



LEGAL NOTICE:

© Copyright 2008 to 2024 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 2024 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 © 2008-2024 NVM Express, Inc.

NVM Express® Technical Errata

Errata ID	120
Revision Date	07/05/2024
Affected Spec Ver.	NVM Express® Base Specification Revision 2.0d NVM Express® NVM Command Set Specification Revision 1.0d NVM Express® Key Value Command Set Specification Revision 1.0d NVM Express® Zoned Namespace Command Set Specification Revision 1.1d NVM Express® PCIe Transport Specification Revision 1.0d NVM Express® RDMA Transport Specification Revision 1.0c NVM Express® TCP Transport Specification Revision 1.0d NVM Express® Management Interface Specification Revision 1.2d NVM Express Boot Specification 1.0
Corrected Spec Ver.	

Errata Author(s)

Name	Company
Mike Allison, Judy Brock, Bill Martin	Samsung
David Black, Austin Bolen, Doug Farley	Dell
Paul Suhler, Fred Knight	Kioxia
Andres Baez, Myron Loewen	Solidigm
Phil Cayton	Intel

Errata Overview

This ECN updates and clarifies various text within the NVM Express Base Specification Revision 2.0d, NVM Express NVM Command Set Specification Revision 1.0d, NVM Express Key Value NVM Command Set Specification Revision 1.0d, NVM Express Zoned Namespace Command Set Specification Revision 1.1d, NVM Express PCIe Transport Specification Revision 1.0d, NVM Express® RDMA Transport Specification Revision 1.0c, NVM Express® TCP Transport Specification Revision 1.0d, NVM Express Boot Specification 1.0, and the NVM Express Management Interface Specification 1.2d.

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Revision History

Revision Date	Change Description
10/11/2023	Initial creation with the solutions for: <ul style="list-style-type: none"> Bug 200 UDID Clarifications Bug 215 Secondary Controller State to Same State Transition
10/25/2023	Incorporated the solution for: <ul style="list-style-type: none"> Bug 90 NSZE description update
11/01/2023	Incorporated the solution for: <ul style="list-style-type: none"> Bug 202 Clarifications for the Read NVMe-MI Data Structure
11/29/2023	Incorporated the solution for: <ul style="list-style-type: none"> Bug 157 PCIe-specific resets Bug 95 Repeated AENs
12/13/2023	Incorporated the solution for: <ul style="list-style-type: none"> Bug 58 Definition of Product Serial Number has overlapping recommendations
12/20/2023	Incorporated the solution for: <ul style="list-style-type: none"> Bug 265 Relaxed Ordering Bit Needs PCIe Base Reference Updated to 2024.
1/2/2024	Removed unaltered sections.
1/10/2024	Incorporated the solution for: <ul style="list-style-type: none"> Bug 277 Remove the word "would" Bug 280 No intended ordering
1/12/2023	Incorporated the solution for: <ul style="list-style-type: none"> Bug 97 Controller behavior on reset for Feature values (with non-controller scope) in memory-based, multi-controller subsystems with shared namespaces Bug 238 Add paragraphs to increase clarity Bug 240 Create I/O Submission/completion queue commands should not be described for Fabrics Bug 244 Controller property language should be harmonized between PCIe and Fabrics Bug 245 Queue Size is PCIe specific (need to add Fabrics description) Bug 249 Controller Model is mostly Fabric specific Bug 253 Definition says Fabrics is a subset of the memory-based transport model
1/12/2024	Updated to reflect NVM Express Base Specification 2.0d, NVM Express NVM Command Set Specification 1.0d, and NVM Express Management Interface Specification 1.2d.
1/17/2024	Incorporated the solution for: <ul style="list-style-type: none"> Bug 281 Asynchronous Event Information field does not include Immediate events Bug 108 Boot Spec HFI Needs clarity on handling Bug 197 - Change "logical block" to "user data" Bug 257 PRP1 Field Definition is Ambiguous
1/23/2024	Incorporated the solution for: <ul style="list-style-type: none"> Bug 200 UDID Clarifications due to additional changes from previous inclusion Bug 267 Unclear which Controller List should be returned in Read NVMe-MI Data Structure Bug 266 Contradictory text regarding controller behavior depending on SGL Data Block Length/Address content when a controller requires the fields to be dword aligned Bug 164 Admin Command or Admin command Bug 122 Usages of Non-volatile memory should be non-volatile storage
3/28/2024	Editorial changes due to review in the Technical WG.
4/11/2024	Editorial updates based on review feedback.
4/18/2024	Editorial changes during review and readiness for 30-day member review.
4/19/2024	Removed section 3 from the NVMe-MI specification since no changes existed.
6/11/2024	Added < Change "non-volatile storage" to "non-volatile storage medium"> to the ZNS specification.
07/05/2024	Integrated

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Description of Changes

NVM Express Base Specification 2.0d:

Backward Incompatible Changes:

- The following may be considered an incompatible change:
 - A Virtualization Management command to place the secondary controller in a state (i.e., Online or Offline) when that secondary controller is in that state is not considered an error and that command shall complete successfully.
 - The effects of resets to different scoped Features has been updated to reflect the differences between resets of:
 - an NVM subsystems with a single controller;
 - an NVM subsystems with multiple controllers;
 - a domain that does reset the entire NVM subsystem when reset; and
 - a domain that does not reset the entire NVM subsystem when reset.
 - Corrects contradictory text on controller behavior for SGL Data Blocks by removing the shall requirement that duplicated the text in the table describing the SGL Data Block Address and Length fields.

Editorial Changes:

- Replaced the list of possible resets with Controller Level Reset in description of Firmware Update Process.
- Added references indicate that Conventional Reset and Function Level Reset are PCIe-specific.
- Clarified that unmasking an asynchronous event may cause that event to re-occur.
- Exempted immediate events from text on masking and clearing asynchronous events.
- Added a reference to the PCIe Base Specification to a use of the Relaxed Ordering bit, which is defined in that specification.
- Removed the use of the word “would” in the description of the firmware update process.
- Clarified Controller Architecture Model for PCIe vs. Fabrics.
- Clarified text to various sections of the specification regarding the behavior of feature saved and default values in different Subsystem configurations across resets.
- Clarified that the Asynchronous Event Information field does include the Immediate Events.
- Generalized “logical block” terminology to “user data”.
- Clarified description of the PRP1 field in the Common Command Format.
- Updates to use consistent case for Admin command, I/O command, and Fabrics command.
- Clarified the consistent usage of “NVM subsystem” and “non-volatile storage medium”.

NVM Express NVM Command Set Specification 1.0d:

Editorial Changes:

- Added a reference to the NSZE field description in Identify Namespace for consistency with NCAP and NUSE fields.
- Clarified the host access to the LBA Status Information log page using the Retain Asynchronous Event bit
- Updates to use consistent case for Admin command.
- Clarified the consistent usage of “NVM subsystem” and “non-volatile storage medium”.

NVM Express Key Value Command Set Specification 1.0d:

Editorial Changes:

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

- Updates to use consistent case for Admin command.
- Clarified the consistent usage of “NVM subsystem” and “non-volatile storage medium”.

NVM Express Zoned Namespace Command Set Specification 1.1d:

Editorial Changes:

- Updates to use consistent case for Admin command.
- Clarified the consistent usage of “NVM subsystem” and “non-volatile storage medium”.

NVM Express PCIe Transport Specification 1.0d:

Editorial Changes:

- Clarified the consistent usage of “NVM subsystem” and “non-volatile storage medium”.

NVM Express RDMA Transport Specification 1.0c:

Editorial Changes:

- Clarified the consistent usage of “NVM subsystem” and “non-volatile storage medium”.

NVM Express TCP Transport Specification 1.0d:

Editorial Changes:

- Clarified the consistent usage of “NVM subsystem” and “non-volatile storage medium”.

NVM Express Boot Specification 1.0:

Editorial Changes:

- Clarified the routing for HFI Global and Local Routes.

NVM Express Management Interface Specification 1.2d:

Editorial Changes:

- Clarified the description of the Unique Device Identifier (UDID)
- Clarified that the IOCSI field is only applicable if the DTYP field value is 04h (Optionally Supported Command List) or 05h (Management Endpoint Buffer Command Support List) and the NMIMT field value is 02h (NVMe Admin Command) and should be ignored for all other scenarios.
- In various fields in the Product Info Area, eliminate padding of both NULL characters or spaces (ASCII character 20h), if any – thereby also eliminating overlapping definitions, if any, with corresponding fields in Identify Controller data structure.
- Clarified the Read NVMe-MI Data Structure command.

Note:

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 destination of moved text without changes. **Purple stricken** text indicates source of moved text without changes. **GREEN** text indicates editor notes.

Description of NVM Express Base Specification 2.0d changes

<Change “non-volatile memory storage medium” to “non-volatile storage medium”>

<Change “non-volatile media” to “non-volatile storage medium”>

Modify section 1 as shown below:

1 Introduction

...

The NVM Express® (NVMe®) interface allows host software to communicate with a non-volatile memory subsystem (NVM subsystem). This interface is optimized for all storage solutions, attached using a variety of transports including PCI Express®, Ethernet, InfiniBand™, and Fibre Channel. The mapping of extensions defined in this document to a specific NVMe Transport are defined in an NVMe Transport binding specification. The NVMe Transport binding specification for Fibre Channel is defined in INCITS 556 Fibre Channel – Non-Volatile Memory Express - 2 (FC-NVMe-2).

...

1.1.1 NVM Express® Specification Family

...

The NVM Express Base (NVM Express Base) Specification (i.e., this specification) defines a protocol for host software to communicate with a non-volatile memory-an NVM subsystems over a variety of memory-based transports and message-based transports.

...

1.3 Outside of Scope

The property interface and command set are specified apart from any usage model for the NVM, but rather only specifies the communication interface to the NVM subsystem. Thus, this specification does not specify whether the non-volatile memory NVM subsystem is used as a solid state drive, a main memory, a cache memory, a backup memory, a redundant memory, etc. Specific usage models are outside the scope, optional, and not licensed.

...

This interface is specified above any non-volatile media memory management, like wear leveling. Erases and other management tasks for NVM technologies like NAND are abstracted.

...

1.5 Definitions

...

1.5.1 NEW memory-based controller

A controller that supports a memory-based transport model (e.g., a PCIe implementation).

...

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

1.5.1 message-based controller

A controller that supports a message-based transport model (e.g., a Fabrics implementation).

...

1.5.43 NVMe over Fabrics

An implementation of the NVMe Express interface that complies ~~to with~~ either the message-only transport model or the message/memory-based transport model ~~implementation of the memory-based transport model definition~~ (refer to Figure 4 and section 2.2).

...

Modify section 2 as shown below:

2 Theory of Operation

...

There are two defined ~~constructs~~ models for communication between the host and the NVMe subsystem, a memory-based transport model and a message-based transport model. All NVMe subsystems require the underlying NVMe Transport to provide reliable NVMe command and data delivery. An NVMe Transport is an abstract protocol layer independent of any physical interconnect properties. A taxonomy of NVMe Transports along with examples is shown in Figure 4. An NVMe Transport may expose a memory-based transport model or a message-based transport model. The message-based transport model has two subtypes: the message-only transport model and the message/memory transport model.

<Editor's Note: new paragraph>

A memory-based transport model is one in which commands, responses, and data are transferred between a host and an NVMe subsystem by performing explicit memory read and write operations.

<Editor's Note: new paragraph>

A message-based transport model is one in which messages containing command capsules and response capsules are sent between a host and an NVMe subsystem. The two subtypes of message-based transport models are differentiated by how data is sent between a host and an NVMe subsystem. In the message-only transport model data is only sent between a host and an NVMe subsystem using capsules or messages. The message/memory-based transport model uses a combination of messages and explicit memory read and write operations to transfer command capsules, response capsules and data between a host and an NVMe subsystem. Data may optionally be included in command capsules and response capsules. Both the message-only transport model and the message/memory-based transport model are referenced as message-based transport models throughout this specification when the description is applicable to both subtypes.

...

There are three types of commands that are defined in NVMe Express: Admin cCommands, I/O cCommands, and Fabrics cCommands. Figure 5 shows these different command types.

2.1 Memory-Based Transport Model (PCIe)

...

2.2 Message-Based Transport Model (Fabrics)

...

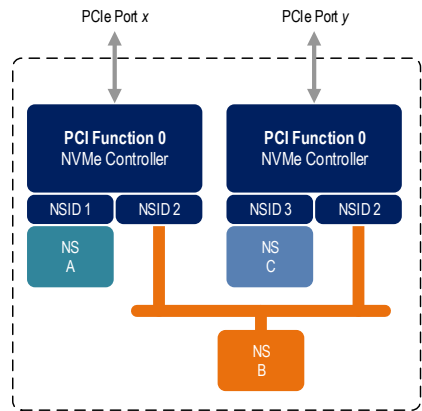
Technical input submitted to the NVMe Express® Workgroup is subject to the terms of the NVMe Express® Participant's agreement. Copyright © 2008-2024 NVMe Express, Inc.

2.4 Extended Capabilities Theory

2.4.1 Multi-Path I/O and Namespace Sharing

Figure 18 illustrates an NVM subsystem with two PCI Express ports, each with an associated controller implementing NVMe over PCIe. Both controllers map to PCI Function 0 of the corresponding port. Each PCI Express port in this example is completely independent and has its own PCI Express Fundamental Reset and reference clock input. A reset of a port only affects the controller associated with that port and has no impact on the other controller, shared namespace, or operations performed by the other controller on the shared namespace. Refer to section 4.2 for Feature behavior on reset. The functional behavior of this example is otherwise the same as that illustrated in Figure 17.

Figure 18: NVM Subsystem with Two Controllers and Two Ports



Modify section 3 as shown below:

3 Admin Command Set

3.1 NVM Controller Architecture

A controller is the interface between a host and an NVM subsystem. This specification defines two controller models, the static controller model and the dynamic controller model. All controllers in an NVM subsystem shall support the same controller model.

In an NVM subsystem that supports the static controller model, state (e.g., controller ID, saved Feature settings) is preserved:

- across a Controller Level Reset for memory-based controllers and message-based controllers; and
- from prior associations for message-based controllers.

In an NVM subsystem that supports the dynamic controller model, the NVM subsystem allocates controllers on demand with no state preserved from prior associations.

3.1.1 Controller Model Memory-Based Controller Architecture (PCIe)

Memory-based controllers shall support only the static controller model.

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

3.1. NEW Message-Based Controller Architecture (Fabrics)

~~This specification defines two controller models. An NVM subsystem may support a static or dynamic controller model. All controllers in the NVM subsystem shall follow the same controller model.~~

Message-based controllers, other than Discovery controllers, may support either the dynamic controller model or the static controller model. A Discovery controller shall support only the dynamic controller model.

In an NVM subsystem that supports the static controller model, ~~each~~ controllers that ~~may be~~ is allocated to a particular host may have different state at the time the association is established (e.g., each controller may have state that is preserved from a prior association). The controllers within such an NVM subsystem are distinguished by their controller identifier. The host may request a particular controller based on the Controller ID (refer to the CNTLID field in Figure 382). ~~All memory-based transport model controllers shall support the static controller model.~~

In an NVM subsystem that supports the dynamic controller model, ~~the~~ each controller is allocated by the NVM subsystem on demand with no state (e.g., Controller ID, Feature settings) preserved from prior associations. In this model, all controllers allocated to a specific host have the same state at the time the association is established, including Feature settings. The initial set of attached namespaces should be the same for all controllers that are allocated to a specific host and accessed via the same NVM subsystem port. The initial set of attached namespaces may differ among controllers that are each accessed via a different NVM subsystem port. Changes to a dynamic controller (e.g., attached namespaces, Feature settings) after the association is established do not impact other dynamic controllers in that NVM subsystem.

~~Controllers using the message-based transport model in an NVM subsystem may use a dynamic or static controller model. A Discovery controller shall support the dynamic controller model.~~

An association is established between a host and a controller when the host connects to a controller's Admin Queue using the Fabrics Connect command (refer to section 6.3). Within the Connect command, the host specifies the Host NQN, NVM Subsystem NQN, Host Identifier, and may request a specific Controller ID (e.g., the static controller model is being used) or may request a connection to any available controller (e.g., the dynamic controller model is being used). A controller has only one association at a time.

~~In a dynamic controller model, the controller is allocated by the NVM subsystem on demand with no state (e.g., Feature settings) preserved from prior associations. In a static controller model, the host may request a particular controller based on the Controller ID where state (e.g., Feature settings) is preserved from prior associations.~~

...

When using the static controller model with a Fabric connected controller, the state that persists across associations is any state that persists across a Controller Level Reset. Additionally, different controllers may present different Feature settings or namespace attachments to the same host. The NVM subsystem may allocate particular controllers to specific hosts.

...

3.1.2 Controller Types

...

3.1.2.2 Administrative Controller

...

3.1.2.2.1 Command Support

...

Figure 28: Administrative Controller – Admin Command Support

Command	Command Support Requirements ¹	Reference
...		
I/O Command Set specific Admin cCommand	Refer to the applicable I/O Command Set specification	Refer to the applicable I/O Command Set specification
...		

...

3.1.2.3 Discovery Controller

...

3.1.2.3.2 Command Support

...

Figure 32: Discovery Controller – Admin Command Support

Command	Command Support Requirements ¹	Reference
...		
I/O Command Set Specific Admin cCommands	P	
...		

...

3.1.3 Controller Properties

A property is a dword, or qword attribute of a controller. The attribute may have read, write, or read/write access. The host shall access a property using the width specified for that property with an offset that is at the beginning of the property unless otherwise noted in a transport specific specification. All reserved properties and all reserved bits within properties are read-only and return 0h when read.

< Editors Note:add paragraph>

For message-based controllers, Pproperties may be read with the Property Get command and may be written with the Property Set command with controllers using the message-based transport model.

< Editors Note:add paragraph>

For controllers using the memory-based transport model controllers, refer to the applicable NVMe Transport binding specification for access methods and rules (e.g., NVMe PCIe Transport Specification).

...

Figure 35: Property Definition

Offset (OFST)	Size (in bytes)	I/O Controller ¹	Admin. Controller ¹	Discovery Controller ¹	Name
0h	8	M	M	M	CAP: Controller Capabilities
...					
1300h		O	O	O	Vendor Specific
Notes:					
1. O/M/P definition: O = Optional, M = Mandatory, R = Reserved, T = Transport Specific					
2. Mandatory for memory-based controllers transport implementations. Reserved for For message-based transport implementations controllers this property is reserved.					

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Figure 35: Property Definition

Offset (OFST)	Size (in bytes)	I/O Controller ¹	Admin. Controller ¹	Discovery Controller ¹	Name
3. Optional for memory-based controllers transport implementations. Reserved for For message-based transport implementations controllers this property is reserved.					

3.1.3.1 Offset 0h: CAP – Controller Capabilities

Figure 36: Offset 0h: CAP – Controller Capabilities

Bits	Type	Reset	Description
...			
16	RO	Impl Spec	<p>Contiguous Queues Required (CQR): This bit is set to '1' if the controller requires that I/O Submission Queues and I/O Completion Queues are required to be physically contiguous. This bit is cleared to '0' if the controller supports I/O Submission Queues and I/O Completion Queues that are not physically contiguous. If this bit is set to '1', then the Physically Contiguous bit (CDW11.PC) in the Create I/O Submission Queue and Create I/O Completion Queue commands shall be set to '1'.</p> <p>For I/O controllers and Discovery controller using a message-based transport, this property shall be set to a value of 1h.</p>
...			

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 Controller Level Resets other than Controller Lever Resets caused by:

- a Controller Reset; and
- a Function Level Reset (refer to the NVM Express NVMe over PCIe Transport Specification).

3.3 NVM Queue Models

3.3.1 Memory-based Transport Queue Model (PCIe)

3.3.2 Message-based Transport Queue Model (Fabrics)

3.3.2.2 Queue Creation

Controllers using the mMessage-based transport queue model controllers use the Connect command (refer to section 6.3) to create controller Admin Queues or I/O Queues. The creation of an Admin Queue establishes an association between a host and the corresponding controller. The message-based transport

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

queue model does not support the Admin Submission Queue Base Address (ASQ), Admin Completion Queue Base Address (ACQ), and Admin Queue Attributes (AQA) properties as all information necessary to establish an Admin Queue is contained in the Connect command. The message-based transport queue model does not support the Admin commands associated with I/O Queue creation and deletion (Create I/O Completion Queue, Create I/O Submission Queue, Delete I/O Completion Queue, Delete I/O Submission Queue).

...

3.3.3 Queueing Data Structures

...

3.3.3.1 Submission Queue Entry

...

Figure 88: Common Command Format

Bytes	Description
...	...
39:24	<div><div><div><div><div>...</div></div><div><div>PRP Entry 1 (PRP1): This field contains:<ul style="list-style-type: none">the first PRP entry for the command; ora PRP List pointer,depending on the command (e.g., the Create I/O Completion Queue command; refer to Figure 156).</div></div></div></div></div>
...	...

...

In addition to the fields commonly defined for the Common Command Format, ~~Admin and NVM~~ Vendor Specific commands may support the Number of Dwords in Data Transfer and Number of Dwords in Metadata Transfer fields. If supported, the command format for the ~~Admin~~ Vendor Specific ~~C~~ommands and ~~NVM~~ Vendor Specific Commands are defined in Figure 89. For more details, refer to section 8.23.

Figure 89: Common Command Format – ~~Admin and NVM~~ Vendor Specific Commands (Optional)

Bytes	Description
03:00	Command Dword 0 (CDW0): This field is common to all commands and is defined in Figure 87.
...	...

...

3.3.3.2 Common Completion Queue Entry

...

3.3.3.2.1 Status Field Definition

...

3.3.3.2.1.4 Path Related Status Definition

...

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Figure 99: Status Code – Path Related Status Values

Value	Description
...	
Host detected Pathing errors	
...	
71h	Command Aborted By Host: The command was aborted as a result of host action (e.g., the host disconnected the Fabric connection).
...	

...

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

For Message-based controllers, the maximum size for the Admin Submission Queue is limited by the value indicated in the ASQSZ field in the Discovery Log Page Entry data structure (refer to Figure 265).

...

3.4 Command Architecture Submission and Completion Mechanism

...

3.4.4 Command Arbitration

...

A command is being processed when the controller and/or namespace state is being accessed or modified by the command such as: {e.g.,

- a Feature setting is being accessed or;
- a Feature setting is being modified;
- or a logical block user data (e.g., a logical block as defined by the NVM Command Set Specification) is being accessed; or
- user data is being modified}.

...

3.5 Controller Initialization

...

3.5.1 Memory-based Transport Controller Initialization (PCIe)

...

3.5.2 Message-based Transport Controller Initialization (Fabrics)

...

3.6 Shutdown Processing

...

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

3.6.1 Memory-based ~~Transport~~ Controller Shutdown (PCIe)

...

3.6.2 Message-based ~~Transport~~ Controller Shutdown (Fabrics)

...

3.7 Resets

...

<This is section 3.7.2 without the TP 4100 changes for integration into a 2.0x revision of the NVM Express Base Specification>

3.7.2 Controller Level Reset

...

The following methods initiate a Controller Level Reset:

- NVM Subsystem Reset;
- Controller Reset (i.e., host clears CC.EN from '1' to '0'); and
- Transport specific reset types (refer to the applicable NVMe Transport binding specification), if any.

A Controller Level Reset consists of the following actions:

- 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
- All controller properties defined in section 3.1.3 and internal controller state are reset, with the following exceptions:
 - for ~~controllers using a~~ memory-based ~~transport~~ controllers:
 - the Admin Queue properties (AQA, ASQ, or ACQ) are not reset as part of a Controller Reset;
 - the Controller Memory Buffer Memory Space Control property (CMBMSC) is reset as part of neither a Controller Reset nor a Function Level Reset (refer to the NVM Express NVMe over PCIe Transport Specification); and
 - the Persistent Memory Region Memory Space Control Upper property (PMRMSCU) and the Persistent Memory Region Memory Space Control Lower property (PMRMSCL) are not reset as part of a Controller Reset;
 - and
 - for ~~controllers using a~~ message-based ~~transport~~ controllers:
 - there are no exceptions.

...

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:

...

3. The host performs an action that results in a ~~reset~~ Controller Level Reset (refer to section 3.7.2) on that controller to cause 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.~~

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

- a. In some cases a Conventional Reset (refer to the NVM Express NVMe over PCIe Transport Specification) 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);

...

The process for a firmware update to be activated on a domain 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 aborts the command with an appropriate status code (e.g., Invalid Firmware Image);
 - b. If the firmware activation was not successful because a Controller Level Reset is required to activate this firmware image, then the controller aborts the command with a status code of Firmware Activation Requires Controller Level Reset and the firmware image is applied at the next Controller Level Reset;
 - c. If the firmware activation was not successful because an NVM Subsystem Reset is required to activate this firmware image, then the controller aborts the command with a status code of Firmware Activation Requires NVM Subsystem Reset and the firmware 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 image, then the controller aborts the command with a status code of Firmware Activation Requires Conventional Reset and the firmware image is applied at the next Conventional Reset; and
 - e. If the firmware activation was not successful because the firmware activation time requires more time than the time reported by ~~would exceed~~ the MTFA field ~~value reported~~ in the Identify Controller data structure, then the controller aborts the command with a status code of Firmware Activation Requires Maximum Time Violation. In this case, the firmware image was committed to the specified firmware slot. To activate that firmware image, the host may issue a Firmware Commit command that specifies:
 - i. a Commit Action set to 010b (i.e., activate using a Controller Level Reset); and
 - ii. the same firmware slot.

...

Modify section 4 as shown below:

4 Data Structures

...

4.1 Data Layout

...

4.1.2 Scatter Gather List (SGL)

...

The controller shall abort a command if:

- an SGL segment contains an SGL Segment descriptor or an SGL Last Segment descriptor in other than the last descriptor in the segment;
- a last SGL segment contains an SGL Segment descriptor, or an SGL Last Segment descriptor; ~~or~~
- an SGL descriptor has an unsupported format; ~~or~~.

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

The controller should abort a command if:

- an SGL Data Block descriptor contains Address or Length fields with either of the two least significant bits set to 1b and the controller supports only dword alignment and granularity as indicated in the SGL Support field of the Identify Controller data structure. Refer to Figure 276.

...

4.2 Feature Values

<Text in orange is from TP4074a>

The Get Features command (refer to section 5.15) and Set Features command (refer to section 5.27) 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 276, then for each Feature, there are three settings: default, saved, and current. If bit 4 is cleared to '0' in the Optional NVM Command Support field of the Identify Controller data structure, then the controller only supports a current and default value for each Feature. In this case, the current value may be persistent across power cycles and resets based on the information specified in Figure 317.

If bit 4 is set to '1' in the ONCS field, then each Feature has supported capabilities (refer to Figure 196), which are discovered using the Supported Capabilities value in the Select field in Get Features (refer to Figure 193).

The default value for each Feature is vendor specific and set by the manufacturer unless otherwise specified. The default value is not changeable.

~~A Feature may be saveable. The saved value is the value that the Feature has after a Controller Level Reset. If a Feature is not saveable or does not have a saved value, then:~~

- ~~a) the default value is used after a Controller Level Reset; and~~
- ~~b) a Get Features command to read the saved value returns the default value.~~

The current value for a Feature is the value in active use by the controller for that Feature.

A Set Features command uses the value specified by the command to set:

- a) the current value for that Feature; or
- b) the current value for that Feature and the saved value for that Feature, if that Feature is saveable.

A Feature may be saveable. If a Feature is saveable (i.e., the controller supports the Save field in the Set Features command and the Select field in the Get Features command), then any Feature Identifier that is namespace specific may be saved on a per namespace basis.

If a Feature is not saveable, or does not have a saved value, then a Get Features command to read the saved value returns the default value.

If a Feature is not saveable and is persistent as specified in Figure 317, then the current value of that Feature shall be persistent across power cycles and resets.

The current value for a Feature, as a result of a Controller Level Reset, is indicated in Figure TBDCHANGE for an NVM subsystem that supports only a single controller (i.e., bit 1 of the CMIC field (refer to Figure 275) is cleared to '0').

The current value for a Feature, as a result of an NVM Subsystem Reset, is indicated in Figure TBDCHANGE for an NVM subsystem:

- that is able to support two or more controllers and does not support multiple domains; or
- supports multiple domains where the NVM Subsystem Reset results in a reset of the entire NVM subsystem.

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Figure **TBDCHANGE**: Current Value after Reset with Scope of Entire NVM Subsystem

Feature Capability	Controller scope or Namespace per controller scope	NVM subsystem scope	Endurance Group scope	NVM Set scope	Namespace scope
Saveable	Set to: <ul style="list-style-type: none"> the saved value if a saved value is set; or the default value if a saved value is not set, unless otherwise specified. 				
Non-saveable and non-persistent	Set to the default value, unless otherwise specified.				

The current value for a Feature, as a result of a Controller Level Reset, is indicated in Figure **TBDNOTCHANGE** for an NVM subsystem that is able to support two or more controllers (i.e., bit 1 of the CMIC field is set to '1').

The current value for a Feature, as a result of an NVM Subsystem Reset, is indicated in Figure **TBDNOTCHANGE** for an NVM subsystem that supports:

- multiple domains where an NVM Subsystem Reset does not reset the entire NVM subsystem as defined in section 3.7.1.2.

Figure **TBDNOTCHANGE**: Current Value after Reset with Scope of Subset of the NVM Subsystem

Feature Capability	Controller scope or Namespace per controller scope	NVM subsystem scope	Endurance Group scope	NVM Set scope	Namespace scope
Saveable	Set to: <ul style="list-style-type: none"> the saved value if a saved value is set; or the default value if a saved value is not set, unless otherwise specified. 	Unchanged, unless otherwise specified.			
Non-saveable and non-persistent	Set to the default value, unless otherwise specified.				

Concurrent access to Feature values by multiple hosts, for features with a scope other than controller scope, requires some form of coordination between hosts. The procedure used to coordinate these hosts is outside the scope of this specification.

Feature settings ~~may~~ apply ~~to~~: based on the feature scope (e.g., a feature is namespace specific if that feature has a namespace scope) as defined in Figure 317.:

- ~~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 scope:

- a) if the NSID field is cleared to 0h or set to 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 NSID field is set to a valid namespace identifier (refer to section 3.2.1.2), then:
 - the Set Features command shall abort with a status code of Feature Not Namespace Specific; and

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

- the Get Features command shall return the current setting of the requested feature value for the controller.

For feature values that apply to a the namespace scope:

- a) if the NSID field is set to an active namespace identifier (refer to section 3.2.1.4), 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 NSID field is set to FFFFFFFFh, then:
 - for the Set Features command, the controller shall:
 - if the MDS bit is set to '1' in the Identify Controller data structure, abort the command with Invalid Field in Command; or
 - if the MDS bit is cleared to '0' in the Identify Controller data structure, unless otherwise specified, set the specified feature value for all namespaces attached to the controller processing the command;
 - and
 - for the Get Features command, the controller shall, unless otherwise specified in section 5.27.1, abort the command with a status code of Invalid Namespace or Format;
 - and
- c) if the NSID field is set to any other value, then the Set Features command and the Get Features command shall be aborted by the controller as described in the rules for namespace identifier usage in Figure 88.

For feature values that apply to NVM subsystem scope, Domain scope, NVM Set scope, or Endurance Group scope:

- a) the NSID field should be cleared to 0h; and
- b) if the NSID field is set to a non-zero value, then the Set Features command and the Get Features command are aborted by the controller as described in Figure 88.

If the controller supports the Save field in the Set Features command and the Select field in the Get Features command, then any Feature Identifier that is namespace specific may be saved on a per namespace basis.

There are mandatory and optional Feature Identifiers defined in Figure 317. If a Get Features command or Set Features command is processed that specifies a Feature Identifier that is not supported, then the controller shall abort the command with a status code of Invalid Field in Command.

...

Modify section 5 as shown below:

5 Admin Command Set

...

Figure 141: Sanitize Operations and Format NVM Command – Admin Commands Allowed

Admin Command	Additional Restrictions for a Format NVM cCommand	Additional Restrictions for a Ssanitize Ooperations
...		

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Figure 141: Sanitize Operations and Format NVM Command – Admin Commands Allowed

Admin Command	Additional Restrictions for a Format NVM Command	Additional Restrictions for a Sanitize Operation
Opcode 7Fh	The Fabric's Commands allowed are listed below. Refer to section 6.	
	Fabric's Commands	Additional Restrictions for both a Format NVM Command and a Sanitize Operation
	Property 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, then, except for immediate events, subsequent events of that event type are automatically masked by the controller until the host clears that event. Unless otherwise stated (e.g., for immediate events), an event is cleared by reading the log page associated with that event (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.

Asynchronous events are reported due to a new entry being added to a log page (e.g., Error Information log page), or a status change update (e.g., status in the SMART / Health log page), or occurrence of a defined condition (e.g., Firmware Activation Starting, NVM Subsystem Normal Shutdown). 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 clearing that event to avoid repeated reporting of asynchronous events caused by clearing an asynchronous event in the absence of change to the status that caused that event to be reported.

5.2.1 Command Completion

Figure 145: Asynchronous Event Request – Completion Queue Entry Dword 0

Bits	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.
	For asynchronous events not associated with a log page (refer to section 5.2), this field should be ignored by the host.
15:08	Asynchronous Event Information: Refer to Figure 146, Figure 147, Figure 148, and Figure 149 for detailed information regarding the asynchronous event. This field indicates additional information about the asynchronous event for the event indicated in the Asynchronous Event Type field.
07:03	Reserved

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Figure 145: Asynchronous Event Request – Completion Queue Entry Dword 0

Bits	Description
02:00	Asynchronous Event Type: Indicates the event type of the asynchronous event. More specific information on the event is provided in the Asynchronous Event Information field.

5.12 Firmware Commit command

...

5.12.1 Command Completion

...

Figure 184: Firmware Commit – Command Specific Status Values

Value	Description
...	
0Bh	Firmware Activation Requires Conventional Reset: The firmware commit was successful, however, activation of the firmware image requires a Conventional Reset (refer to the NVM Express NVMe over PCIe Transport Specification). If an FLR a Function Level Reset (refer to the NVM Express NVMe over PCIe Transport Specification) or Controller Reset occurs prior to a Conventional Reset, the controller shall continue operation with the currently executing firmware image.
...	

...

5.14 Format NVM command

...

The Format NVM command may be aborted with a status code defined in this specification under circumstances defined by a security specification (e.g., invalid security state as specified in TCG Storage Interface Interactions specification). If there are I/O commands being processed for a namespace, then a Format NVM command that is submitted affecting that namespace may be aborted; if aborted, then a status code of Command Sequence Error should be returned. If a Format NVM command is in progress, then an I/O command that is submitted for any namespace affected by that Format NVM command may be aborted; if aborted, then a status code of Format in Progress should be returned. Refer to section 5 for further information about restrictions on Admin [c](#)Commands during Format NVM.

...

5.17 Identify command

...

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

5.17.2 Identify Data Structures

5.17.2.1 Identify Controller Data Structure (CNS 01h)

...

Figure 276: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O ¹	Admin ¹	Disc ¹	Description								
...												
99:96	M	M	O ⁴	Controller Attributes (CTRATT): This field indicates attributes of the controller.								
				<table><tr><th>Bits</th><th>Description</th></tr><tr><td>...</td><td></td></tr><tr><td>6</td><td>Traffic Based Keep Alive Support (TBKAS): If set to '1', then the controller uses Traffic Based Keep Alive (refer to section 3.9.4.1). If cleared to '0', then the controller does not use Traffic Based Keep Alive.</td></tr><tr><td>...</td><td></td></tr></table>	Bits	Description	...		6	Traffic Based Keep Alive Support (TBKAS): If set to '1', then the controller uses Traffic Based Keep Alive (refer to section 3.9.4.1). If cleared to '0', then the controller does not use Traffic Based Keep Alive.	...	
				Bits	Description							
...												
6	Traffic Based Keep Alive Support (TBKAS): If set to '1', then the controller uses Traffic Based Keep Alive (refer to section 3.9.4.1). If cleared to '0', then the controller does not use Traffic Based Keep Alive.											
...												
...												
264	M	M	R	Admin Vendor Specific Command Configuration (AVSCC): This field indicates the configuration settings for Admin-Vendor-Specific-vendor specific Admin command handling. Refer to section 8.23. Bits 7:1 are reserved. Bit 0 if set to '1' indicates that all Admin-Vendor-Specific-Commands vendor specific Admin commands use the format defined in Figure 89. If cleared to '0' indicates that the format of all Admin-Vendor-Specific Commands vendor specific Admin commands is are vendor specific.								
...												
530	M	M	R	I/O Command Set Vendor Specific Command Configuration (ICSVSCC): This field indicates the configuration settings for I/O Command-Set-Vendor-Specific vendor specific I/O command handling. Refer to section 8.23. Bits 7:1 are reserved. Bit 0 if set to '1' indicates that all NVM-Vendor-Specific-Commands vendor specific I/O commands use the format defined in Figure 89. If cleared to '0' indicates that the format of all NVM-Vendor-Specific Commands-vendor specific I/O commands is are vendor specific.								
...												
Fabric Specific												
1795:1792	M ²	R M ²	R	I/O Queue Command Capsule Supported Size (IOCCSZ): This field defines the maximum I/O command capsule size in 16 byte units. The minimum value that shall be indicated is 4 corresponding to 64 bytes.								
1799:1796	M ²	R M ²	R	I/O Queue Response Capsule Supported Size (IORCSZ): This field defines the maximum I/O response capsule size in 16 byte units. The minimum value that shall be indicated is 1 corresponding to 16 bytes.								

Commented [DA1]: Why is this included in the ECN? No changes

Figure 276: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O ¹	Admin ¹	Disc ¹	Description
1801:1800	M ²	R M ²	R	<p>In Capsule Data Offset (ICDOFF): This field defines the offset where data starts within a capsule. This value is applicable to I/O Queues only (the Admin Queue shall use a value of 0h).</p> <p>The value is specified in 16 byte units. The offset is from the end of the submission queue entry within the command capsule (starting at 64 bytes in the command capsule). The minimum value is 0 and the maximum value is FFFFh.</p>
...				
1805:1804	M ²	M ²	R	<p>Optional Fabrics Commands Support (OFCS): Indicate whether the controller supports optional F fabrics commands.</p> <p>Bits 15:1 are reserved.</p> <p>Bit 0 if cleared to '0' then the controller does not support the Disconnect command. Bit 0 if set to '1' then the controller supports the Disconnect command and deletion of individual I/O Queues.</p>
...				
<p>Notes:</p> <ol style="list-style-type: none"> 1. O/M/R definition: O = Optional, M = Mandatory, R = Reserved. 2. Mandatory for I/O message-based controllers using a message-based transport. Reserved for For memory-based controllers this field is reserved using a memory-based transport. 3. Mandatory for Discovery controllers that support explicit persistent connections. Reserved for For Discovery controllers that do not support explicit persistent connections this field is reserved. 4. For Discovery controllers, the TBKAS bit is optional for Discovery controllers, and all other bits are reserved for Discovery controllers. 				

...

5.24 Sanitize command

...

5.24.1 Command Completion

...

Figure 306: Sanitize – Command Specific Status Values

Value	Description
0Bh	<p>Firmware Activation Requires Conventional Reset: The sanitize operation could not be started because a firmware activation is pending and a Conventional Reset (refer to the NVM Express NVMe over PCIe Transport Specification) is required.</p>
...	

...

5.27 Set Features command

...

5.27.1 Feature Specific Information

Figure 317 defines the Features that may be configured with a Set Features command and retrieved with a Get Features command. Refer to section 3.1.2 for mandatory, optional, and prohibited features for the various controller types. 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. If a Feature is not persistent across power cycles and resets, then the current value of that Feature shall be set to the default

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

value of that Feature as part of a Controller Level Reset. For more information on Features, including default value definitions, saved value definitions, and current value definitions, refer to section 4.2.

5.28 Virtualization Management command

Figure 372: Virtualization Management – Command Dword 10

Bits	Description												
...													
03:00	Action (ACT): This field indicates the operation for the command to perform as described below.												
	<table><tr><th>Value</th><th>Description</th></tr><tr><td>...</td><td></td></tr><tr><td>7h</td><td>Secondary Controller Offline: Place the secondary controller in the Offline state and remove all Flexible Resources. If the Controller Identifier field does not correspond to a secondary controller associated with this primary controller, then a status code of Invalid Controller Identifier is returned. It is not an error to request a secondary controller be placed in the offline state if that secondary controller is already in the offline state.</td></tr><tr><td>8h</td><td>Secondary Controller Assign: Assign the number of controller resources specified in Number of Controller Resources to the secondary controller. If the Controller Identifier field does not correspond to a secondary controller associated with this primary controller, then an error of Invalid Controller Identifier is returned. If the secondary controller is not in the Offline state, then a status code of Invalid Secondary Controller State is returned.</td></tr><tr><td>9h</td><td>Secondary Controller Online: Place the secondary controller in the Online state. If the Controller Identifier field does not correspond to a secondary controller associated with this primary controller, then an error of Invalid Controller Identifier is returned. If the secondary controller is not configured appropriately (refer to section 8.26) or the primary controller is not enabled, then a status code of Invalid Secondary Controller State is returned. It is not an error to request a secondary controller be placed in the online state if that secondary controller is already in the online state.</td></tr><tr><td>...</td><td></td></tr></table>	Value	Description	...		7h	Secondary Controller Offline: Place the secondary controller in the Offline state and remove all Flexible Resources. If the Controller Identifier field does not correspond to a secondary controller associated with this primary controller, then a status code of Invalid Controller Identifier is returned. It is not an error to request a secondary controller be placed in the offline state if that secondary controller is already in the offline state.	8h	Secondary Controller Assign: Assign the number of controller resources specified in Number of Controller Resources to the secondary controller. If the Controller Identifier field does not correspond to a secondary controller associated with this primary controller, then an error of Invalid Controller Identifier is returned. If the secondary controller is not in the Offline state, then a status code of Invalid Secondary Controller State is returned.	9h	Secondary Controller Online: Place the secondary controller in the Online state. If the Controller Identifier field does not correspond to a secondary controller associated with this primary controller, then an error of Invalid Controller Identifier is returned. If the secondary controller is not configured appropriately (refer to section 8.26) or the primary controller is not enabled, then a status code of Invalid Secondary Controller State is returned. It is not an error to request a secondary controller be placed in the online state if that secondary controller is already in the online state.	...	
	Value	Description											
	...												
	7h	Secondary Controller Offline: Place the secondary controller in the Offline state and remove all Flexible Resources. If the Controller Identifier field does not correspond to a secondary controller associated with this primary controller, then a status code of Invalid Controller Identifier is returned. It is not an error to request a secondary controller be placed in the offline state if that secondary controller is already in the offline state.											
	8h	Secondary Controller Assign: Assign the number of controller resources specified in Number of Controller Resources to the secondary controller. If the Controller Identifier field does not correspond to a secondary controller associated with this primary controller, then an error of Invalid Controller Identifier is returned. If the secondary controller is not in the Offline state, then a status code of Invalid Secondary Controller State is returned.											
9h	Secondary Controller Online: Place the secondary controller in the Online state. If the Controller Identifier field does not correspond to a secondary controller associated with this primary controller, then an error of Invalid Controller Identifier is returned. If the secondary controller is not configured appropriately (refer to section 8.26) or the primary controller is not enabled, then a status code of Invalid Secondary Controller State is returned. It is not an error to request a secondary controller be placed in the online state if that secondary controller is already in the online state.												
...													

Modify section 7 as shown below:

7 I/O Commands

The submission queue entry (SQE) structure and the fields that are common to all I/O commands are defined in section 3.3.3. The completion queue entry (CQE) structure and the fields that are common to all I/O commands are defined in section 3.3.3.2. The command specific fields in the SQE and CQE structures (i.e., SQE Command Dwords 10-15, CQE Dword 0, and CQE Dword 1) for I/O cCommands supported across all I/O Command Sets are defined in this section.

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Modify section 8 as shown below:

8 Extended Capabilities

8.5 Controller Memory Buffer

...

A host ~~Host software~~ may configure the CMBMSC property so that CMB operates when the controller is assigned to a virtual machine that only supports NVM Express Base Specification revision 1.3 and earlier. To prevent that virtual machine from unintentionally clearing the CMBMSC property to 0h, the contents of the CMBMSC property are preserved across Controller Reset and Function Level Reset (refer to the [NVM Express NVMe over PCIe Transport Specification](#)).

...

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. The host shall ensure that all writes to the CMB that are needed for a command have been sent before updating the SQ Tail doorbell property. The Memory Write Request to the SQ Tail doorbell property shall not have the Relaxed Ordering bit set to '1' (refer to the [PCI Express Base Specification](#)), to ensure that prior writes to the CMB have completed.

...

8.21 Sanitize Operations

...

Controller attributes specific to sanitize operations include:

- The No-Deallocate Modifies Media After Sanitize (NODMMAS) field which indicates if media is modified by the controller after a sanitize operation successfully completes that had been requested with No-Deallocate After Sanitize set to '1' in the Sanitize command that started the sanitize operation; and
- No-Deallocate Inhibited (NDI) bit which indicates if the controller supports the No-Deallocate After Sanitize bit in the Sanitize ~~c~~Command.

...

To start a sanitize operation, the host submits a Sanitize command specifying one of the sanitize operation types (i.e., Block Erase, Overwrite, or Crypto Erase). The host sets command parameters, including the Allow Unrestricted Sanitize Exit bit and the No-Deallocate After Sanitize bit. After validating the Sanitize command parameters, the controller starts the sanitize operation in the background, updates the Sanitize Status log page and then completes the Sanitize command with Successful Completion status. If the sanitize operation is to be followed by an associated additional media modification operation (refer to NODMMAS in Figure 276), then the associated additional media modification operation shall be completed before the controller reports sanitize operation complete. If a Sanitize command is completed with any status code other than Successful Completion, then the controller shall not start the sanitize operation and shall not update the Sanitize Status log page. The controller ignores Critical Warning(s) in the SMART / Health Information log page (e.g., read only mode) and attempts to complete the sanitize operation requested. Refer to section 5 for further information about restrictions on Admin ~~c~~Commands during the processing of a Format NVM command.

...

8.21.1 Sanitize Operation Restrictions

...

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

While performing a sanitize operation and while a failed sanitize operation has occurred but successful recovery from that failure has not occurred, all enabled controllers and namespaces in the NVM subsystem are restricted to performing only a limited set of actions.

While a sanitize operation is in progress:

- All controllers in the NVM subsystem shall only process the Admin commands listed in Figure 141 subject to the additional restrictions stated in that figure;
- All I/O **c**Commands other than a Flush command shall be aborted with a status code of Sanitize In Progress;
- Processing of a Flush command is specified in section 7.1;
- Any command or command option that is not explicitly permitted in Figure 141 shall be aborted with a status code of Sanitize In Progress if fetched by any controller in the NVM subsystem; and
- The Persistent Memory Region shall be prevented from being enabled (i.e., setting PMRCTL.EN to '1' does not result in PMRSTS.NRDY being cleared to '0').

While a failed sanitize operation has occurred, a subsequent sanitize operation has not started and successful recovery from the failed sanitize operation has not occurred:

- All controllers in the NVM subsystem shall only process the Sanitize command (refer to section 5.24) and the Admin commands listed in Figure 141 subject to the additional restrictions noted in that figure;
- All I/O **c**Commands other than a Flush command (refer to section 7.1) shall be aborted with a status code of Sanitize Failed;
- The Sanitize command is permitted with action restrictions (refer to section 5.24);
- Aside from the Sanitize command, any other command or command option that is not explicitly permitted in Figure 141 shall be aborted with a status code of Sanitize Failed if fetched by any controller in the NVM subsystem; and
- The Persistent Memory Region shall be prevented from being enabled (i.e., setting PMRCTL.EN to '1' does not result in PMRSTS.NRDY being cleared to '0').

...

8.23 Standard Vendor Specific Command Format

Controllers may support the standard Vendor Specific command format defined in Figure 89. Host storage drivers may use the Number of Dwords fields to ensure that the application is not corrupting physical memory (e.g., overflowing a data buffer). The controller indicates support of this format in the Identify Controller data structure in Figure 276; refer to [the Admin Vendor Specific Command Configuration field](#) and [the I/O Command Set NVM Vendor Specific Command Configuration field](#).

...

8.26 Virtualization Enhancements

...

8.26.3 Secondary Controller States and Resource Configuration

...

To ensure that the host accurately detects capabilities of the secondary controller, the host should complete the following procedure to bring a secondary controller Online:

1. Use the Virtualization Management command to set the secondary controller to the Offline state;
2. Use the Virtualization Management command to assign VQ resources and VI resources;
3. Perform a Controller Level Reset. If the secondary controller is a VF, then this should be a VF Function Level Reset ([refer to the NVM Express NVMe over PCIe Transport Specification](#)); and
4. Use the Virtualization Management command to set the secondary controller to the Online state.

...

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Description of NVM Express NVM Command Set Specification 1.0d changes

< Change “non-volatile storage” to “non-volatile storage medium”>

< Change “non-volatile media” to “non-volatile storage medium”>

Modify section 1 as shown below:

1 Introduction

1.1 Overview

NVM Express® (NVMe®) Base Specification defines an interface for host software to communicate with a non-volatile memory subsystems (NVM subsystem) over a variety of memory-based transports and message-based transports.

...

Modify section 2 as shown below:

2 NVM Command Set Model

The NVM Express Base Specification defines a property level interface for host software to communicate with a non-volatile memory subsystem (NVM subsystem). This specification defines additional functionality for the NVM Command Set.

...

Modify section 3 as shown below:

3 I/O Commands for the NVM Command Set

This section specifies the NVM Command Set I/O cCommands.

...

Modify section 4 as shown below:

...

4 Admin Commands for the NVM Command Set

4.1 Admin Command behavior for the NVM Command Set

The Admin cCommands are as defined in the NVM Express Base Specification. The NVM Command Set specific behavior for Admin cCommands is described in this section.

...

4.1.5 Identify Command

...

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

4.1.5.1 NVM Command Set Identify Namespace Data Structure (CNS 00h)<Document change>

...

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 <i>n</i> consists of LBA 0 through (<i>n</i> - 1). The number of logical blocks is based on the formatted LBA size. Refer to section 2.1.1 for details on the usage of this field.
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 formatted 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 LBA size. Refer to section 2.1.1 for details on the usage of this field.
...		

...

4.2 I/O Command Set Specific Admin commands

In addition to the I/O Command Set Specific Admin commands defined in the NVM Express Base Specification, the NVM Command Set defines the Admin cCommands defined in this section.

...

Modify section 5 as shown below:

...

5 Extended Capabilities

5.8 Command Set Specific Capability

5.8.1 Get LBA Status

...

Upon receiving an LBA Status Information Alert asynchronous event, the host should send one or more Get Log Page commands for Log Identifier 0Eh with the Retain Asynchronous Event bit set to '1' in-order to read the entire LBA Status Information log page (refer to section 4.1.4.5).

Once the host has started reading the LBA Status Information log page with the Retain Asynchronous Event bit set to '1', the controller shall not modify the contents of that log page until the host re-reads the LBA Status Information log page with the Retain Asynchronous Event bit cleared to '0'.

...

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Description of NVM Express Key Value Command Set Specification

1.0d changes

Modify section 1 as shown below:

1 Introduction

1.1 Overview

NVM Express[®] (NVMe[®]) Base Specification defines an interface for host software to communicate with a non-volatile memory subsystems (NVM subsystem) over a variety of memory-based transports and message-based transports.

...

Modify section 2 as shown below:

2 Key Value Command Set Model

The NVM Express Base Specification defines a register level interface for host software to communicate with a non-volatile memory subsystem (NVM subsystem). This specification defines additional functionality for the Key Value Command Set.

...

Modify section 3 as shown below:

3 I/O Commands for the Key Value Command Set

This section specifies the Key Value Command Set I/O cCommands.

...

Modify section 4 as shown below:

4 Admin Commands for the Key Value Command Set

4.1 Admin Command behavior for the Key Value Command Set

The Admin cCommands are as defined in the NVM Express Base Specification. The Key Value Command Set specific behavior for Admin cCommands is described in this section.

...

Technical input submitted to the NVM Express[®] Workgroup is subject to the terms of the NVM Express[®] Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Description of NVM Express Zoned Namespace Command Set Specification 1.1d changes

< Change “non-volatile storage” to “non-volatile storage medium”>

<Editor: Change “non-volatile media” to “non-volatile storage medium”>

Modify section 1 as shown below:

1 Introduction

1.1 Overview

The NVM Express® (NVMe®) Base Specification defines an interface for host software to communicate with a non-volatile memory subsystem (NVM subsystem) over a variety of memory-based transports and message-based transports.

...

Modify section 2 as shown below:

2 Zoned Namespace Command Set Model

The NVM Express Base Specification defines an interface for host software to communicate with a non-volatile memory subsystem (NVM subsystem). This specification defines additional functionality for the Zoned Namespace Command Set.

...

Modify section 3 as shown below:

3 I/O Commands for the Zoned Namespace Command Set

This section specifies the NVM Command Set I/O cGCommands.

...

Modify section 4 as shown below:

4 Admin Commands for the Zoned Namespace Command Set

4.1 Admin Command behavior for the Zoned Namespace Command Set

The Admin cGCommands are as defined in the NVM Express Base Specification. The Zoned Namespace Command Set specific behavior for Admin cGCommands is defined in this section.

...

4.2 I/O Command Set Specific Admin commands

In addition to the I/O Command Set Specific Admin commands defined in the NVM Express Base Specification, the NVM Command Set defines the Admin cGCommands defined in this section.

...

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Description of Specification Changes for the NVM Express PCIe Transport Specification 1.0d

Modify section 1 as shown below:

1 Introduction

1.1 Overview

NVM Express® (NVMe®) Base Specification defines an interface for host software to communicate with a non-volatile memory subsystems (NVM subsystem) over a variety of memory-based transports and message-based transports.

...

Description of Specification Changes for the NVM Express RDMA Transport Specification 1.0c

Modify section 1 as shown below:

1 Introduction

1.1 Overview

NVM Express® (NVMe®) Base Specification defines an interface for host software to communicate with a non-volatile memory subsystems (NVM subsystem) over a variety of memory-based transports and message-based transports.

...

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Description of Specification Changes for the NVM Express TCP Transport Specification 1.0d

Modify section 1 as shown below:

1 Introduction

1.1 Overview

NVM Express® (NVMe®) Base Specification defines an interface for host software to communicate with a non-volatile memory subsystems (NVM subsystem) over a variety of memory-based transports and message-based transports.

...

Description of NVM Express Boot Specification 1.0 changes

Modify section 1 as shown below:

1 Introduction

The NVM Express® (NVMe®) Base Specification defines an interface for host software to communicate with a non-volatile memory subsystems (NVMe subsystem) over a variety of memory-based transports and message-based transports.

...

Modify section 3 as shown below:

3 Boot Mechanisms

...

3.1 ACPI NVMe Boot Firmware Table (NBFT)

...

3.1.2 NBFT Structure

...

3.1.2.4 Host Fabric Interface (HFI) Descriptor

...

3.1.2.4.1 HFI Transport Info Descriptor

...

3.1.2.4.1.1 HFI Transport Info Descriptor – NVMe/TCP

...

Figure 13: HFI Transport Info Descriptor – NVMe/TCP

Bytes	O/M ¹	Description
...		

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Figure 13: HFI Transport Info Descriptor – NVMe/TCP

Bytes	O/M ¹	Description										
06	M	HFI Transport Flags:										
		<table><tr><th>Bits</th><th>Description</th></tr><tr><td>07:03</td><td>Reserved</td></tr><tr><td>02</td><td>DHCP Override: If set to '1', then HFI information was populated by consuming the DHCP on this interface. If cleared to '0', then the HFI information was set administratively by a configuration interface to the driver and pre-OS environment.</td></tr><tr><td>01</td><td>Global Route vs. Link Local Override Flag: If set to '1', then the BIOS-pre-OS driver utilized this interface described by HFI to be the default route with highest priority. If cleared to '0', then routes are local to their own scope. Refer to section 3.1.2.4.1.X.</td></tr><tr><td>00</td><td>Descriptor Valid: If set to '1', then this descriptor is valid. If cleared to '0', then this descriptor is reserved.</td></tr></table>	Bits	Description	07:03	Reserved	02	DHCP Override: If set to '1', then HFI information was populated by consuming the DHCP on this interface. If cleared to '0', then the HFI information was set administratively by a configuration interface to the driver and pre-OS environment.	01	Global Route vs. Link Local Override Flag: If set to '1', then the BIOS-pre-OS driver utilized this interface described by HFI to be the default route with highest priority. If cleared to '0', then routes are local to their own scope. Refer to section 3.1.2.4.1.X.	00	Descriptor Valid: If set to '1', then this descriptor is valid. If cleared to '0', then this descriptor is reserved.
		Bits	Description									
		07:03	Reserved									
		02	DHCP Override: If set to '1', then HFI information was populated by consuming the DHCP on this interface. If cleared to '0', then the HFI information was set administratively by a configuration interface to the driver and pre-OS environment.									
01	Global Route vs. Link Local Override Flag: If set to '1', then the BIOS-pre-OS driver utilized this interface described by HFI to be the default route with highest priority. If cleared to '0', then routes are local to their own scope. Refer to section 3.1.2.4.1.X.											
00	Descriptor Valid: If set to '1', then this descriptor is valid. If cleared to '0', then this descriptor is reserved.											
...												
Notes:												
1. O/M definition: O = Optional, M = Mandatory.												

...

3.1.2.4.1.2 HFI Info Payload Host Name

...

3.1.2.4.1.X HFI Global and Local Routes

An HFI may optionally specify an IP Gateway, which indicates that a pre-OS driver route entry for a specific HFI descriptor exists. If the Global Route vs. Link Local Override Flag bit is cleared to '0' in the HFI Transport Flags field (refer to Figure 13), the HFI route specified by that descriptor is link local. If the Global Route vs. Link Local Override Flag bit is set to '1', the HFI route specified by that descriptor is the default route that was used by the pre-OS driver for the purposes of reachability. There shall be at most one descriptor per NBFT that sets the Global Route vs. Link Local Override Flag bit to '1'.

...

Description of NVM Express Management Interface Specification
1.2d changes

Modify section 2 as shown below:

2 Physical Layer

...

2.2 SMBus/I2C

...

If ~~ARP~~ the ~~Get UDID command~~ is supported by an NVM Subsystem, then all SMBus/I2C elements associated with that NVM Subsystem shall use the ~~SMBus-Address-Resolution-Protocol~~ Unique Device Identifier (UDID) shown in Figure 17. ~~The UDID and the Get UDID command are defined by the SMBus Address Resolution Protocol (ARP) (refer to the SMBus Specification).~~ The ~~ARP~~ UDID is a unique identifier. The UDID ~~Type field~~ ~~Vendor ID bits 31:30~~ ~~allow~~ allows up to four SMBus/I2C elements to be grouped together ~~with~~ within the same NVM Subsystem. ~~Those elements shall indicate UDIDs that are identical except for their UDID Type fields. The only difference within this group of UDIDs is the most-significant two bits of the Vendor Specific ID.~~ This fact may be used by the Management Controller to associate an SMBus/I2C Management Endpoint with its corresponding FRU Information Device.

If there are multiple NVM Subsystems ~~that support the Get UDID command~~ in an ~~SMBus-ARP-capable~~ NVMe Storage Device or NVMe Enclosure, then ~~for each NVM Subsystem the Unique-NVM-Storage value of the UDID Device ID fields of the UDID~~ shall be ~~sequential~~ ~~incremented by one for each NVM Subsystem.~~

If the Upstream Connector has an SMBus/I2C port, then the FRU Information Device associated with that connector shall be present directly on the SMBus/I2C channel connected to the Upstream Connector.

...

< **Note to editor:** The changes below to the UDID Type field description second sentence are based on NVMe-MI Next of 2023-08-23. That sentence is different than in NVMe- MI 1.2c. >

Figure 17: ~~SMBus/I2C Element~~ UDID

Bits	Field	Description
...		

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Figure 17: SMBus/I2C Element UDID

Bits	Field	Description																		
31:00	Vendor Specific ID	This field ensures all UDIDs from a vendor are unique and is used to associate elements implemented within an NVMe Storage Device or NVMe Enclosure.																		
		<table><tr><th>Bits</th><th>Description</th></tr><tr><td rowspan="6">31:30</td><td>UDID Type: This field distinguishes which NVM Subsystem that implements multiple SMBus elements is providing the UDID. Note that Management Controllers complaint compliant to version 1.0 and earlier of this specification may be incompatible with devices NVM Subsystems using values 1h and 3h.</td></tr><tr><td><table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>FRU Information Device</td></tr><tr><td>1h</td><td>SMBus/I2C Mux</td></tr><tr><td>2h</td><td>Management Endpoint</td></tr><tr><td>3h</td><td>Vendor Specific Devices</td></tr></table></td></tr><tr><td rowspan="2">29:00</td><td>UDID Device ID: This field indicates contains a unique vendor assigned ID for the NVM Subsystem. The ID is different in each NVM Subsystem instance This field shall contain a value that results in a unique UDID as specified by the SMBus Specification, and remains static during the life of the device NVM Subsystem.</td></tr><tr><td></td></tr></table>	Bits	Description	31:30	UDID Type: This field distinguishes which NVM Subsystem that implements multiple SMBus elements is providing the UDID. Note that Management Controllers complaint compliant to version 1.0 and earlier of this specification may be incompatible with devices NVM Subsystems using values 1h and 3h.	<table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>FRU Information Device</td></tr><tr><td>1h</td><td>SMBus/I2C Mux</td></tr><tr><td>2h</td><td>Management Endpoint</td></tr><tr><td>3h</td><td>Vendor Specific Devices</td></tr></table>	Value	Description	0h	FRU Information Device	1h	SMBus/I2C Mux	2h	Management Endpoint	3h	Vendor Specific Devices	29:00	UDID Device ID: This field indicates contains a unique vendor assigned ID for the NVM Subsystem. The ID is different in each NVM Subsystem instance This field shall contain a value that results in a unique UDID as specified by the SMBus Specification, and remains static during the life of the device NVM Subsystem.	
		Bits	Description																	
		31:30	UDID Type: This field distinguishes which NVM Subsystem that implements multiple SMBus elements is providing the UDID. Note that Management Controllers complaint compliant to version 1.0 and earlier of this specification may be incompatible with devices NVM Subsystems using values 1h and 3h.																	
			<table><tr><th>Value</th><th>Description</th></tr><tr><td>0h</td><td>FRU Information Device</td></tr><tr><td>1h</td><td>SMBus/I2C Mux</td></tr><tr><td>2h</td><td>Management Endpoint</td></tr><tr><td>3h</td><td>Vendor Specific Devices</td></tr></table>	Value		Description	0h	FRU Information Device	1h	SMBus/I2C Mux	2h	Management Endpoint	3h	Vendor Specific Devices						
			Value	Description																
0h	FRU Information Device																			
1h	SMBus/I2C Mux																			
2h	Management Endpoint																			
3h	Vendor Specific Devices																			
29:00	UDID Device ID: This field indicates contains a unique vendor assigned ID for the NVM Subsystem. The ID is different in each NVM Subsystem instance This field shall contain a value that results in a unique UDID as specified by the SMBus Specification, and remains static during the life of the device NVM Subsystem.																			

...

Modify section 5 as shown below:

5 Management Interface Command Set

...

5.7 Read NVMe-MI Data Structure

...

Figure 92: Read NVMe-MI Data Structure – NVMe Management Dword 0

Bits	Description																								
31:24	Data Structure Type (DTYP): This field specifies the data structure to return.																								
	<table><tr><th>Value</th><th>Definition</th><th>Reference</th></tr><tr><td>00h</td><td>NVM Subsystem Information</td><td>5.7.1</td></tr><tr><td>01h</td><td>Port Information</td><td>5.7.2</td></tr><tr><td>02h</td><td>Controller List</td><td>5.7.3</td></tr><tr><td>03h</td><td>Controller Information</td><td>5.7.4</td></tr><tr><td>04h</td><td>Optionally Supported Command List</td><td>5.7.5</td></tr><tr><td>05h</td><td>Management Endpoint Buffer Command Support List</td><td>5.7.6</td></tr><tr><td>06h to FFh</td><td>Reserved</td><td></td></tr></table>	Value	Definition	Reference	00h	NVM Subsystem Information	5.7.1	01h	Port Information	5.7.2	02h	Controller List	5.7.3	03h	Controller Information	5.7.4	04h	Optionally Supported Command List	5.7.5	05h	Management Endpoint Buffer Command Support List	5.7.6	06h to FFh	Reserved	
	Value	Definition	Reference																						
	00h	NVM Subsystem Information	5.7.1																						
	01h	Port Information	5.7.2																						
	02h	Controller List	5.7.3																						
	03h	Controller Information	5.7.4																						
	04h	Optionally Supported Command List	5.7.5																						
05h	Management Endpoint Buffer Command Support List	5.7.6																							
06h to FFh	Reserved																								
23:16	Port Identifier (PORTID): This field contains the identifier of the port whose data structure is returned. If the DTYP field value is 01h (Port Information) or 05h (i.e., Management Endpoint Buffer Command Support List), then this field contains the Port Identifier whose information shall be returned. For all other non-reserved values of the DTYP field, this field should be ignored by the Management Endpoint.																								
	Controller Identifier (CTRLID): This field specifies the Controller Identifier of the Controller whose data structure is returned-used during processing of the command as follows: If the DTYP field value is 02h (Controller List), 03h (Controller Information), or 04h (Optionally Supported Command List), then this field specifies the Controller Identifier of the Controller in the NVM Subsystem whose information shall be returned.																								

Technical input submitted to the NVMe Express® Workgroup is subject to the terms of the NVMe Express® Participant's agreement. Copyright © 2008-2024 NVMe Express, Inc.

Figure 92: Read NVMe-MI Data Structure – NVMe Management Dword 0

Bits	Description								
	<p>If the DTYP field value is 04h (Optionally Supported Command List), then this field is only applicable for commands in the Optionally Supported Command List Data Structure with NMIMT set to a value of 02h (NVMe Admin Command) sent via the out-of-band mechanism and shall be ignored for commands with NMIMT set to any value other than 02h or sent via the in-band tunneling mechanism.</p> <table> <tr> <th>DTYP Value 1</th><th>CTRLID Usage</th></tr> <tr> <td>02h</td><td>This field contains the Controller Identifier used to return a Controller List data structure for the NVM Subsystem as described in section 5.7.3.</td></tr> <tr> <td>03h</td><td>This field contains the Controller Identifier of the Controller for which the information is returned as described in section 5.7.4.</td></tr> <tr> <td>04h</td><td>This field contains the Controller Identifier of the Controller used to filter which optional NVMe Express Admin Command Set commands (i.e., the NMIMT field is set to 02h) are returned in the Optionally Supported Command List data structure entries as described in section 5.7.5.</td></tr> </table> <p>Notes: 1. For all other non-reserved values of the DTYP field, this field should be ignored by the Management Endpoint.</p> <p><Editors note: the above "should" is a "shall" in the .NEXT version (see ECN115).></p>	DTYP Value 1	CTRLID Usage	02h	This field contains the Controller Identifier used to return a Controller List data structure for the NVM Subsystem as described in section 5.7.3.	03h	This field contains the Controller Identifier of the Controller for which the information is returned as described in section 5.7.4.	04h	This field contains the Controller Identifier of the Controller used to filter which optional NVMe Express Admin Command Set commands (i.e., the NMIMT field is set to 02h) are returned in the Optionally Supported Command List data structure entries as described in section 5.7.5.
DTYP Value 1	CTRLID Usage								
02h	This field contains the Controller Identifier used to return a Controller List data structure for the NVM Subsystem as described in section 5.7.3.								
03h	This field contains the Controller Identifier of the Controller for which the information is returned as described in section 5.7.4.								
04h	This field contains the Controller Identifier of the Controller used to filter which optional NVMe Express Admin Command Set commands (i.e., the NMIMT field is set to 02h) are returned in the Optionally Supported Command List data structure entries as described in section 5.7.5.								

Figure 93: Read NVMe-MI Data Structure – NVMe Management Dword 1

Bits	Description
31:08	Reserved
07:00	<p>I/O Command Set Identifier (IOCSI): If the DTYP field value is 04h (i.e., Optionally Supported Command List) or 05h (i.e., Management Endpoint Buffer Command Support List), then for commands with the NMIMT field set to a value of 02h (i.e., NVMe Admin Command) in the:</p> <ul style="list-style-type: none"> a) Optionally Supported Command List data structure; or b) Management Endpoint Buffer Supported Command List data structure, <p>this field specifies the I/O Command Set that shall be used to select the optional I/O Command Set Specific Admin commands. For more information about I/O Command Sets refer to the NVM Express Base Specification.</p> <p><Note to editor: Globally add "i.e.," in front of DTYP descriptions that are in parentheses such as (Optionally Supported Command List) and (Management Endpoint Buffer Command Support List) as shown in the prior paragraph.></p> <p>For all other non-reserved values of the DTYP field, other than the values 04h and 05h, this field is not applicable and should be ignored by the Management Endpoint.</p> <p><Note to editor: ECN115 converted the "should" in the prior sentence to a "shall" in the non-errata version of the integration spec and remains a "shall" in that version of the spec.></p> <p>If the DTYP field is 04h or 05h, then for commands with the NMIMT field set to any non-reserved value other than 02h in the Optionally Supported Command List data structure or Management Endpoint Buffer Supported Command List data structure, this field is not applicable and should be ignored by the Management Endpoint.</p> <p><Note to editor: Convert the "should" in the prior sentence to a "shall" in the non-errata version of the integration spec.></p> <p>The I/O Command Set specified by this field is not required to be enabled (refer to the NVM Express Base Specification).</p>

5.7.1 NVM Subsystem Information Response Data

The NVM Subsystem Information data structure contains information about the NVM Subsystem. The Port Identifier field and the Controller Identifier field in the NVMe Management Dword 0 field are reserved. The format of the NVM Subsystem Information data structure is shown in Figure 95.

...

5.7.2 Port Information Response Data

The Port Information data structure contains information about a port within the NVM Subsystem. The Port Identifier field in the NVMe Management Dword 0 field specifies the port. The Controller Identifier field in the NVMe Management Dword 0 field is reserved. The format of the Port Information data structure is shown in Figure 96.

Figure 96: Port Information Data Structure

Bytes	Description		
00	Port Type: Specifies the port type.		
	Value	Definition	Reference
	0h	Inactive	
	1h	PCIe	Figure 97
	2h	SMBus	Figure 98
...	3h to FFh	Reserved	

...

5.7.3 Controller List Response Data

The Controller List data structure shall contain a list of all NVMe Controllers in the NVM Subsystem (e.g., all I/O Controllers, Administrative Controllers, primary Controllers, secondary Controllers) that have a Controller Identifier that is greater than or equal to the value specified in the Controller Identifier (CTRLID) field in the NVMe Management Dword 0 field. A Controller List may contain up to 2,047 Controller Identifiers. Refer to the NVM Express Base Specification for a definition of the Controller List.

...

5.7.4 Controller Information Response Data

The Controller Information data structure shall contain information about the Controller in the NVM Subsystem that is specified in the Controller Identifier field in the NVMe Management Dword 0 field. The format of the Controller Information data structure is shown in Figure 99.

Figure 99: Controller Information Data Structure

Bytes	Description
...	

5.7.5 Optionally Supported Command List Response Data

For Management Interface Command Set commands and PCIe Command Set commands, the Optionally Supported Command List data structure shall indicate a list of optional commands supported by the Responder that received the Read NVMe MI Data Structure command.

For the out-of-band mechanism, the Optionally Supported Command List data structure for NVM Express Admin Command Set commands shall indicate a list of optional commands supported by the specified

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Controller (refer to the Controller ID field in the NVMe Management Dword 0 field in Figure 92) on the Management Endpoint that received the Read NVMe-MI Data Structure command.

The Optionally Supported Command List data structure shall contain a list of the following optional commands that a Responder supports on the interface over which the Read NVMe-MI Data Structure command was received:

- PCIe Command Set commands (refer to Figure 130);
- Management Interface Command Set commands (refer to Figure 60); and
- NVMe Express Admin Command Set commands (refer to Figure 116) supported by the Management Endpoint on the Controller specified by the Controller Identifier (CTRLID) field in the NVMe Management Dword 0 field.

For the in-band tunneling mechanism, the Optionally Supported Command List data structure does not contain any NVMe Express Admin Command Set commands because NVMe Express Admin Command Set commands are prohibited in the in-band tunneling mechanism.

If the DTYP field value is 04h (i.e., Optionally Supported Command List) and the NMIMT field value is 02h (i.e., NVMe Admin Command), then the I/O Command Set Identifier (IOCSI) field in the NVMe Management Dword 1 field specifies the I/O Command Set for the I/O Command Set Specific Admin commands that shall be returned in the Optionally Supported Command List data structure.

The Controller Identifier (CTRLID) field in the NVMe Management Dword 0 field shall be ignored for all optionally supported commands other than NVMe Express Admin Command Set commands.

The Optionally Supported Command List data structure shall contain no more than 2,047 commands and shall be minimally sized (e.g., the data structure size is 2 bytes if there are no optionally supported commands and the data structure size is 4 bytes if there is one optionally supported command). The format of the Optionally Supported Command List data structure is shown in Figure 100.

...

Figure 101: Optionally Supported Command Data Structure

Bytes	Description	
00	Command Type: This field specifies the type of command used by the optionally supported command.	
	Bits	Description
	7	Reserved
	6:3	NVMe-MI Message Type (NMIMT): This field specifies the NVMe-MI Message Type. Refer to the NMIMT field shown in Figure 19 .
	2:0	Reserved
01	Opcode: This field specifies the opcode used for the optionally supported command.	

5.7.6 Management Endpoint Buffer Command Support List Response Data

If the Management Endpoint Buffer Size field in the Port Information Data Structure is not 0h, then returning of the Management Endpoint Buffer Command Support List data structure shall be supported by the Management Endpoint. If the Management Endpoint Buffer Size field in the Port Information Data Structure is 0h, then the Data Structure Type value for Management Endpoint Buffer Command Support List is reserved.

The Management Endpoint Buffer Command Support List data structure contains a list of commands that support the use of the Management Endpoint Buffer. If the DTYP field value is 05h (i.e., Management Endpoint Buffer Command Support List) and the NMIMT field value is 02h (i.e., NVMe Admin Command), then the I/O Command Set Identifier (IOCSI) field in the NVMe Management Dword 1 field selects the I/O Command Set for the I/O Command Set Specific Admin commands that are returned in the Management

Technical input submitted to the NVMe Express® Workgroup is subject to the terms of the NVMe Express® Participant's agreement. Copyright © 2008-2024 NVMe Express, Inc.

Endpoint Buffer Command Support List data structure. The data structure may contain up to 2,047 commands, and shall be minimally sized (i.e., if there is 1 optionally supported command, the data structure is 4 bytes total).

The list of commands that support the Management Endpoint Buffer may be different among Management Endpoints within the NVM Subsystem. The Port Identifier (PORTID) field in the NVMe Management Dword 0 field of the Read NVMe-MI Data Structure specifies the port of the Management Endpoint whose Management Endpoint Buffer Command Support List data structure is returned. The format of the Management Endpoint Buffer Supported Command List data structure is shown in Figure 102.

...

Figure 103: Management Endpoint Buffer Supported Command Data Structure

Bytes	Description	
00	Command Type: This field specifies the type of command that supports the Management Endpoint Buffer.	
	Bits	Description
	7	Reserved
	6:3	NVMe-MI Message Type (NMIMT): This field specifies the NVMe-MI Message Type. Refer to the NMIMT field shown in Figure 19 .
	2:0	Reserved
01	Opcode: This field specifies the opcode of the command that supports the Management Endpoint Buffer.	

...

Modify section 8 as shown below:

8 Management Architecture

...

8.2 Vital Product Data

...

8.2.2 Product Info Area (offset 8 bytes)

...

Figure 152: Product Info Area Factory Default Values

Factory Default	Description
...	...
Impl Spec	Manufacturer Name Type/Length (MNTL): This field indicates the type and length of the Manufacturer Name field. The maximum length is 8.
Impl Spec	Manufacturer Name (MNAME): This field indicates the Manufacturer name in 8-bit ASCII. Unused bytes should be NULL characters. The Manufacturer name in this field should correspond to that in the PCI Subsystem Vendor ID (SSVID) field and the IEEE OUI Identifier fields in the Identify Controller Data Structure and should not be padded. If padded, then those pad bytes should be NULL characters.

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.

Figure 152: Product Info Area Factory Default Values

Factory Default	Description
Impl Spec	Product Name Type/Length (PNTL): This field indicates the type and length of the Product Name field. The maximum length is 24.
Impl Spec	Product Name (PNAME): This field indicates the P product name in 8-bit ASCII. Unused bytes should be NULL characters; and should not be padded. If padded, then those pad bytes should be NULL characters.
Impl Spec	Product Part/Model Number Type/Length (PPMNNTL): This field indicates the type and length of the Product Part/Model Number field. The maximum length is 40.
Impl Spec	Product Part/Model Number (PPMN): This field indicates the P product P part/ M model n Number in 8-bit ASCII. Unused bytes should be NULL characters. This field should contain the same value as the Model Number (N MN) field in the NVMe Identify Controller D data S structure with the exclusion of any spaces (i.e., ASCII character 20h) added in that MN field for padding. If padding is added to this field, then those pad bytes should be NULL characters.
Impl Spec	Product Version Type/Length (PVTL): This field indicates the type and length of the Product Version field. The maximum length is 2.
Impl Spec	Product Version (PVER): This field indicates the P product V version in 8-bit ASCII. Unused bytes should be NULL characters; and should not be padded. If padded, then those pad bytes should be NULL characters.
Impl Spec	Product Serial Number Type/Length (PSNTL): This field indicates the type and length of the Product Serial Number field. The maximum length is 20.
Impl Spec	Product Serial Number (PSN): This field indicates the P product S serial N number in 8-bit ASCII. Unused bytes should be NULL characters. This field should contain the same value as the Serial Number (SN) field in the NVMe Identify Controller D data S structure with the exclusion of any spaces (i.e., ASCII character 20h) added in that SN field for padding. If padding is added to this field, then those pad bytes should be NULL characters.
...	...

Technical input submitted to the NVM Express® Workgroup is subject to the terms of the NVM Express® Participant's agreement. Copyright © 2008-2024 NVM Express, Inc.