., Default) and specifies a Host Metadata feature, the controller shall return the replaced default value containing the most recent vendor specific strings for that Host Metadata feature.

...

Modify a portion of section 8 as shown below:

8 Extended Capabilities

...

8.5 Controller Memory Buffer

...

The controller may support PRP Lists and SGLs in the Controller Memory Buffer. If the CMBLOC.CDPMLS bit (refer to Figure 52) is cleared to '0', then for a particular PRP List or SGL associated with a single command, all memory ~~containing associated with~~ the PRP List or SGL shall be either entirely located in the Controller Memory Buffer or entirely located outside the Controller Memory Buffer.

...

8.18 Replay Protected Memory Block

...

8.18.2 RPMB Operations

...

8.18.2.3 Authenticated Data Write

The Authenticated Data Write is initiated by a Security Send command. The RPMB Data Frame delivered from the host to the controller includes the Request Message Type = 0003h, Block Count, Address, Write Counter, Data and MAC.

When the controller receives this RPMB Data Frame, that controller first checks whether the Write Counter has expired. If the Write Counter has expired, then that controller sets the ~~RPMB Operation Result~~ to ~~0005h~~ 0085h (write failure, write counter expired) and no data is written to the RPMB data area.

...

8.19 Reservations

...

8.19.3 Registering

...

A host that is a registrant of a namespace may register the same reservation key value multiple times with the namespace on the same or different controllers. For a Reservation Register command with the RREGA field cleared to 000b:

- a) the IEKEY field shall be ignored; and
- b) if a host that is already a registrant of a namespace attempts to register with that namespace using a different ~~reservation registration~~ key value, then the command shall be aborted with a status code of Reservation Conflict.

...

8.19.7 Preempting a Reservation or Registration

...

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:

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2023 NVM Express, Inc.

- a) If the PRKEY field value matches the reservation key of the current reservation holder, then the following occur as an atomic operation:
- all registrants with a matching reservation 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;

...

8.24 Telemetry

...

The NVM subsystem is allowed to provide a Telemetry Host-Initiated log page per controller or a shared Telemetry Host-Initiated log page across all controllers in the NVM subsystem. If a shared Telemetry Host-Initiated log page is implemented, the Telemetry Host-Initiated Data Generation Number field in the Telemetry Host-Initiated log page is used to allow the host to detect that the Telemetry Host-Initiated log page has been changed by:

- a host through a different controller; or
- a Management Controller through a Management Endpoint (refer to the NVM Express Management Interface Specification).

...

Modify a portion of Annex B as shown below:

Annex B Host Considerations (Informative)

B.1 Basic Steps when Building a Command

When host software builds a command for the controller to execute, it first checks to make sure that the appropriate Submission Queue (SQ) is not full. The Submission Queue is full when the number of entries in the queue is one less than the queue size. Once an empty slot (pFreeSlot) is available:

1. Host software builds a command at SQ[pFreeSlot] with:
 - a. CDW0.OPC is set to the appropriate command to be executed by the controller;
 - b. CDW0.FUSE is set to the appropriate value, depending on whether the command is a fused operation;
 - c. CDW0.CID is set to a unique identifier for the command when combined with the Submission Queue identifier;
 - d. The Namespace Identifier, NSID field, is set to the namespace the command applies to;
 - e. MPTR shall be filled in with the offset to the beginning of the Metadata Region, if there is a data transfer and the namespace format contains metadata as a separate buffer;
 - f. PRP1 and/or PRP2 (or SGL Entry 1 if SGLs are used) are set to the source/destination of data transfer, if there is a data transfer; and
 - g. CDW10 – CDW15 are set to any command specific information;
2. Host software then completes a transport specific action in order to submit the command for processing.

...

Description of NVM Express NVM Command Set specification 1.0c changes

Modify a portion of section 1 as shown below:

1 Introduction

...

1.4 Definitions from the NVM Express Base Specification

...

1.4.2 Definitions in the NVM Express Base Specification specified in the NVM Command set

...

1.4.2.TBD1 logical block data size

The size in bytes of a logical block, excluding metadata, if any. The size is calculated using the following formula:

$$2^{\text{DataExponent}}$$

Where:

- DataExponent is the value in the LBA Data Size field in the NVM Command Set specific LBA Format data structure (refer to [Figure 98](#)).

1.4.2.TBD2 logical block size

The size in bytes of a logical block, including metadata size. The size is calculated using the following formula:

$$\text{logical block data size} + \text{MetadataBytes}$$

Where:

- logical block data size is defined in section [1.4.2.TBD1](#).
- MetadataBytes is the value in the Metadata Size field in the NVM Command Set specific LBA Format data structure.

...

Modify a portion of section 2 as shown below:

2 NVM Command Set Model

...

2.1 Theory of operation

An NVM subsystem is comprised of some number of controllers, where each controller may access some number of namespaces. For the NVM Command Set, each namespace is comprised 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 [data](#) sizes may be 512 bytes, 1 KiB,

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2023 NVM Express, Inc.

2 KiB, 4 KiB, 8 KiB, etc. NVM Command Set commands are used to access and modify logical block contents within a namespace.

...

2.1.6 Metadata Region (MR)

...

The controller may support several physical formats of logical block **data** size and associated metadata size. There may be performance differences between different physical formats. This is indicated as part of the Identify Namespace data structure.

If the namespace is formatted to use end-to-end data protection (refer to section 5.2), then the ~~first bytes~~ ~~or~~ last bytes of the metadata is used for protection information (specified as part of the Format NVM command).

...

Modify a portion of section 3 as shown below:

3 I/O Commands for the NVM Command Set

...

3.2 NVM Command Set Commands

...

3.2.2 Copy command

...

The number of logical blocks written by the Copy command is the sum of all Number of Logical Blocks fields in all Source Range entries specified in the list of Source Range entries.

The data bytes in the LBAs specified by each Source Range Entry shall be copied to the destination LBA range in the same order those LBAs are listed in the Source Range entries (e.g., the LBAs specified by Source Range entry 0 are copied to the lowest numbered LBAs specified by the SDLBA field, the LBAs specified by Source Range entry 1 are copied to the next consecutively numbered LBAs specified by the SDLBA field). The read operations and write operations used to perform the copy may operate sequentially or in parallel.

The host should not specify a destination LBA range that overlaps the LBA in any of the Source Range entries. If the host specifies a destination LBA range that overlaps with any LBAs specified in one or more of the Source Range entries, then upon completion of the Copy command, the data stored in each logical block in that overlapping destination LBA range may, within the constraints of the atomicity rules described in section 2.1.4, be from any of the one or more Source Range entries in which that LBA is contained. This is a result of the possibility that overlapping Source Range entries may be processed in any order.

If the read portion of a copy operation attempts to access a deallocated or unwritten logical block, the controller shall operate as described in section 3.2.3.2.1.

...

3.2.2.1 Command Completion

When the command is completed, the controller shall post a completion queue entry to the associated I/O Completion Queue indicating the status for the command.

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2023 NVM Express, Inc.

If the command completes with failure, then:

- Dword 0 of the completion queue entry contains the number of the lowest numbered Source Range entry that was not successfully copied (e.g., if Source Range 0, Source Range 1, Source Range 2, and Source Range 5 are copied successfully and Source Range 3 and Source Range 4 are not copied successfully, then Dword 0 is set to 3); and
- prior to aborting the command, the controller may or may not have copied some (or all) of the data specified by that Copy command for the Source Ranges that have a number greater than or equal to the value in Dword 0.

If no data was written to the destination LBAs, then Dword 0 of the completion queue entry shall be cleared to 0h.

...

Figure 37: Copy – Command Specific Status Values

Value	Description
...	
83h	Command Size Limit Size Exceeded: One or more of the Copy command processing limits (i.e., non-zero value of the NR, MSSRL, and MCL fields in the Identify Namespace data structure) was exceeded.

...

Modify a portion of section 4 as shown below:

4 Admin Commands for the NVM Command Set

...

4.1 Admin Command behavior for the NVM Command Set

...

4.1.4 Get Log Page command

...

4.1.4.1 Error Information (Log Identifier 01h)

The Error Information log page is as defined in the NVM Express Base Specification. Figure 90 describes the NVM Command Set specific definition of the LBA field.

Figure 90: Error Information Log Entry Data Structure – User Data

Bytes	Description
23:16	LBA: This field indicates the first lowest-numbered LBA that experienced an the error condition, if applicable.

...

4.1.4.2 SMART / Health Information (02h)

The SMART / Health Information log page is as defined in the NVM Express Base Specification. For the Data Units Read and Data Units Written fields, when the logical block ~~LBA~~ size is a value other than 512 bytes, the controller shall convert the amount of data read to 512 byte units.

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2023 NVM Express, Inc.

...

4.1.5 Identify Command

...

4.1.5.1 NVM Command Set Identify Namespace Data Structure (CNS 00h)

...

Figure 97: Identify – Identify Namespace Data Structure, NVM Command Set

Bytes	O/M ¹	Description
07:00	M	Namespace Size (NSZE): This field indicates the total size of the namespace in logical blocks. A namespace of size n consists of LBA 0 through $(n - 1)$. The number of logical blocks is based on the formatted logical block LBA size.
15:08	M	Namespace Capacity (NCAP): This field indicates the maximum number of logical blocks that may be allocated in the namespace at any point in time. The number of logical blocks is based on the logical block LBA size. Spare LBAs are not reported as part of this field. Refer to section 2.1.1 for details on the usage of this field.
23:16	M	Namespace Utilization (NUSE): This field indicates the current number of logical blocks allocated in the namespace. This field is less than or equal to the Namespace Capacity. The number of logical blocks is based on the formatted logical block LBA size. Refer to section 2.1.1 for details on the usage of this field.
...		
25	M	Number of LBA Formats (NLBAF): This field defines the number of supported LBA data size and metadata size combinations supported by the namespace. LBA formats shall be packed sequentially starting at the LBA Format 0 Support (LBAF0) field. This is a 0's based value. The maximum number of LBA formats that may be indicated as supported is: <ul style="list-style-type: none"> a) 16 if the LBA Format Extension Enable (LBAFEE) field is cleared to 0h in the Host Behavior Support feature (refer to the Host Behavior Support section in the NVM Express Base Specification); or b) 64 if the LBAFEE field is set to 1h in the Host Behavior Support feature (refer to the Host Behavior Support section in the NVM Express Base Specification). The supported LBA formats are indicated in bytes 128 to 383 in this data structure. The LBA Format fields with an index beyond the value set in this field are invalid and not supported. LBA Formats that are valid, but not currently available may be indicated by setting the LBA Data Size for that LBA Format to 0h. The metadata may be either transferred as part of the logical block LBA (creating an extended LBA which is a larger LBA size that is exposed to the application) or may be transferred as a separate contiguous buffer of data. The metadata shall not be split between the logical block LBA and a separate metadata buffer. Refer to section 2.1.6. It is recommended that software and controllers transition to a logical block 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 5.8.3.
...		
35:34	O	Namespace Atomic Write Unit Normal (NAWUN): This field indicates the namespace specific size of the write operation guaranteed to be written atomically to the NVM during normal operation. If the NSABP bit is cleared to '0', then this field is reserved. A value of 0h indicates that the size for this namespace is the same size as that reported in the AWUN field of the Identify Controller data structure. All other values specify a size in terms of logical blocks using the same encoding as the AWUN field (i.e., with the exception of the value 0, this field is a 0's based value) . Refer to section 2.1.4.

Figure 97: Identify – Identify Namespace Data Structure, NVM Command Set

Bytes	O/M ¹	Description
37:36	O	<p>Namespace Atomic Write Unit Power Fail (NAWUPF): This field indicates the namespace specific size of the write operation guaranteed to be written atomically to the NVM during a power fail or error condition. If the NSABP bit is cleared to '0', then this field is reserved.</p> <p>A value of 0h indicates that the size for this namespace is the same size as that reported in the AWUPF field of the Identify Controller data structure. All other values specify a size in terms of logical blocks using the same encoding as the AWUPF field (i.e., with the exception of the value 0, this field is a 0's based value). Refer to section 2.1.4.</p>
39:38	O	<p>Namespace Atomic Compare & Write Unit (NACWU): This field indicates the namespace specific size of the write operation guaranteed to be written atomically to the NVM for a Compare and Write fused command. If the NSABP bit is cleared to '0', then this field is reserved.</p> <p>A value of 0h indicates that the size for this namespace is the same size as that reported in the ACWU field of the Identify Controller data structure. All other values specify a size in terms of logical blocks using the same encoding as the ACWU field (i.e., with the exception of the value 0, this field is a 0's based value). Refer to section 2.1.4.</p>
41:40	O	<p>Namespace Atomic Boundary Size Normal (NABSN): This field indicates the atomic boundary size for this namespace for the NAWUN value. This field is specified in logical blocks. Writes to this namespace that cross atomic boundaries are not guaranteed to be atomic to the NVM with respect to other read or write commands.</p> <p>A value of 0h indicates that there are no atomic boundaries for normal write operations. All other values specify a size in terms of logical blocks using the same encoding as the AWUN field (i.e., with the exception of the value 0, this field is a 0's based value). Refer to section 2.1.4.</p> <p>Refer to section 5.8.2 for how this field is utilized.</p>
43:42	O	<p>Namespace Atomic Boundary Offset (NABO): This field indicates the LBA on this namespace where the first atomic boundary starts.</p> <p>If the NABSN and NABSPF fields are cleared to 0h, then the NABO field shall be cleared to 0h. NABO shall be less than or equal to NABSN and NABSPF. Refer to section 2.1.4.</p> <p>Refer to section 5.8.2 for how this field is utilized.</p>
45:44	O	<p>Namespace Atomic Boundary Size Power Fail (NABSPF): This field indicates the atomic boundary size for this namespace specific to the Namespace Atomic Write Unit Power Fail value. This field is specified in logical blocks. Writes to this namespace that cross atomic boundaries are not guaranteed to be atomic with respect to other read or write commands and there is no guarantee of data returned on subsequent reads of the associated logical blocks.</p> <p>A value of 0h indicates that there are no atomic boundaries for power fail or error conditions. All other values specify a size in terms of logical blocks using the same encoding as the AWUPF field (i.e., with the exception of the value 0, this field is a 0's based value). Refer to section 2.1.4.</p>
...		

...

4.1.5.2 I/O Command Set specific fields within Identify Controller data structure (CNS 01h)

The following table describes the NVM Command Set specific fields within the Identify Controller data structure described in the NVM Express Base Specification.

Figure 99: Identify – Identify Controller data structure, NVM Command Set Specific Fields

Bytes	O/M ¹	Description
527:526	M	<p>Atomic Write Unit Normal (AWUN): 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 during normal operation. This field is specified in logical blocks and is a 0's based value.</p> <p>If a specific namespace guarantees a larger size than is reported in this field, then this namespace specific size is reported in the NAWUN field in the Identify Namespace data structure. Refer to section 2.1.4.</p> <p>If a write command is submitted with size less than or equal to the AWUN value, the host is guaranteed that the write command is atomic to the NVM with respect to other read or write commands. If a write command is submitted with size greater than the AWUN value, then there is no guarantee of command atomicity. AWUN does not have any applicability to write errors caused by power failure (refer to Atomic Write Unit Power Fail).</p> <p>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 logical block LBA size).</p>
...		

...

5 Extended Capabilities

...

5.2 End-to-end Data Protection

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 [the host software](#) to determine the integrity of the logical block. This additional protection information, if present, is either the first bytes of metadata or the last bytes of metadata, based on the format of the namespace (refer to the PIP bit in the DPS field shown in Figure 97). If the [value in the](#) Metadata Size field (refer to [Figure 98 404](#)) is greater than the number of bytes of protection information and the protection information is contained in the first bytes of the metadata, then the CRC does not cover any metadata bytes. If the Metadata Size is greater than the number of bytes of protection information and the protection information is contained in the last bytes of the metadata, then the CRC covers all metadata bytes up to but excluding the protection information. As described in section 5.8.3, metadata and hence this protection information may be configured to be contiguous with the logical block data or stored in a separate buffer.

...

5.2.1.2 32b Guard Protection Information

The 32b Guard Protection Information is shown in Figure 116 and is contained in the metadata associated with each logical block. The 32b Guard Protection Information shall only be available to namespaces that have [a logical block an LBA](#) size (refer to the LBADS field in Figure 98) greater than or equal to 4 KiB.

...

5.2.1.3 64b Guard Protection Information

The 64b Guard Protection Information is shown in Figure 118 and is contained in the metadata associated with each logical block. 64b Guard Protection Information shall only be available to namespaces that have a logical block ~~an~~ LBA size (refer to the LBADS field in Figure 98) greater than or equal to 4 KiB.

...

5.2.1.4 Storage Tag and Logical Block Reference Tag from Storage and Reference Space

...

5.2.1.4.1 Storage Tag Ffield and Logical Block Reference Tag Ffield

...

For an example of the 16b Guard Protection Information usage of Command Dword 2, Command Dword 3, and Command Dword 14 assume a namespace is formatted with the following:

- a) LBA Data Size field (refer to Figure 98404) cleared to 0h specifying a 512B logical block data LBA size;
- b) Metadata Size field (refer to Figure 98404) set to 8h specifying an 8B metadata size;
- c) ...

...

For an example of the 32b Guard Protection Information usage of Command Dword 2, Command Dword 3, and Command Dword 14 assume a namespace is formatted with the following:

- a) LBA Data Size field (refer to Figure 98404) set to Ch specifying a 4 KiB logical block data LBA size;
- b) Metadata Size field (refer to Figure 98404) set to 10h specifying a 16B metadata size;
- c) ...

...

For an example of the 64b Guard Protection Information usage of Command Dword 2, Command Dword 3, and Command Dword 14 assume a namespace is formatted with the following:

- a) LBA Data Size field (refer to Figure 98404) set to Ch specifying a 4 KiB logical block data LBA size;
- b) Metadata Size field (refer to Figure 98404) set to 10h specifying a 16B metadata size;
- c) ...

...

5.2.3 Control of Protection Information Checking - PRCHK

Checking of protection information consists of the following operations performed by the controller.

- ...
- If the Reference Tag is defined (refer to Figure 101) then:
 - If the Reference Tag Check bit of the PRCHK field is set to '1' and the namespace is formatted for Type 1 or Type 2 protection, then the controller compares the Logical Block Reference Tag to the computed reference tag. The computed reference tag depends on the Protection Type:
 - If the namespace is formatted for Type 1 protection, the value of the computed reference tag for the first logical block of the command is the value contained in the Initial Logical Block Reference Tag (ILBRT) or Expected Initial Logical Block Reference Tag (EILBRT) field in the command, and the computed reference tag is incremented for each subsequent logical block. The controller shall complete the

command with a status of Invalid Protection Information if the ILBRT field or the EILBRT field does not match the value of the least-significant bits four bytes of the SLBA field sized to the number of bits in the Logical Block Reference Tag (refer to section 5.2.1.4).

Note: Unlike SCSI Protection Information Type 1 protection which implicitly uses the least significant four bytes of the LBA, the controller always uses the ILBRT or EILBRT field and requires the host software to initialize the ILBRT or EILBRT field to the least significant bits of the LBA sized to the number of bits in the Logical Block Reference Tag (refer to section 5.2.1.4) when Type 1 protection is used.

- If the namespace is formatted for Type 2 protection, the value of the computed reference tag for the first logical block of the command is the value contained in the Initial Logical Block Reference Tag (ILBRT) or Expected Initial Logical Block Reference Tag (EILBRT) field in the command, and the computed reference tag is incremented for each subsequent logical block.

○ ...

...

5.3 Namespace Management

...

If a Namespace Management command create operation specifies values such that:

- a) the product of NSZE and the logical block size Formatted LBA Size value is an integral multiple of the Namespace Size Granularity;
- b) the product of NCAP and the logical block size Formatted LBA Size value is an integral multiple of the Namespace Capacity Granularity; and
- c) NSZE is equal to NCAP,

...

5.8 Command Set Specific Capability

5.8.1 Get LBA Status

Potentially Unrecoverable LBAs are LBAs that, when read, may result in the command that caused the media to be read being aborted with a status code of Unrecovered Read Error. The Get LBA Status capability provides the host with the ability to identify Potentially Unrecoverable LBAs. The logical block data and metadata, if any, are able to be recovered from another location and re-written.

...

Description of NVM Express Zoned Namespace Command Set specification 1.1c changes

Modify a portion of section 2 as shown below:

2 Zoned Namespace Command Set Model

...

2.1 Theory of operation

...

2.1.1 Namespaces

...

2.1.1.3 Zone State Machine

There is a state machine associated with each zone. The state machine controls the operational characteristics of each zone. The state machine consists of the following states: ZSE:Empty, ZSIO:Implicitly Opened, ZSEO:Explicitly Opened, ZSC:Closed, ZSF:Full, ZSRO:Read Only, and ZSO:Offline.

The initial state of a zone state machine is set as a result of:

- a) an NVM Subsystem Reset; or
- b) all controllers in the NVM subsystem reporting controller shutdown processing complete (i.e., the Shutdown Type (ST) bit cleared to '0' and the Shutdown Status (SHST) field set to 10b, refer to the NVM Express Base Specification).

The initial state for each zone is the:

- a) ZSE:Empty state, if the write pointer is valid, the write pointer points to the lowest LBA in the zone, and the Zone Descriptor Extension Valid bit is cleared to '0';
- b) ZSC:Closed state:
 - 1. ~~a)~~ if the write pointer is valid and does not point to the lowest LBA in the zone; or
 - 2. ~~b)~~ if the write pointer is valid and the Zone Descriptor Extension Valid bit is set to '1';
- c) ZSF:Full state:
 - 1. ~~a)~~ if the most recent state was the ZSF:Full; or
 - 2. ~~b)~~ if the zone state was transitioned to the ZSF:Full state as a result of the NVM Subsystem Reset;
- d) ZSRO:Read Only state, if the most recent zone state was the ZSRO:Read Only state; and
- e) ZSO:Offline state, if the most recent zone state was the ZSO:Offline state.

...

Modify a portion of section 3 as shown below:

3 I/O Commands for the Zoned Namespace Command Set

...

3.4 Zoned Namespace Command Set I/O Commands

3.4.1 Zone Append command

...

3.4.1.1 Protection Information

For the Zone Append command, the actual LBA where data is written by the command is not known to the host until the Zone Append command completes (refer to the ALBA field defined in section 3.4.1.2). The host is not able to provide the actual LBA of the data in the reference tags in the Protection Information at the time the Zone Append command is issued (refer to the End-to-end Data Protection section of the NVM Command Set Specification). As a result, handling of LBA based reference tags in the Protection Information is handled as defined in this section.

Unless otherwise specified, for the protection information received by the controller from the host, the checks performed by the controller shall be performed as defined in the Control of Protection Information Checking – PRCHK section of the NVM Command Set Specification.

For the Reference Tag written to the media, if the PIREMAP bit is:

- a) cleared to '0', then the controller shall write the Reference Tag to the media per the End-to-end Protection Information section of the NVM Command Set Specification without modification; or
- b) set to '1', then the controller shall write the Logical Block Reference Tag to media for the first and subsequent LBAs as follows:
 - A) Media Reference Tag[0] = ILBRT + (ALBA - ZSLBA); and
 - B) Media Reference Tag[n+1] = Media Reference Tag[n] + 1

Where: Media Reference Tag [0] is the Logical Block Reference Tag written to media for the first LBA, and Media Reference Tag [n+1] is the Logical Block Reference Tag written to media for all other LBAs.

If the specified zoned namespace is formatted for Type 1 protection and the PIREMAP bit is set to '1', then the host should initialize the ILBRT field to the least-significant bits of the Zone Start Logical Block Address (ZSLBA) field sized to the number of bits in the Logical Block Reference Tag (refer to the Storage Tag and Logical Block Reference Tag from Storage and Reference Space section in the NVM Command Set Specification). If the value in the ILBRT field does not match the value in the ZSLBA field sized to the number of bits in the Logical Block Reference Tag, then the controller shall abort the command with a status code of Invalid Protection Information.

Note: When writing protection information using the Zone Append command with the PIREMAP bit set to '1', the Logical Block Reference Tag written to media is not the value transferred from the host to the controller, but contains the value calculated as defined in this section.

The controller shall write all protection information received from the host, other than the Logical Block Reference Tag, to the media without modification.

...

Description of NVM Express Management Interface Specification

1.2c changes

Modify a portion of section 4 as shown below:

4 Message Servicing Model

...

4.3 In-Band Tunneling Message Servicing Model

...

4.3.1 NVMe-MI Send Command

...

4.3.1.1 NVMe-MI Send Command ~~Request Message~~ to NVMe Admin Command ~~SQE~~ Mapping

In order to tunnel an NVMe-MI Command in-band via NVMe-MI Send, an NVMe-MI Request Message is mapped onto an NVMe Submission Queue Entry (SQE) as shown pictorially in Figure 46 and in table form in Figure 47. An NVMe-MI Response Message is mapped on to an NVMe Completion Queue Entry (CQE) as shown pictorially in Figure 48 and in table form in Figure 49. Refer to the NVM Express Base Specification for details on an NVMe Submission Queue Entry and an NVMe Completion Queue Entry.

...

4.3.1.2 NVMe-MI Send Command Servicing Model

The NVMe-MI Send command servicing model is illustrated in Figure 51 as a series of phases and NVMe/NVMe-MI Contexts. The phases of the NVMe-MI Send command servicing model are further described in this section. The behavior of the portions of the figure in the NVMe Context are specified by the NVM Express Base Specification. The behavior of the portions of the figure in the NVMe-MI Context are specified by this specification. The phases and NVMe/NVMe-MI Contexts are logical constructs that illustrate the NVMe-MI Send command servicing model and do not mandate a particular implementation.

This section describes the NVMe-MI Send command servicing model starting at NVMe Processing as shown in phase 1 of Figure 51. In phase 1, CDW0 to CDW9 are checked for errors per the NVM Express Base Specification. If any errors are encountered in CDW0 to CDW9, then the NVMe-MI Send command is completed with an error status code in the Status Code field contained in the Status field as per the NVM Express Base Specification and the Tunneled Status and Tunneled NVMe Management Response fields shall be cleared to 0h.

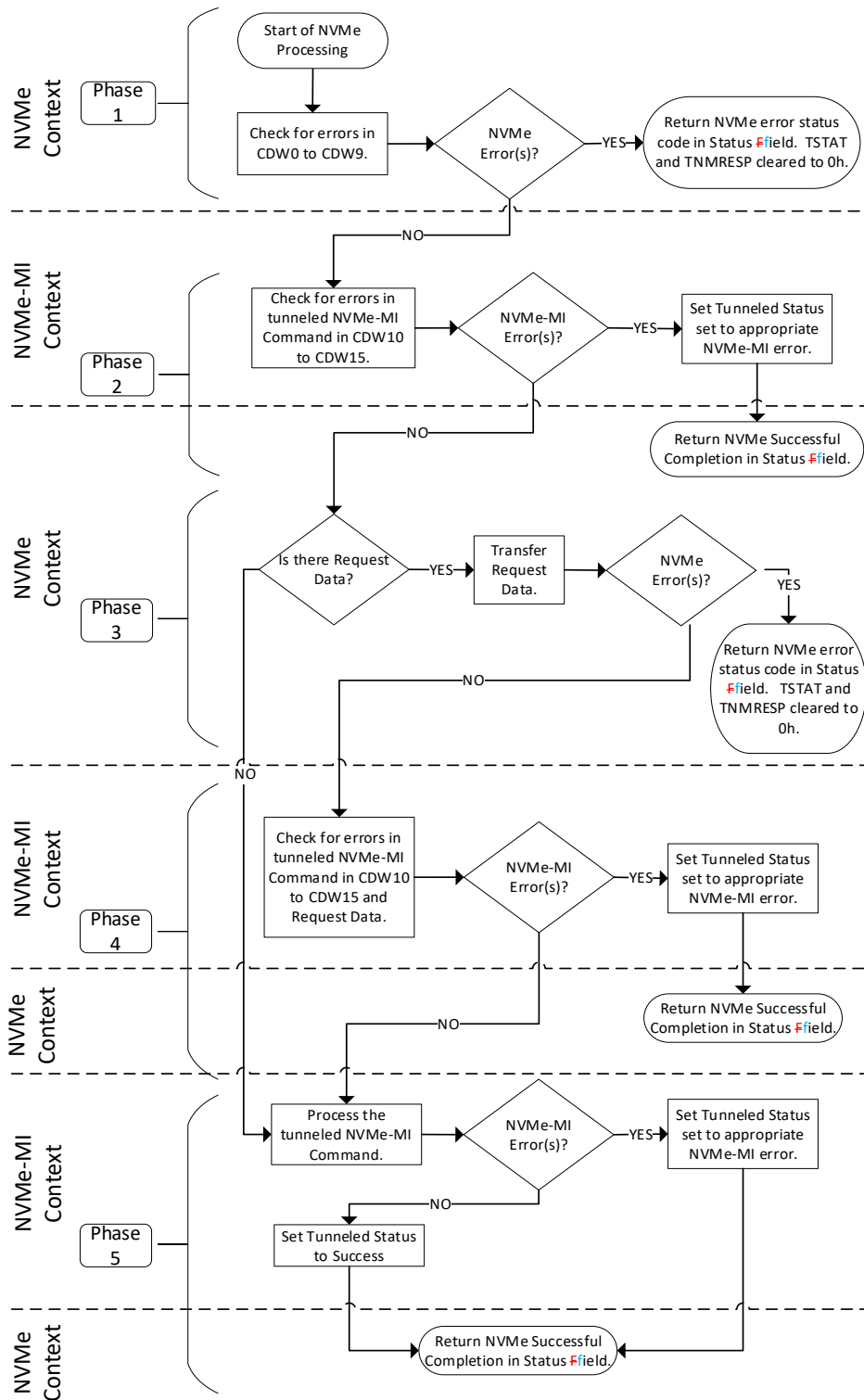
If there are no errors in CDW0 to CDW9, then command servicing enters phase 2 where the portion of the tunneled NVMe-MI Command in CDW10 to CDW15 is checked for errors. Note that if there is no Request Data, then CDW10 to CDW15 contain the entire tunneled NVMe-MI Command. If any errors are encountered in the portion of the tunneled NVMe-MI Command in CDW10 to CDW15, then the NVMe-MI Send command is completed with a status code of Successful Completion in the Status field as defined in the NVM Express Base Specification. The Tunneled Status field contains the error Response Message Status for the portion of the tunneled NVMe-MI Command in CDW10 to CDW15 and the Tunneled NVMe Management Response field contains the NVMe Management Response field from the NVMe-MI Command that is being tunneled in-band.

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2023 NVM Express, Inc.

If there are no errors in phase 2, then command servicing enters phase 3 where there is a check to determine if there is any Request Data for the tunneled NVMe-MI Command. If there is no Request Data for the tunneled NVMe-MI Command, then command servicing skips to phase 5. If there is Request Data, then the Request Data is transferred from the buffer pointed to by DPTR. If any errors are encountered transferring the Request Data, then the command is completed with an error status code [in the Status Code field contained](#) in the Status field as per the NVM Express Base Specification and the Tunneled Status and Tunneled NVMe Management Response fields shall be cleared to 0h.

...

Figure 51: NVMe-MI Send Command Servicing Model



...

4.3.2.1 NVMe-MI Receive Command **Request Message** to NVMe Admin Command **SQE** Mapping

In order to tunnel an NVMe-MI Command in-band via NVMe-MI Receive, an NVMe-MI Request Message is mapped onto an NVMe Submission Queue Entry (SQE) as shown pictorially in Figure 52 and in table form in Figure 53. An NVMe-MI Response Message is mapped on to an NVMe Completion Queue Entry (CQE) as shown pictorially in Figure 52 and in table form in Figure 54. Refer to the NVM Express Base Specification for details on an NVMe Submission Queue Entry and NVMe Completion Queue Entry.

...

6 NVM Express Admin Command Set

...

Figure 116: List of NVMe Admin Commands Supported using the Out-of-Band Mechanism

Command	Opcode	NVMe Storage Device O/M/P ¹	NVMe Enclosure O/M/P ¹	Reference Specification
...				
NVMe-MI Receive Send	1Dh	P	P	NVM Express Base Specification
NVMe-MI Send Receive	1Eh	P	P	NVM Express Base Specification
...				

...

Appendix A Technical Note: NVM Express Basic Management Command

...

Example 1: SMBus block read of the drive's status (status flags, SMART warnings, temperature):

Start	Addr	W	Cmd Code	Ack	Restart	Addr	R	Length	Ack	Status Flags	Ack	SMART Warnings	Ack	Temp	Ack	Drive Life Used	Ack	Warning Temp	Ack	Power State	Ack	PEC	NACK	Stop
	D4h		00h			D5h		06h		BFh		FFh		1Eh		01h		3Ch		08h		10h 2Dh		

...

Example 4: I2C read of status and vendor content, I2C allows reading across SMBus block boundaries:

Start	Addr	W	Cmd Code	Ack	Restart	Addr	R	Length	Ack	Status Flags	Ack	SMART Warnings	Ack	Temp	Ack	Drive Life Used	Ack	Warning Temp	Ack	Power State	Ack	PEC	Ack	Stop
	D4h		00h			D5h		06h		BFh		FFh		1Eh		01h		3Ch		08h		10h 2Dh		
	VID		VID	Ack		Serial # 'A'	Ack	Serial # 'Z'	Ack	Serial # '1'	Ack	Serial # '2'	Ack	Serial # '3'	Ack	Serial # '4'	Ack	Serial # '5'	Ack	Serial # '6'	Ack	Serial # ''	Ack	
	12h		34h			41h		5Ah		31h		32h		33h		34h		35h		36h		20h		
	Serial # ''		Serial # ''	Ack		Serial # ''	Ack	Serial # ''	Ack	Serial # ''	Ack	Serial # ''	Ack	Serial # ''	Ack	Serial # ''	Ack	Serial # ''	Ack	Serial # ''	Ack	Serial # ''	Ack	
	20h		20h			20h		20h		20h		20h		20h		20h		20h		20h		20h		
																						B0h		

...

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2023 NVM Express, Inc.

Figure 177: Subsystem Management Data Structure

Command Code	Offset (byte)	Description						
0	...							
	06	Current Power (Optional): This field reports the current NVM Subsystem power consumption. If both bit mapped fields are cleared to 0h, then this field is not reported.						
		<table><tr><th>Bit</th><th>Definition</th></tr><tr><td>...</td><td></td></tr><tr><td>6:0</td><td>NVM Subsystem Power (NVMSI): This field reports the ceiling function of the power consumed by the NVM Subsystem in watts. If the power consumed by the NVM Subsystem in watts is greater than or equal to 127 W, then 127 W is reported in this field. Power reported by the NVM Subsystem is determined in the following manner. If the NVMSI bit is set to '1', then the value returned in this field is:<ul style="list-style-type: none">equal to the value reported in the Idle Power (IDLP) field in the Power State Descriptor Data Structure for the corresponding NVMe power state if the IDLP field is set to a non-zero value; orequal to the value reported in the Maximum Power (MP) field in the Power State Descriptor data structure for the corresponding NVMe power state, if the IDLP field is cleared to 0h.If the NVMSI bit is cleared to '0', then the value returned in this field is:<ul style="list-style-type: none">equal to the power value reported by in the Active Power Workload (ACTPW) field in the Power State Descriptor Structure for the corresponding NVMe power state if the ACTPW field is set to a non-zero value; orequal to the value reported in the Maximum Power (MP) field in the Power State Descriptor data structure for the corresponding NVMe power state, if the APW field is cleared to 0h.</td></tr></table>	Bit	Definition	...		6:0	NVM Subsystem Power (NVMSI): This field reports the ceiling function of the power consumed by the NVM Subsystem in watts. If the power consumed by the NVM Subsystem in watts is greater than or equal to 127 W, then 127 W is reported in this field. Power reported by the NVM Subsystem is determined in the following manner. If the NVMSI bit is set to '1', then the value returned in this field is: <ul style="list-style-type: none">equal to the value reported in the Idle Power (IDLP) field in the Power State Descriptor Data Structure for the corresponding NVMe power state if the IDLP field is set to a non-zero value; orequal to the value reported in the Maximum Power (MP) field in the Power State Descriptor data structure for the corresponding NVMe power state, if the IDLP field is cleared to 0h. If the NVMSI bit is cleared to '0', then the value returned in this field is: <ul style="list-style-type: none">equal to the power value reported by in the Active Power Workload (ACTPW) field in the Power State Descriptor Structure for the corresponding NVMe power state if the ACTPW field is set to a non-zero value; orequal to the value reported in the Maximum Power (MP) field in the Power State Descriptor data structure for the corresponding NVMe power state, if the APW field is cleared to 0h.
		Bit	Definition					
		...						
6:0	NVM Subsystem Power (NVMSI): This field reports the ceiling function of the power consumed by the NVM Subsystem in watts. If the power consumed by the NVM Subsystem in watts is greater than or equal to 127 W, then 127 W is reported in this field. Power reported by the NVM Subsystem is determined in the following manner. If the NVMSI bit is set to '1', then the value returned in this field is: <ul style="list-style-type: none">equal to the value reported in the Idle Power (IDLP) field in the Power State Descriptor Data Structure for the corresponding NVMe power state if the IDLP field is set to a non-zero value; orequal to the value reported in the Maximum Power (MP) field in the Power State Descriptor data structure for the corresponding NVMe power state, if the IDLP field is cleared to 0h. If the NVMSI bit is cleared to '0', then the value returned in this field is: <ul style="list-style-type: none">equal to the power value reported by in the Active Power Workload (ACTPW) field in the Power State Descriptor Structure for the corresponding NVMe power state if the ACTPW field is set to a non-zero value; orequal to the value reported in the Maximum Power (MP) field in the Power State Descriptor data structure for the corresponding NVMe power state, if the APW field is cleared to 0h.							
...								
...								

...