



# **NVM Express<sup>®</sup>**

## **Base Specification**

**Revision 2.1**  
**August 5th, 2024**

*Please send comments to [info@nvmexpress.org](mailto:info@nvmexpress.org)*

NVM Express® Base Specification, Revision 2.1 is available for download at <https://nvmexpress.org>. The NVM Express Base Specification, Revision 2.1 incorporates NVM Express Base Specification, Revision 2.0, ratified on June 3, 2021, ECN 001, ECN102, ECN105, ECN106, ECN107, ECN109, ECN110, ECN111, ECN112, ECN113, ECN114, ECN 115, ECN116, ECN117, ECN118, ECN119, ECN120, ECN121, ECN122, TP4029a, TP4034a, TP4055, TP4058, TP4074a, TP4077, TP4094a, TP4095a, TP 4097a, TP 4099, TP4100, TP 4103, TP4104a, TP4109a, TP4110a, TP 4112, TP 4113a, TP 4114, TP4119b, TP 4124, TP4129, TP4130a, TP4135, TP4136, TP4142, TP4145, TP4146b, TP4150, TP4152, TP4155, TP4156a, TP4159, TP4160, TP4162a, TP4165, TP4167, TP4169, TP4170, TP4171, TP4181, TP4182, TP6011a, TP6021, TP6032, TP6034a, TP6036, TP6038, TP8009, TP8010a, TP 8013a, TP8012, TP 8014, TP8016, TP8017a, TP8018, TP8019a, TP8020, TP8021, TP8024a and TP8025 (refer to <https://nvmexpress.org/changes-in-nvme-revision-2-1> for details).

#### SPECIFICATION DISCLAIMER

#### **LEGAL NOTICE:**

© Copyright 2008 to 2024 NVM Express, Inc. ALL RIGHTS RESERVED.

This NVM Express Base Specification, Revision 2.1 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 NVM Express Base Specification, Revision 2.1 subject, however, to the Member’s continued compliance with the Company’s Intellectual Property Policy and Bylaws and the Member’s Participation Agreement.

**NOTICE TO NON-MEMBERS OF NVM EXPRESS, INC.:** If you are not a Member of NVM Express, Inc. and you have obtained a copy of this document, you only have a right to review this document or make reference to or cite this document. Any such references or citations to this document must acknowledge NVM Express, Inc. copyright ownership of this document. The proper copyright citation or reference is as follows: “© 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. Redfish® is a registered trademark of DMTF.

NVM Express Workgroup  
c/o VTM, Inc.  
3855 SW 153<sup>rd</sup> Drive  
Beaverton, OR 97003 USA  
[info@nvmexpress.org](mailto:info@nvmexpress.org)

## Table of Contents

<b>1</b>	<b>INTRODUCTION .....</b>	<b>1</b>
1.1	Overview.....	1
1.1.1	NVM Express® Specification Family.....	1
1.2	Scope.....	2
1.3	Outside of Scope .....	2
1.4	Conventions.....	2
1.4.1	Keywords .....	2
1.4.2	Numerical Descriptions .....	3
1.4.3	Byte, Word, and Dword Relationships.....	5
1.5	Definitions .....	5
1.5.1	admin label.....	5
1.5.2	admin label ASCII.....	5
1.5.3	admin label UTF-8.....	6
1.5.4	Admin Queue .....	6
1.5.5	Administrative controller .....	6
1.5.6	allocated namespace .....	6
1.5.7	Allowed Host List.....	6
1.5.8	arbitration burst .....	6
1.5.9	arbitration mechanism .....	6
1.5.10	association .....	6
1.5.11	audit .....	6
1.5.12	authentication commands .....	6
1.5.13	cache.....	6
1.5.14	candidate command.....	6
1.5.15	capsule.....	7
1.5.16	Centralized Discovery controller (CDC).....	7
1.5.17	Channel.....	7
1.5.18	command completion .....	7
1.5.19	command submission.....	7
1.5.20	controller .....	7
1.5.21	Controller Reset .....	7
1.5.22	Directive .....	7
1.5.23	Direct Discovery controller (DDC) .....	7
1.5.24	Discovery controller.....	8
1.5.25	discovery information .....	8
1.5.26	Discovery Service.....	8
1.5.27	dispersed namespace .....	8
1.5.28	dynamic controller .....	8
1.5.29	Domain .....	8
1.5.30	embedded management controller.....	8
1.5.31	emulated controller.....	8
1.5.32	Endurance Group.....	8
1.5.33	Entry Key.....	8
1.5.34	Exported Namespace.....	8
1.5.35	Exported NVM Resources.....	8
1.5.36	Exported NVM Subsystem .....	9
1.5.37	Exported Port .....	9
1.5.38	Exported Port ID.....	9
1.5.39	fabric (network fabric).....	9
1.5.40	Fabric Zoning .....	9
1.5.41	firmware/boot partition image update command sequence .....	9
1.5.42	firmware slot.....	9
1.5.43	host .....	9
1.5.44	host-accessible memory.....	9
1.5.45	host management agent.....	9
1.5.46	host memory.....	9
1.5.47	idempotent command.....	9
1.5.48	Identify Controller data structures.....	10
1.5.49	Identify Namespace data structures.....	10

1.5.50	I/O command.....	10
1.5.51	I/O Completion Queue.....	10
1.5.52	I/O controller.....	10
1.5.53	I/O Submission Queue .....	10
1.5.54	Media Unit.....	10
1.5.55	memory-based controller.....	10
1.5.56	message-based controller .....	10
1.5.57	metadata .....	10
1.5.58	MMC.....	10
1.5.59	MMH.....	10
1.5.60	MMHD .....	11
1.5.61	MMHS .....	11
1.5.62	namespace.....	11
1.5.63	namespace ID (NSID) .....	11
1.5.64	NVM .....	11
1.5.65	NVM Set.....	11
1.5.66	NVM subsystem .....	11
1.5.67	NVM subsystem port.....	11
1.5.68	NVMe over Fabrics.....	11
1.5.69	NVMe Transport .....	11
1.5.70	NVMe Transport binding specification.....	11
1.5.71	participating NVM subsystem.....	11
1.5.72	physical fabric interface (physical ports).....	12
1.5.73	Placement Handle .....	12
1.5.74	Placement Identifier.....	12
1.5.75	Port ID .....	12
1.5.76	Ports List .....	12
1.5.77	Power Loss Acknowledge (PLA) .....	12
1.5.78	Power Loss Notification (PLN).....	12
1.5.79	primary controller.....	12
1.5.80	private namespace .....	12
1.5.81	property .....	12
1.5.82	Reclaim Group (RG).....	12
1.5.83	Reclaim Unit (RU) .....	12
1.5.84	Reclaim Unit Handle (RUH) .....	12
1.5.85	rotational media.....	13
1.5.86	Runtime D3 (Power Removed).....	13
1.5.87	sanitize operation .....	13
1.5.88	sanitization target .....	13
1.5.89	secondary controller.....	13
1.5.90	shared namespace.....	13
1.5.91	specified namespace.....	13
1.5.92	spindown .....	13
1.5.93	spinup.....	13
1.5.94	static controller .....	13
1.5.95	Underlying Namespace .....	13
1.5.96	Underlying Namespace List .....	13
1.5.97	Underlying NVM Subsystem .....	14
1.5.98	Underlying Port.....	14
1.5.99	user data .....	14
1.6	I/O Command Set specific definitions used in this specification .....	14
1.6.1	Endurance Group Host Read Command.....	14
1.6.2	Format Index .....	14
1.6.3	SMART Data Units Read Command .....	14
1.6.4	SMART Host Read Command .....	14
1.6.5	User Data Format.....	14
1.6.6	User Data Out Command.....	14
1.7	NVM Command Set specific definitions used in this specification .....	14
1.7.1	logical block.....	14
1.7.2	logical block address (LBA).....	14
1.8	References .....	14
1.9	References Under Development .....	16



<b>2</b>	<b>THEORY OF OPERATION .....</b>	<b>17</b>
2.1	Memory-Based Transport Model (PCIe) .....	19
2.2	Message-Based Transport Model (Fabrics) .....	21
2.2.1	Fabrics and Transports .....	21
2.2.2	NVM Subsystem Ports for Fabrics .....	22
2.2.3	Discovery Service.....	23
2.2.4	Capsules and Data Transfer .....	23
2.2.5	Authentication.....	24
2.3	NVM Storage Model .....	24
2.3.1	Storage Entities.....	24
2.3.2	I/O Command Sets.....	30
2.3.3	NVM Subsystem Examples.....	30
2.4	Extended Capabilities Theory .....	32
2.4.1	Multi-Path I/O and Namespace Sharing.....	32
2.4.2	Asymmetric Controller Behavior.....	35
<b>3</b>	<b>NVM EXPRESS ARCHITECTURE.....</b>	<b>36</b>
3.1	NVM Controller Architecture.....	36
3.1.1	Memory-Based Controller Architecture (PCIe).....	36
3.1.2	Message-Based Controller Architecture (Fabrics).....	36
3.1.3	Controller Types.....	37
3.1.4	Controller Properties .....	49
3.2	NVM Subsystem Entities .....	75
3.2.1	Namespaces .....	75
3.2.2	NVM Sets.....	77
3.2.3	Endurance Groups .....	79
3.2.4	Reclaim Groups, Reclaim Unit Handles, and Reclaim Units .....	81
3.2.5	Domains and Divisions.....	82
3.3	NVM Queue Models .....	85
3.3.1	Memory-based Transport Queue Model (PCIe) .....	85
3.3.2	Message-based Transport Queue Model (Fabrics).....	88
3.3.3	Queueing Attributes.....	97
3.4	Command Processing .....	98
3.4.1	Command Ordering Requirements.....	98
3.4.2	Fused Operations.....	98
3.4.3	Atomic Operations.....	99
3.4.4	Command Arbitration .....	99
3.4.5	Outstanding Commands.....	101
3.5	Controller Initialization .....	102
3.5.1	Memory-based Controller Initialization (PCIe).....	102
3.5.2	Message-based Controller Initialization (Fabrics).....	103
3.5.3	Controller Ready Modes During Initialization .....	106
3.5.4	Controller Ready Timeouts During Initialization .....	108
3.6	Shutdown Processing.....	110
3.6.1	Memory-based Controller Shutdown (PCIe).....	111
3.6.2	Message-based Controller Shutdown (Fabrics) .....	112
3.6.3	NVM Subsystem Shutdown.....	113
3.7	Resets.....	117
3.7.1	NVM Subsystem Reset .....	117
3.7.2	Controller Level Reset.....	118
3.7.3	Queue Level Reset.....	119
3.8	NVM Capacity Model.....	119
3.8.1	Overview .....	119
3.8.2	Media Unit Organization Examples .....	120
3.8.3	Capacity Reporting.....	123
3.9	Keep Alive .....	124
3.9.1	Keep Alive Timer Configuration.....	125
3.9.2	Keep Alive Timer Activation .....	125
3.9.3	Command Based Keep Alive .....	125

3.9.4	Traffic Based Keep Alive .....	127
3.9.5	Keep Alive Timeout Cleanup .....	129
3.10	Privileged Actions .....	130
3.11	Firmware Update Process .....	130
<b>4</b>	<b>DATA STRUCTURES .....</b>	<b>133</b>
4.1	Submission Queue Entry .....	133
4.1.1	Admin Command and I/O Command Common SQE .....	133
4.1.2	Fabrics Command Common SQE .....	137
4.2	Completion Queue Entry .....	138
4.2.1	Admin Command and I/O Command Common CQE .....	138
4.2.2	Fabrics Command Common CQE .....	139
4.2.3	Status Field Definition .....	139
4.2.4	Phase Tag .....	149
4.3	Data Pointer Layouts (PRPs and SGLs) .....	151
4.3.1	Physical Region Page Entry and List .....	151
4.3.2	Scatter Gather List (SGL) .....	153
4.3.3	Metadata Region (MR) .....	159
4.4	Feature Values .....	159
4.5	Identifier Format and Layout (Informative) .....	161
4.5.1	PCI Vendor ID (VID) and PCI Subsystem Vendor ID (SSVID) .....	161
4.5.2	Serial Number (SN) and Model Number (MN) .....	162
4.5.3	IEEE OUI Identifier (IEEE) .....	162
4.5.4	IEEE Extended Unique Identifier (EUI64) .....	162
4.5.5	Namespace Globally Unique Identifier (NGUID) .....	163
4.5.6	Universally Unique Identifier (UUID) .....	164
4.6	List Data Structures .....	164
4.6.1	Controller List .....	164
4.6.2	Namespace List .....	164
4.7	NVMe Qualified Names .....	165
4.7.1	Unique Identifier .....	166
4.8	UTF-8 String Processing .....	167
<b>5</b>	<b>ADMIN COMMAND SET .....</b>	<b>169</b>
5.1	Common Admin Commands .....	172
5.1.1	Abort command .....	172
5.1.2	Asynchronous Event Request command .....	173
5.1.3	Capacity Management command .....	181
5.1.4	Controller Data Queue command .....	184
5.1.5	Device Self-test command .....	188
5.1.6	Directive Receive command .....	190
5.1.7	Directive Send command .....	191
5.1.8	Firmware Commit command .....	192
5.1.9	Firmware Image Download command .....	194
5.1.10	Format NVM command .....	195
5.1.11	Get Features command .....	198
5.1.12	Get Log Page command .....	201
5.1.13	Identify command .....	295
5.1.14	Keep Alive command .....	348
5.1.15	Lockdown command .....	348
5.1.16	Migration Receive command .....	350
5.1.17	Migration Send command .....	353
5.1.18	NVMe-MI Receive command .....	361
5.1.19	NVMe-MI Send command .....	361
5.1.20	Namespace Attachment command .....	361
5.1.21	Namespace Management command .....	363
5.1.22	Sanitize command .....	365
5.1.23	Security Receive command .....	368
5.1.24	Security Send command .....	369
5.1.25	Set Features command .....	370

5.1.26	Track Receive command.....	411
5.1.27	Track Send command .....	414
5.2	Memory-Based Transport Admin Commands (PCIe) .....	418
5.2.1	Create I/O Completion Queue command .....	419
5.2.2	Create I/O Submission Queue command.....	420
5.2.3	Delete I/O Completion Queue command.....	422
5.2.4	Delete I/O Submission Queue command .....	422
5.2.5	Doorbell Buffer Config command .....	423
5.2.6	Virtualization Management command .....	424
5.3	Message-Based Transport Admin Commands (Fabrics) .....	426
5.3.1	Clear Exported NVM Resource Configuration command .....	426
5.3.2	Create Exported NVM Subsystem command.....	426
5.3.3	Discovery Information Management command .....	427
5.3.4	Fabric Zoning Lookup command.....	435
5.3.5	Fabric Zoning Receive command.....	435
5.3.6	Fabric Zoning Send command .....	437
5.3.7	Manage Exported Namespace command .....	438
5.3.8	Manage Exported NVM Subsystem command.....	440
5.3.9	Manage Exported Port command.....	444
5.3.10	Send Discovery Log Page command .....	447
<b>6</b>	<b>FABRICS COMMAND SET .....</b>	<b>449</b>
6.1	Authentication Receive Command and Response.....	449
6.2	Authentication Send Command and Response .....	450
6.3	Connect Command and Response .....	451
6.4	Disconnect Command and Response .....	456
6.5	Property Get Command and Response .....	457
6.6	Property Set Command and Response.....	458
<b>7</b>	<b>I/O COMMANDS.....</b>	<b>460</b>
7.1	Cancel command.....	460
7.1.1	Command Completion.....	462
7.2	Flush command .....	463
7.2.1	Command Completion.....	463
7.3	I/O Management Receive command.....	464
7.3.1	I/O Management Receive Operations .....	464
7.3.2	Command Completion.....	465
7.4	I/O Management Send command .....	465
7.4.1	I/O Management Send Operations.....	466
7.4.2	Command Completion.....	467
7.5	Reservation Acquire command .....	467
7.5.1	Command Completion.....	468
7.6	Reservation Register command .....	468
7.6.1	Command Completion.....	470
7.7	Reservation Release command .....	470
7.7.1	Command Completion.....	471
7.8	Reservation Report command.....	471
7.8.1	Command Completion.....	474
<b>8</b>	<b>EXTENDED CAPABILITIES .....</b>	<b>475</b>
8.1	Common Extended Capabilities .....	475
8.1.1	Asymmetric Namespace Access Reporting .....	475
8.1.2	Asymmetric Namespace Access Reporting – Host Considerations (Informative) .....	481
8.1.3	Boot Partitions .....	483
8.1.4	Capacity Management .....	491
8.1.5	Command and Feature Lockdown .....	494
8.1.6	Controller Data Queue .....	495
8.1.7	Device Self-test Operations.....	496
8.1.8	Directives .....	498

8.1.9	Dispersed Namespaces .....	509
8.1.10	Flexible Data Placement .....	516
8.1.11	Host-Initiated Refresh Operation .....	521
8.1.12	Host Managed Live Migration .....	522
8.1.13	Key Per I/O .....	526
8.1.14	Management Addresses .....	527
8.1.15	Namespace Management .....	528
8.1.16	Namespace Write Protection .....	531
8.1.17	Power Management .....	534
8.1.18	Predictable Latency Mode .....	539
8.1.19	Reachability Reporting architecture .....	543
8.1.20	Read Recovery Level .....	546
8.1.21	Replay Protected Memory Block .....	547
8.1.22	Reservations .....	558
8.1.23	Rotational Media .....	566
8.1.24	Sanitize Operations .....	567
8.1.25	Submission Queue (SQ) Associations .....	581
8.1.26	Standard Vendor Specific Command Format .....	582
8.1.27	Telemetry .....	582
8.1.28	Universally Unique Identifiers (UUIDs) for Vendor Specific Information .....	586
8.2	Memory-Based Transport Extended Capabilities (PCIe) .....	589
8.2.1	Controller Memory Buffer .....	589
8.2.2	Doorbell Stride for Software Emulation .....	591
8.2.3	Host Memory Buffer .....	591
8.2.4	Persistent Memory Region .....	591
8.2.5	Power Loss Signaling .....	593
8.2.6	Virtualization Enhancements .....	600
8.3	Message-Based Transport Extended Capabilities (Fabrics) .....	604
8.3.1	Automated Discovery of NVMe-oF Discovery Controllers for IP Based Fabrics .....	604
8.3.2	Centralized Discovery for IP-based Fabrics .....	610
8.3.3	Exporting NVM Resources .....	635
8.3.4	NVMe over Fabrics Secure Channel and In-band Authentication .....	639
<b>9</b>	<b>ERROR REPORTING AND RECOVERY .....</b>	<b>668</b>
9.1	Command and Queue Error Handling .....	668
9.2	Media and Data Error Handling .....	668
9.3	Memory Error Handling .....	668
9.4	Internal Controller Error Handling .....	668
9.5	Controller Fatal Status Condition .....	668
9.6	Communication Loss Handling .....	669
9.6.1	Host Communication Loss with a Controller .....	669
9.6.2	End of Controller Processing of Outstanding Commands .....	670
9.6.3	Command Retry After Communication Loss .....	670
<b>ANNEX A.</b>	<b>SANITIZE OPERATION CONSIDERATIONS (INFORMATIVE) .....</b>	<b>674</b>
A.1	Overview .....	674
A.2	Hidden Storage (Overprovisioning) .....	674
A.3	Integrity checks and No-Deallocate After Sanitize .....	674
A.4	Bad Media and Vendor Specific NAND Use .....	675
<b>ANNEX B.</b>	<b>HOST CONSIDERATIONS (INFORMATIVE) .....</b>	<b>676</b>
B.1	Basic Steps when Building a Command .....	676
B.2	Creating an I/O Submission Queue .....	676
B.3	Executing a Fused Operation .....	677
B.4	Asynchronous Event Request Host Software Recommendations .....	679
B.5	Updating Controller Doorbell Properties using a Shadow Doorbell Buffer .....	680
B.5.1.	Shadow Doorbell Buffer Overview .....	680
B.5.2.	Example Algorithm for Controller Doorbell Property Updates .....	680
B.6	Examples of Incorrect Command Retry Handling After Communication Loss .....	680

B.6.1.	Write after Write .....	680
B.6.2.	Non-Idempotent Command .....	681
B.6.3.	Retried Command Does Not Succeed .....	682
B.6.4.	Retried Command Affects Another Host .....	683
<b>ANNEX C.</b>	<b>POWER MANAGEMENT AND CONSUMPTION (INFORMATIVE).....</b>	<b>684</b>

## Table of Figures

Figure 1: NVMe Family of Specifications .....	1
Figure 2: Decimal and Binary Units .....	3
Figure 3: Byte, Word, and Dword Relationships .....	5
Figure 4: Taxonomy of Transport Models .....	18
Figure 5: Types of NVMe Command Sets .....	19
Figure 6: Queue Pair Example, 1:1 Mapping .....	20
Figure 7: Queue Pair Example, <i>n</i> :1 Mapping .....	20
Figure 8: NVMe over Fabrics Layering .....	22
Figure 9: Command Capsule Format .....	23
Figure 10: Response Capsule Format .....	24
Figure 11: Simple NVM Storage Hierarchy with NVM Sets .....	25
Figure 12: Simple NVM Storage Hierarchy with One Reclaim Group .....	26
Figure 13: Simple NVM Storage Hierarchy with Multiple Reclaim Groups .....	27
Figure 14: Complex NVM Storage Hierarchy with NVM Sets .....	28
Figure 15: Complex NVM Storage Hierarchy with Multiple Reclaim Groups .....	29
Figure 16: Single-Namespace NVM Subsystem .....	30
Figure 17: Two-Namespace NVM Subsystem .....	31
Figure 18: Complex NVM Subsystem .....	32
Figure 19: NVMe Express Controller with Two Namespaces .....	33
Figure 20: NVM Subsystem with Two Controllers and One Port .....	33
Figure 21: NVM Subsystem with Two Controllers and Two Ports .....	34
Figure 22: PCI Express Device Supporting Single Root I/O Virtualization (SR-IOV) .....	35
Figure 23: Controller Types .....	37
Figure 24: NVM Subsystem with Three I/O Controllers .....	38
Figure 25: NVM Subsystem with One Administrative and Two I/O Controllers .....	40
Figure 26: NVM Subsystem with One Administrative Controller .....	40
Figure 27: Controller IDs FFF0h to FFFFh .....	42
Figure 28: Admin Command Support Requirements .....	43
Figure 29: Fabrics Command Support Requirements .....	44
Figure 30: Common I/O Command Support Requirements .....	45
Figure 31: Log Page Support Requirements .....	45
Figure 32: Feature Support Requirements .....	47
Figure 33: Property Definition .....	49
Figure 34: Memory-Based Property Definition .....	51
Figure 35: Message-Based Property Definition .....	51
Figure 36: Offset 0h: CAP – Controller Capabilities .....	52
Figure 37: Specification Version Descriptor .....	55
Figure 38: NVMe Express Base Specification Version Property Reset Values .....	55
Figure 39: Offset Ch: INTMS – Interrupt Mask Set .....	56
Figure 40: Offset 10h: INTMC – Interrupt Mask Clear .....	56
Figure 41: Offset 14h: CC – Controller Configuration .....	57
Figure 42: Offset 1Ch: CSTS – Controller Status .....	60
Figure 43: Offset 20h: NSSR – NVM Subsystem Reset .....	63
Figure 44: Offset 24h: AQA – Admin Queue Attributes .....	63
Figure 45: Offset 28h: ASQ – Admin Submission Queue Base Address .....	63
Figure 46: Offset 30h: ACQ – Admin Completion Queue Base Address .....	64
Figure 47: Offset 38h: CMBLOC – Controller Memory Buffer Location .....	64
Figure 48: Offset 3Ch: CMBSZ – Controller Memory Buffer Size .....	65
Figure 49: Offset 40h: BPINFO – Boot Partition Information .....	66
Figure 50: Offset 44h: BPRSEL – Boot Partition Read Select .....	66
Figure 51: Offset 48h: BPMBL – Boot Partition Memory Buffer Location .....	67
Figure 52: Offset 50h: CMBMSC – Controller Memory Buffer Memory Space Control .....	67
Figure 53: Offset 58h: CMBSTS – Controller Memory Buffer Status .....	68
Figure 54: Offset 5Ch: CMBEBS – Controller Memory Buffer Elasticity Buffer Size .....	68
Figure 55: Offset 60h: CMBSWTP – Controller Memory Buffer Sustained Write Throughput .....	68
Figure 56: Offset 64h: NSSD – NVM Subsystem Shutdown .....	69
Figure 57: Offset 68h: CRTO – Controller Ready Timeouts .....	70
Figure 58: Offset E00h: PMRCAP – Persistent Memory Region Capabilities .....	70
Figure 59: Offset E04h: PMRCTL – Persistent Memory Region Control .....	71
Figure 60: Offset E08h: PMRSTS – Persistent Memory Region Status .....	72
Figure 61: Offset E0Ch: PMREBS – Persistent Memory Region Elasticity Buffer Size .....	73

Figure 62: Offset E10h: PMRSWTP – Persistent Memory Region Sustained Write Throughput .....	73
Figure 63: Offset E14h: PMRMSCL – Persistent Memory Region Memory Space Control Lower .....	74
Figure 64: Offset E18h: PMRMSCU – Persistent Memory Region Memory Space Control Upper .....	74
Figure 65: NSID Types and Relationship to Namespace .....	75
Figure 66: NSID Types.....	76
Figure 67: NVM Sets and Associated Namespaces.....	78
Figure 68: NVM Set Aware Admin Commands .....	78
Figure 69: NVM Sets and Associated Namespaces.....	80
Figure 70: Flexible Data Placement Logical View of Non-Volatile Storage .....	82
Figure 71: Example 1 Domain Structure .....	83
Figure 72: Example 2 Domain Structure .....	84
Figure 73: Empty Queue Definition .....	87
Figure 74: Full Queue Definition.....	88
Figure 75: Command Capsule .....	89
Figure 76: Response Capsule.....	89
Figure 77: Data and SGL Locations within a Command Capsule .....	91
Figure 78: SGL Example Using Memory Transactions .....	91
Figure 79: SGL Example Using In Capsule Data Transfer.....	92
Figure 80: Round Robin Arbitration.....	100
Figure 81: Weighted Round Robin with Urgent Priority Class Arbitration.....	101
Figure 82: Queue Creation Flow .....	104
Figure 83: Discovery Controller Initialization process flow .....	106
Figure 84: Admin Commands Permitted to Return a Status Code of Admin Command Media Not Ready .....	107
Figure 85: Shutdown Processing Interactions.....	110
Figure 86: Simple NVM Subsystem .....	121
Figure 87: Vertically-Organized NVM Subsystem .....	122
Figure 88: Horizontally-Organized Dual NAND NVM Subsystem.....	123
Figure 89: Capacity Information Field Usage .....	124
Figure 90: Detecting Timeout Takes up to 2 * KATT.....	128
Figure 91: Command Dword 0 .....	133
Figure 92: Common Command Format.....	134
Figure 93: Common Command Format – Vendor Specific Commands (Optional).....	137
Figure 94: Fabrics Command – Submission Queue Entry Format.....	137
Figure 95: Fabrics Command – Command Dword 0 .....	138
Figure 96: Common Completion Queue Entry Layout – Admin and All I/O Command Sets .....	138
Figure 97: Completion Queue Entry: DW 2.....	138
Figure 98: Completion Queue Entry: DW 3.....	139
Figure 99: Fabrics Response – Completion Queue Entry Format.....	139
Figure 100: Completion Queue Entry: Status Field .....	140
Figure 101: Status Code – Status Code Type Values.....	140
Figure 102: Status Code – Generic Command Status Values .....	141
Figure 103: Status Code – Command Specific Status Values .....	144
Figure 104: Status Code – Command Specific Status Values, I/O Commands .....	146
Figure 105: Status Code – Command Specific Status Values, Fabrics Commands .....	147
Figure 106: Status Code – Media and Data Integrity Error Values .....	148
Figure 107: Status Code – Path Related Status Values.....	148
Figure 108: Phase Tag bit Transition Example .....	150
Figure 109: PRP Entry Layout.....	151
Figure 110: PRP Entry – Page Base Address and Offset .....	152
Figure 111: PRP List Layout for Physically Contiguous Memory Pages .....	152
Figure 112: PRP List Layout for Physically Non-Contiguous Memory Pages .....	152
Figure 113: SGL Segment .....	154
Figure 114: Generic SGL Descriptor Format.....	154
Figure 115: SGL Descriptor Type.....	154
Figure 116: SGL Descriptor Sub Type Values .....	154
Figure 117: SGL Data Block descriptor.....	155
Figure 118: SGL Bit Bucket descriptor .....	155
Figure 119: SGL Segment descriptor.....	156
Figure 120: SGL Last Segment descriptor .....	156
Figure 121: Keyed SGL Data Block descriptor.....	157
Figure 122: Transport SGL Data Block descriptor.....	157
Figure 123: SGL Read Example .....	158

Figure 124: Current Value after Reset with Scope of Entire NVM Subsystem .....	160
Figure 125: Current Value after Reset with Scope of Subset of the NVM Subsystem .....	160
Figure 126: PCI Vendor ID (VID) and PCI Subsystem Vendor ID (SSVID) .....	162
Figure 127: Serial Number (SN) and Model Number (MN) .....	162
Figure 128: IEEE OUI Identifier (IEEE) .....	162
Figure 129: IEEE Extended Unique Identifier (EUI64), MA-L Format.....	162
Figure 130: IEEE Extended Unique Identifier (EUI64), OUI Identifier .....	163
Figure 131: IEEE Extended Unique Identifier (EUI64), Ext. ID (cont).....	163
Figure 132: MA-L similarity to WWN .....	163
Figure 133: Namespace Globally Unique Identifier (NGUID) .....	163
Figure 134: Namespace Globally Unique Identifier (NGUID), OUI .....	163
Figure 135: Namespace Globally Unique Identifier (NGUID), Extension Identifier (continued).....	163
Figure 136: Namespace Globally Unique Identifier (NGUID), NGUID similarity to WWN .....	164
Figure 137: Controller List Format.....	164
Figure 138: Namespace List Format.....	164
Figure 139: NQN Construction for Older NVM Subsystems.....	166
Figure 140: UTF-8 Input Processing .....	168
Figure 141: Opcodes for Admin Commands .....	169
Figure 142: Sanitize Operations and Format NVM Command – Admin Commands Allowed .....	171
Figure 143: Abort – Command Dword 10.....	172
Figure 144: Abort Command – Completion Queue Entry Dword 0.....	173
Figure 145: Abort – Command Specific Status Values .....	173
Figure 146: Status Code – Command Specific Status Values .....	175
Figure 147: Asynchronous Event Request – Completion Queue Entry Dword 0 .....	175
Figure 148: Asynchronous Event Request – Completion Queue Entry Dword 1 .....	176
Figure 149: Asynchronous Event Information – Error Status .....	176
Figure 150: Asynchronous Event Information – SMART / Health Status .....	176
Figure 151: Asynchronous Event Information – Notice .....	177
Figure 152: Asynchronous Event Information – I/O Command Specific Status .....	179
Figure 153: Asynchronous Event Information – Immediate.....	180
Figure 154: Asynchronous Event Information – One Shot .....	180
Figure 155: Controller Data Queue Tail Pointer – Event Specific Parameter.....	180
Figure 156: Capacity Management – Command Dword 10 .....	181
Figure 157: Capacity Management – Command Dword 11 .....	181
Figure 158: Capacity Management – Command Dword 12 .....	182
Figure 159: Capacity Management – Command Specific Status Values .....	184
Figure 160: Capacity Management – Completion Queue Entry Dword 0.....	184
<b>Figure 161: Controller Data Queue – Command Dword 10 .....</b>	<b>185</b>
<b>Figure 162: Create Controller Data Queue – PRP Entry 1 .....</b>	<b>186</b>
Figure 163: Create Controller Data Queue – Command Dword 11.....	186
Figure 164: Create Queue – Command Dword 12.....	186
Figure 165: Create Controller Data Queue – Management Operation Specific .....	186
Figure 166: User Data Migration Queue – Create Queue Specific .....	187
Figure 167: Delete Controller Data Queue – Command Dword 11 .....	187
Figure 168: Controller Data Queue – Completion Queue Entry Dword 0.....	188
Figure 169: Controller Data Queue – Command Specific Status Values .....	188
Figure 170: Device Self-test Namespace Test Action .....	188
Figure 171: Device Self-test – Command Dword 10 .....	188
Figure 172: Device Self-test – Command Dword 15 .....	189
Figure 173: Device Self-test – Command Processing.....	189
Figure 174: Device Self-test – Command Specific Status Values.....	190
Figure 175: Directive Receive – Data Pointer .....	190
Figure 176: Directive Receive – Command Dword 10 .....	191
Figure 177: Directive Receive – Command Dword 11 .....	191
Figure 178: Directive Send – Data Pointer.....	191
Figure 179: Directive Send – Command Dword 10.....	191
Figure 180: Directive Send – Command Dword 11 .....	191
Figure 181: Firmware Commit – Command Dword 10.....	192
Figure 182: Firmware Commit – Completion Queue Entry Dword 0 .....	193
Figure 183: Firmware Commit – Command Specific Status Values.....	194
Figure 184: Firmware Image Download – Data Pointer .....	195
Figure 185: Firmware Image Download – Command Dword 10 .....	195



Figure 186: Firmware Image Download – Command Dword 11 .....	195
Figure 187: Firmware Image Download – Command Specific Status Values .....	195
Figure 188: Format NVM – Operation Scope .....	196
Figure 189: Format NVM – Command Dword 10 .....	197
Figure 190: Format NVM – Command Specific Status Values .....	198
Figure 191: Get Features – Data Pointer .....	199
Figure 192: Get Features – Command Dword 10 .....	199
Figure 193: Get Features – Command Dword 14 .....	199
Figure 194: Get Features – Feature Identifiers .....	199
Figure 195: Completion Queue Entry Dword 0 when Select is set to 11b .....	201
Figure 196: Get Log Page – Data Pointer .....	202
Figure 197: Get Log Page – Command Dword 10 .....	202
Figure 198: Get Log Page – Command Dword 11 .....	202
Figure 199: Get Log Page – Command Dword 12 .....	203
Figure 200: Get Log Page – Command Dword 13 .....	203
Figure 201: Get Log Page – Command Dword 14 .....	203
Figure 202: Get Log Page – Log Page Identifiers .....	204
Figure 203: Supported Log Pages Log Page .....	206
Figure 204: LID Supported and Effects Data Structure .....	206
Figure 205: Error Information Log Entry Data Structure .....	207
Figure 206: SMART / Health Information Log Page .....	209
Figure 207: Temperature Sensor Data Structure .....	212
Figure 208: Firmware Slot Information Log Page .....	213
Figure 209: Commands Supported and Effects Log Page .....	214
Figure 210: Commands Supported and Effects Data Structure .....	215
Figure 211: Device Self-test Log Page .....	217
Figure 212: Self-test Result Data Structure .....	218
Figure 213: Telemetry Host-Initiated Log Specific Parameter Field .....	219
Figure 214: Telemetry Host-Initiated Log Page .....	221
Figure 215: Telemetry Host-Initiated Log Page - LID Specific Parameter Field .....	222
Figure 216: Telemetry Controller-Initiated Log Page .....	223
Figure 217: Endurance Group Identifier - Log Specific Identifier .....	224
Figure 218: Endurance Group Information Log Page .....	225
Figure 219: NVM Set Identifier – Log Specific Identifier .....	226
Figure 220: Predictable Latency Per NVM Set Log Page .....	227
Figure 221: Predictable Latency Event Aggregate Log Page .....	228
Figure 222: Asymmetric Namespace Access Log Specific Parameter Field .....	229
Figure 223: Asymmetric Namespace Access Log Page .....	229
Figure 224: ANA Group Descriptor format .....	230
Figure 225: Persistent Event Log Specific Parameter Field .....	233
Figure 226: Persistent Event Log Page .....	234
Figure 227: Persistent Event Format .....	236
Figure 228: Persistent Event LID Specific Parameter Field .....	238
Figure 229: Persistent Event Log Event Types .....	238
Figure 230: SMART / Health Log Snapshot Event Data Format (Event Type 01h) .....	239
Figure 231: Firmware Commit Event Data Format (Event Type 02h) .....	239
Figure 232: Timestamp Change Event Format (Event Type 03h) .....	239
Figure 233: Power-on or Reset Event (Event Type 04h) .....	240
Figure 234: Controller Reset Information descriptor .....	240
Figure 235: NVM Subsystem Hardware Error Event Format (Event Type 05h) .....	241
Figure 236: NVM Subsystem Hardware Error Event Codes .....	241
Figure 237: Additional Hardware Error Information for Unexpected Power Loss Errors .....	243
Figure 238: Additional Hardware Error Information for correctable and uncorrectable PCIe errors .....	243
Figure 239: Additional Hardware Error Information for Controller Ready Timeout Exceeded errors .....	244
Figure 240: Change Namespace Event Data Format (Event Type 06h) .....	244
Figure 241: Format NVM Start Event Data Format (Event Type 07h) .....	246
Figure 242: Format NVM Completion Event Data Format (Event Type 08h) .....	246
Figure 243: Sanitize Start Event Data Format (Event Type 09h) .....	247
Figure 244: Sanitize Completion Event Data Format (Event Type 0Ah) .....	247
Figure 245: Feature Persistent Event Logging Requirements .....	248
Figure 246: Set Feature Event Data Format .....	249
Figure 247: Telemetry Log Create Event Data Format (Event Type 0Ch) .....	250

Figure 248: Thermal Excursion Event Data Format (Event Type 0Dh) .....	251
Figure 249: Vendor Specific Event Format (Event Type DEh) .....	252
Figure 250: Vendor Specific Event Descriptor Format .....	252
Figure 251: Vendor Specific Event Data Type Codes .....	252
Figure 252: Endurance Group Event Aggregate Log Page .....	253
Figure 253: Domain Identifier – Log Specific Identifier .....	254
Figure 254: Media Unit Status Log Page .....	254
Figure 255: Media Unit Status Descriptor .....	255
Figure 256: Supported Capacity Configuration List Log Page .....	256
Figure 257: Capacity Configuration Descriptor .....	257
Figure 258: Endurance Group Configuration Descriptor .....	257
Figure 259: Channel Configuration Descriptor .....	258
Figure 260: Media Unit Configuration Descriptor .....	259
Figure 261: Feature Identifiers Effects Log Page .....	259
Figure 262: FID Supported and Effects Data Structure .....	260
Figure 263: NVMe-MI Commands Supported and Effects Log Page .....	261
Figure 264: NVMe-MI Commands Supported and Effects Data Structure .....	262
Figure 265: Command and Feature Lockdown Log Specific Parameter Field .....	263
Figure 266: Command and Feature Lockdown Log Page .....	264
Figure 267: Boot Partition Log Specific Parameter Field .....	265
Figure 268: Boot Partition Log Page .....	265
Figure 269: Rotational Media Information Log Page .....	265
Figure 270: Dispersed Namespace Participating NVM Subsystems Log Page .....	266
Figure 271: Management Address List – Log Page .....	267
Figure 272: Management Address Descriptor .....	267
Figure 273: Reachability Groups Log Specific Parameter Field .....	268
Figure 274: Reachability Groups Log Page .....	268
Figure 275: Reachability Group Descriptor format .....	269
Figure 276: Reachability Associations Log Specific Parameter Field .....	270
Figure 277: Reachability Associations Log Page .....	270
Figure 278: Reachability Association Descriptor format .....	271
Figure 279: FDP Configurations Log Page .....	272
Figure 280: FDP Configuration Descriptor .....	272
Figure 281: Reclaim Unit Handle Descriptor .....	273
Figure 282: Placement Identifier Format without Reclaim Group Identifier .....	274
Figure 283: Placement Identifier Format with a non-zero RGIF .....	274
Figure 284: Reclaim Unit Handle Usage Log Page .....	275
Figure 285: Reclaim Unit Handle Usage Descriptor .....	275
Figure 286: FDP Statistics Log Page .....	276
Figure 287: Command Dword 10 – Log Specific Field .....	277
Figure 288: FDP Events Log Page .....	277
Figure 289: FDP Event .....	277
Figure 290: Reservation Notification Log Page .....	280
Figure 291: Sanitize Status Log Page .....	280
Figure 292: Discovery Log Page – LID Specific Parameter Field .....	284
Figure 293: Discovery Log Specific Parameter Field .....	286
Figure 294: Discovery Log Page Entry Data Structure .....	286
Figure 295: Extended Discovery Log Page Entry .....	289
Figure 296: Discovery Log Page .....	290
Figure 297: Host Discovery – LID Specific Parameter Field .....	291
Figure 298: Host Discovery Log Specific Parameter Field .....	291
Figure 299: Host Extended Discovery Log Page Entry .....	292
Figure 300: Host Discovery Log Page .....	292
Figure 301: Get Log Page – AVE Discovery Log Page .....	293
Figure 302: Get Log Page – AVE Discovery Log Page Entry .....	294
Figure 303: AVE Transport Record .....	294
Figure 304: Pull Model DDC Request Log Page .....	295
Figure 305: Get Log Page – Command Specific Status Values .....	295
Figure 306: Identify – Data Pointer .....	295
Figure 307: Identify – Command Dword 10 .....	296
Figure 308: Identify – Command Dword 11 .....	296
Figure 309: Identify – Command Dword 14 .....	296

Figure 310: Identify – CNS Values .....	297
Figure 311: Command Set Identifiers.....	298
Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent .....	299
Figure 313: Identify – Power State Descriptor Data Structure.....	328
Figure 314: Time Scale Values .....	331
Figure 315: Identify – Namespace Identification Descriptor .....	332
Figure 316: Command Dword 11 - CNS Specific Identifier .....	333
Figure 317: NVM Set List .....	333
Figure 318: NVM Set Attributes Entry .....	333
Figure 319: Identify – I/O Command Set Independent Identify Namespace Data Structure .....	335
Figure 320: UUID List.....	340
Figure 321: UUID List Entry .....	340
Figure 322: Command Dword 11 - CNS Specific Identifier .....	340
Figure 323: Domain List .....	340
Figure 324: Domain Attributes Entry .....	341
Figure 325: Command Dword 11 - CNS Specific Identifier .....	341
Figure 326: Endurance Group List .....	341
Figure 327: Identify I/O Command Set Data Structure.....	342
Figure 328: I/O Command Set Vector .....	343
Figure 329: Supported Controller State Formats Data Structure .....	343
Figure 330: Identify – Primary Controller Capabilities Structure.....	344
Figure 331: Secondary Controller List.....	346
Figure 332: Secondary Controller Entry .....	346
Figure 333: Underlying Namespace List Data Structure .....	346
Figure 334: Underlying Namespace Entry Data Structure .....	347
Figure 335: Ports List Data Structure .....	347
Figure 336: Underlying Fabrics Transport Entry Data Structure .....	347
Figure 337: Lockdown – Command Dword 10.....	348
Figure 338: Lockdown – Command Dword 14 .....	349
Figure 339: Lockdown – Command Specific Status Values.....	350
Figure 340: Migration Receive – Command Dword 10.....	350
Figure 341: Migration Receive – Command Dword 12.....	350
Figure 342: Migration Receive – Command Dword 13.....	350
Figure 343: Migration Receive – Command Dword 14.....	351
Figure 344: Migration Receive – Command Dword 15.....	351
Figure 345: Get Controller State Management Operation – Management Operation Specific.....	352
Figure 346: Get Controller State Management Operation – Command Dword 11 .....	352
Figure 347: Get Controller State Management Operation – Completion Queue Entry Dword 0 .....	352
<b>Figure 348: Migration Receive – Command Specific Status Values .....</b>	<b>353</b>
Figure 349: Migration Send – Command Dword 10 .....	353
Figure 350: Migration Send – Command Dword 14 .....	353
Figure 351: Suspend – Command Dword 11 .....	354
Figure 352: Resume – Command Dword 11 .....	355
<b>Figure 353: Set Controller State – Management Operation Specific Field .....</b>	<b>357</b>
Figure 354: Set Controller State – Command Dword 11 .....	357
Figure 355: Set Controller State – Command Dword 12 .....	357
Figure 356: Set Controller State – Command Dword 13 .....	357
Figure 357: Set Controller State – Command Dword 15 .....	357
Figure 358: Controller State Data Structure .....	358
Figure 359: NVMe Controller State Data Structure .....	359
Figure 360: I/O Submission Queue State Data Structure.....	359
Figure 361: I/O Completion Queue State Data Structure .....	360
Figure 362: Migration Send – Command Specific Status Values .....	361
Figure 363: Namespace Attachment – Data Pointer .....	362
Figure 364: Namespace Attachment – Command Dword 10 .....	362
Figure 365: Namespace Attachment – Command Specific Status Values.....	362
Figure 366: Namespace Management – Data Pointer .....	363
Figure 367: Namespace Management – Command Dword 10 .....	364
Figure 368: Namespace Management – Command Dword 11 .....	364
Figure 369: Namespace Management – Data Structure for Create .....	364
Figure 370: Namespace Management – Command Specific Status Values .....	364
Figure 371: Namespace Management – Completion Queue Entry Dword 0.....	365

Figure 372: Sanitize – Command Dword 10 .....	367
Figure 373: Sanitize – Command Dword 11 .....	368
Figure 374: Sanitize – Command Specific Status Values .....	368
Figure 375: Security Receive – Data Pointer .....	369
Figure 376: Security Receive – Command Dword 10 .....	369
Figure 377: Security Receive – Command Dword 11 .....	369
Figure 378: Security Protocol EAh – Security Protocol Specific Field Values .....	369
Figure 379: Security Send – Data Pointer .....	370
Figure 380: Security Send – Command Dword 10 .....	370
Figure 381: Security Send – Command Dword 11 .....	370
Figure 382: Set Features – Data Pointer .....	370
Figure 383: Set Features – Command Dword 10 .....	371
Figure 384: Set Features – Command Dword 14 .....	371
Figure 385: Set Features – Feature Identifiers .....	371
Figure 386: Arbitration & Command Processing – Command Dword 11 .....	374
Figure 387: Power Management – Command Dword 11 .....	374
Figure 388: Power Management – Completion Queue Entry Dword 0 .....	374
Figure 389: Temperature Threshold – Command Dword 11 .....	376
Figure 390: Volatile Write Cache – Command Dword 11 .....	377
Figure 391: Asynchronous Event Configuration – Command Dword 11 .....	377
Figure 392: Autonomous Power State Transition – Command Dword 11 .....	379
Figure 393: Autonomous Power State Transition – Data Structure Entry .....	379
Figure 394: Interactions between APSTE and NOPPME .....	379
Figure 395: Timestamp – Data Structure for Set Features .....	380
Figure 396: Timestamp – Data Structure for Get Features .....	380
Figure 397: Keep Alive Timer – Command Dword 11 .....	381
Figure 398: HCTM – Command Dword 11 .....	382
Figure 399: Non-Operational Power State Config – Command Dword 11 .....	382
Figure 400: Read Recovery Level Config – Command Dword 11 .....	383
Figure 401: Read Recovery Level Config – Command Dword 12 .....	383
Figure 402: Predictable Latency Mode Config – Command Dword 11 .....	384
Figure 403: Predictable Latency Mode Config – Command Dword 12 .....	384
Figure 404: Predictable Latency Mode – Deterministic Threshold Configuration Data Structure .....	384
Figure 405: Predictable Latency Mode Window – Command Dword 11 .....	385
Figure 406: Predictable Latency Mode Window – Command Dword 12 .....	385
Figure 407: Host Behavior Support – Data Structure .....	386
Figure 408: Sanitize Config – Command Dword 11 .....	387
Figure 409: Asynchronous Event Configuration – Command Dword 11 .....	388
Figure 410: I/O Command Set Profile – Command Dword 11 .....	388
Figure 411: I/O Command Set Profile – Completion Queue Entry Dword 0 .....	389
Figure 412: Spinup Control – Command Dword 11 .....	389
Figure 413: Completion Queue Entry Dword 0 .....	389
Figure 414: Power Loss Signaling Config – Command Dword 11 .....	390
Figure 415: Flexible Data Placement – Command Dword 11 .....	390
Figure 416: Flexible Data Placement – Command Dword 12 .....	390
Figure 417: FDP Events – Completion Queue Entry Dword 0 .....	391
Figure 418: FDP Events – Command Dword 11 .....	391
Figure 419: Flexible Data Placement – Command Dword 12 .....	391
Figure 420: FDP Events – Set Feature Data Structure .....	392
Figure 421: FDP Events – Get Feature Data Structure .....	392
Figure 422: Supported FDP Event Descriptor .....	392
Figure 423: Namespace Admin Label – Data Structure .....	393
Figure 424: Controller Data Queue – Command Dword 11 .....	393
Figure 425: Controller Data Queue – Command Dword 12 .....	394
Figure 426: Controller Data Queue – Command Dword 13 .....	394
Figure 427: Controller Data Queue – Data Structure .....	394
Figure 428: Set Features – Command Specific Status Values .....	394
Figure 429: Host Management Agent Address .....	395
Figure 430: Get Features – Command Dword 11 .....	395
Figure 431: Set Features – Command Dword 11 .....	396
Figure 432: Host Metadata Data Structure .....	398
Figure 433: Metadata Element Descriptor .....	398

Figure 434: Controller Metadata Element Types.....	398
Figure 435: Namespace Metadata Element Types .....	400
Figure 436: Software Progress Marker – Command Dword 11.....	400
Figure 437: Host Identifier – Command Dword 11 .....	401
Figure 438: Host Identifier – Data Structure Entry.....	401
Figure 439: Reservation Notification Configuration – Command Dword 11 .....	402
Figure 440: Reservation Persistence Configuration – Command Dword 11 .....	403
Figure 441: Write Protection – Command Dword 11.....	403
Figure 442: Boot Partition Write Protection Config - Command Dword 11.....	404
Figure 443: Number of Queues – Command Dword 11 .....	405
Figure 444: Number of Queues – Completion Queue Entry Dword 0 .....	406
Figure 445: Interrupt Coalescing – Command Dword 11 .....	406
Figure 446: Interrupt Vector Configuration – Command Dword 11 .....	407
Figure 447: Host Memory Buffer – Command Dword 11 .....	408
Figure 448: Host Memory Buffer – Command Dword 12 .....	409
Figure 449: Host Memory Buffer– Command Dword 13 .....	409
Figure 450: Host Memory Buffer – Command Dword 14 .....	409
Figure 451: Host Memory Buffer – Command Dword 15 .....	409
Figure 452: Host Memory Buffer – Host Memory Descriptor List .....	409
Figure 453: Host Memory Buffer – Host Memory Buffer Descriptor Entry.....	410
Figure 454: Host Memory Buffer – Completion Queue Entry Dword 0.....	410
Figure 455: Host Memory Buffer – Attributes Data Structure .....	410
Figure 456: Set Features – Command Specific Status Values .....	411
Figure 457: Track Receive – Command Dword 10 .....	411
Figure 458: Track Receive – Command Dword 12 .....	411
Figure 459: Tracked Memory Changes – Command Dword 11 .....	412
Figure 460: Tracked Memory Change Data Structure.....	412
Figure 461: Tracked Memory Changed Descriptor .....	413
<b>Figure 462: Track Receive – Command Specific Status Values .....</b>	<b>414</b>
Figure 463: Track Send – Command Dword 10.....	414
Figure 464: Log User Data Changes – Management Operation Specific Field.....	415
Figure 465: Log User Data Changes – Command Dword 11.....	415
Figure 466: Track Memory Changes – Management Operation Specific Field.....	416
Figure 467: Track Memory Changes – Command Dword 11 .....	416
Figure 468: Track Memory Changes Data Structure.....	417
Figure 469: Memory Range Tracking Descriptor.....	417
Figure 470: Track Memory Changes – Completion Queue Entry Dword 0 .....	418
Figure 471: Track Memory Changes – Completion Queue Entry Dword 1 .....	418
Figure 472: Track Send – Command Specific Status Values.....	418
Figure 473: Create I/O Completion Queue – PRP Entry 1 .....	419
Figure 474: Create I/O Completion Queue – Command Dword 10.....	419
Figure 475: Create I/O Completion Queue – Command Dword 11 .....	419
Figure 476: Create I/O Completion Queue – Command Specific Status Values.....	420
Figure 477: Create I/O Submission Queue – PRP Entry 1.....	420
Figure 478: Create I/O Submission Queue – Command Dword 10.....	420
Figure 479: Create I/O Submission Queue – Command Dword 11.....	421
Figure 480: Create I/O Submission Queue – Command Dword 12.....	421
Figure 481: Create I/O Submission Queue – Command Specific Status Values .....	422
Figure 482: Delete I/O Completion Queue – Command Dword 10 .....	422
Figure 483: Delete I/O Completion Queue – Command Specific Status Values .....	422
Figure 484: Delete I/O Submission Queue – Command Dword 10.....	423
Figure 485: Delete I/O Submission Queue – Command Specific Status Values.....	423
Figure 486: Doorbell Buffer Config – Shadow Doorbell and EventIdx.....	423
Figure 487: Doorbell Buffer Config – PRP Entry 1 .....	424
Figure 488: Doorbell Buffer Config – PRP Entry 2 .....	424
Figure 489: Virtualization Management – Command Dword 10.....	424
Figure 490: Virtualization Management – Command Dword 11 .....	425
Figure 491: Virtualization Management – Command Specific Status Values.....	425
Figure 492: Virtualization Management – Completion Queue Entry Dword 0 .....	426
Figure 493: Create Exported NVM Subsystem – Data Pointer .....	427
Figure 494: Create Exported NVM Subsystem – Command Dword 10.....	427
Figure 495: Discovery Information Management – Data Pointer.....	429

Figure 496: Discovery Information Management – Command Dword 10.....	429
Figure 497: Discovery Information Management – Data.....	429
Figure 498: Extended Discovery Information Entry.....	432
Figure 499: Extended Attribute.....	433
Figure 500: Extended Discovery Information Entry – Applicable Fields.....	434
Figure 501: Discovery Information Management – Command Specific Status Values.....	435
Figure 502: Fabric Zoning Lookup (FZL) – Data Pointer.....	435
Figure 503: Fabric Zoning Lookup (FZL) – Command Specific Status Values.....	435
Figure 504: Fabric Zoning Lookup (FZL) – Completion Queue Entry Dword 0.....	435
Figure 505: Fabric Zoning Receive (FZR) – Data Pointer.....	436
Figure 506: Fabric Zoning Receive (FZR) – Command Dword 10.....	436
Figure 507: Fabric Zoning Receive (FZR) – Command Dword 11.....	436
Figure 508: Fabric Zoning Receive (FZR) – Command Dword 12.....	436
Figure 509: Fabric Zoning Receive (FZR) – Command Specific Status Values.....	436
Figure 510: Fabric Zoning Receive (FZR) – Completion Queue Entry Dword 0.....	437
Figure 511: Fabric Zoning Send (FZS) – Data Pointer.....	437
Figure 512: Fabric Zoning Send (FZS) – Command Dword 10.....	437
Figure 513: Fabric Zoning Send (FZS) – Command Dword 11.....	437
Figure 514: Fabric Zoning Send (FZS) – Command Dword 12.....	437
Figure 515: Fabric Zoning Send (FZS) – Command Specific Status Values.....	438
Figure 516: Manage Exported Namespace – Data Pointer.....	438
Figure 517: Manage Exported Namespace – Command Dword 10.....	438
Figure 518: Associate Namespace Data Structure.....	439
Figure 519: Disassociate Namespace Data Structure.....	440
Figure 520: Manage Exported NVM Subsystem – Data Pointer.....	440
Figure 521: Manage Exported NVM Subsystem– Command Dword 10.....	441
Figure 522: Management Operation Specific: Change Access Mode operation.....	441
Figure 523: Subsystem Management Data Structure.....	443
Figure 524: Host Entry Data Structure.....	443
Figure 525: Exported NVM Subsystem Entry Data Structure.....	443
Figure 526: Modify Host Access – Command Operation Specific Status Values.....	444
Figure 527: Manage Exported Port – Data Pointer.....	444
Figure 528: Manage Exported Port Data Structure– Command Dword 10.....	445
Figure 529: Management Operation Specific: Create operation.....	445
Figure 530: Create Data Structure.....	445
Figure 531: Create Completion Queue Entry Dword 0 Data Structure.....	446
Figure 532: Delete Data Structure.....	446
Figure 533: Send Discovery Log Page (SDLP) – Data Pointer.....	447
Figure 534: Send Discovery Log Page (SDLP) – Command Dword 10.....	447
Figure 535: Send Discovery Log Page (SDLP) – Command Dword 11.....	447
Figure 536: Send Discovery Log Page (SDLP) – Command Dword 12.....	447
Figure 537: Send Discovery Log Page (SDLP) – Command Dword 13.....	447
Figure 538: Send Discovery Log Page (SDLP) – Allowed Log Page Identifiers.....	448
Figure 539: Send Discovery Log Page (SDLP) – Completion Queue Entry Dword 0.....	448
Figure 540: Fabrics Command Type.....	449
Figure 541: Authentication Receive Command – Submission Queue Entry.....	449
Figure 542: Authentication Receive Response.....	450
Figure 543: Authentication Send Command – Submission Queue Entry.....	450
Figure 544: Authentication Send Response.....	451
Figure 545: Connect Command – Submission Queue Entry.....	453
Figure 546: Connect Command – Data.....	455
Figure 547: Connect Response.....	455
Figure 548: Connect Response – Dword 0 Value Based on Status Code.....	456
Figure 549: Disconnect Command – Submission Queue Entry.....	457
Figure 550: Disconnect Response.....	457
Figure 551: Property Get Command – Submission Queue Entry.....	457
Figure 552: Property Get Response.....	458
Figure 553: Property Set Command – Submission Queue Entry.....	458
Figure 554: Property Set Response.....	458
Figure 555: Opcodes for I/O Commands.....	460
Figure 556: Cancel – Command Dword 10.....	461
Figure 557: Cancel – Command Dword 11.....	462

Figure 558: Cancel – Command Specific Status Values .....	462
Figure 559: Cancel – Completion Queue Entry Dword 0 .....	463
Figure 560: I/O Management Receive – Data Pointer .....	464
Figure 561: I/O Management Receive – Command Dword 10 .....	464
Figure 562: I/O Management Receive – Command Dword 11 .....	464
Figure 563: Reclaim Unit Handle Status .....	465
Figure 564: I/O Management Send – Data Pointer .....	465
Figure 565: I/O Management Send – Command Dword 10 .....	465
Figure 566: Management Operation Specific – Reclaim Unit Handle Update Operation .....	466
Figure 567: Reclaim Unit Handle Update – Data Buffer .....	467
Figure 568: Reservation Acquire – Data Pointer .....	467
Figure 569: Reservation Acquire – Command Dword 10 .....	467
Figure 570: Reservation Acquire Data Structure .....	468
Figure 571: Reservation Type Encoding .....	468
Figure 572: Reservation Register – Data Pointer .....	469
Figure 573: Reservation Register – Command Dword 10 .....	469
Figure 574: Reservation Register Data Structure .....	470
Figure 575: Reservation Release – Data Pointer .....	470
Figure 576: Reservation Release – Command Dword 10 .....	470
Figure 577: Reservation Release Data Structure .....	471
Figure 578: Reservation Report – Data Pointer .....	471
Figure 579: Reservation Report – Command Dword 10 .....	472
Figure 580: Reservation Report – Command Dword 11 .....	472
Figure 581: Reservation Status Data Structure .....	472
Figure 582: Reservation Status Extended Data Structure .....	473
Figure 583: Registered Controller Data Structure .....	473
Figure 584: Registered Controller Extended Data Structure .....	474
Figure 585: Namespace B and C optimized through Controller 2 .....	476
Figure 586: Namespace B optimized through Controller 1 .....	477
Figure 587: Multiple Namespace groups .....	478
Figure 588: ANA effects on Command Processing .....	480
Figure 589: Boot Partition Overview .....	484
Figure 590: Set Features Boot Partition Write Protection State Machine Model .....	486
Figure 591: Set Features Boot Partition Write Protection State Definitions .....	487
Figure 592: RPMB Boot Partition Write Protection State Machine Model .....	488
Figure 593: RPMB Boot Partition Write Protection State Definitions .....	488
Figure 594: Boot Partition Write Protection State Machine Model .....	490
Figure 595: Example Device Self-test Operation (Informative) .....	497
Figure 596: Format NVM command Aborting a Device Self-Test Operation .....	498
Figure 597: Directive Types .....	499
Figure 598: Directive Specific Field Interpretation .....	499
Figure 599: Identify Directive – Directive Operations .....	500
Figure 600: Identify Directive – Return Parameters Data Structure .....	500
Figure 601: Enable Directive – Command Dword 12 .....	502
Figure 602: Enable Directive – Command Specific Status Values .....	502
Figure 603: Directive Streams – Stream Alignment and Granularity .....	502
Figure 604: Streams – Directive Operations .....	503
Figure 605: Example Multi-Stream and NSSC .....	504
Figure 606: Streams Directive – Command Specific Status Values .....	505
Figure 607: Streams Directive – Return Parameters Data Structure .....	506
Figure 608: Streams Directive – Get Status Data Structure .....	508
Figure 609: Allocate Resources – Command Dword 12 .....	508
Figure 610: Allocate Resources – Completion Queue Entry Dword 0 .....	508
Figure 611: Online Data Migration .....	510
Figure 612: Data Replication Example 1 .....	511
Figure 613: Data Replication Example 2 .....	511
Figure 614: Data Replication Example 3 .....	512
Figure 615: Dispersed Namespaces Command Restrictions - Prohibited Admin Commands .....	515
Figure 616: Flexible Data Placement Model .....	518
Figure 617: Initially Isolated Reclaim Unit Handles .....	519
Figure 618: Persistently Isolated Reclaim Unit Handle .....	519
Figure 619: Host Managed Live Migration Host and Controller Naming .....	522

Figure 620: CETYPE Definition.....	527
Figure 621: Namespace Write Protection State Definitions .....	531
Figure 622: Namespace Write Protection State Machine Model .....	532
Figure 623: Commands Allowed when Specifying a Write Protected NSID .....	533
Figure 624: Dynamic Power Management .....	534
Figure 625: Example Power State Descriptor Table .....	534
Figure 626: Workload Hints.....	537
Figure 627: HCTM Example.....	539
Figure 628: Deterministic and Non-Deterministic Windows .....	540
Figure 629: DTWIN Attributes and Estimates .....	541
Figure 630: Typical and Reliable Estimate Example .....	542
Figure 631: Reachability Associations Example.....	545
Figure 632: Read Recovery Level Overview .....	547
Figure 633: RPMB Device Configuration Block Data Structure.....	548
Figure 634: RPMB Request and Response Message Types .....	549
Figure 635: RPMB Operation Result.....	550
Figure 636: RPMB Contents .....	550
Figure 637: RPMB Data Frame.....	551
Figure 638: RPMB – Authentication Key Data Flow.....	552
Figure 639: RPMB – Read Write Counter Value Flow .....	553
Figure 640: RPMB – Authenticated Data Write Flow .....	555
Figure 641: RPMB – Authenticated Data Read Flow .....	556
Figure 642: RPMB – Authenticated Device Configuration Block Write Flow .....	557
Figure 643: RPMB – Authenticated Device Configuration Block Read Flow .....	558
Figure 644: Example Multi-Host System .....	559
Figure 645: Command Behavior in the Presence of a Reservation.....	561
Figure 646: Command Behavior in the Presence of a Reservation.....	561
Figure 647: Sanitize Operations – Overwrite Mechanism .....	570
Figure 648: Sanitize Operation State Machine.....	573
Figure 649: Idle State Transition Conditions .....	574
Figure 650: Restricted Processing State Transition Conditions .....	575
Figure 651: Restricted Failure State Transition Conditions .....	576
Figure 652: Unrestricted Processing State Transition Conditions .....	577
Figure 653: Unrestricted Failure State Transition Conditions.....	578
Figure 654: Media Verification State Transition Conditions.....	579
Figure 655: Post-Verification Deallocation state Transition Conditions .....	579
Figure 656: Telemetry Log Example – All Data Areas Populated .....	585
Figure 657: Telemetry Log Example – Data Area 2 Populated .....	586
Figure 658: UUID Index Field.....	587
Figure 659: Power Loss Signaling Variables.....	593
Figure 660: Power Loss Signaling Processing State Machine .....	595
Figure 661: PLS States.....	596
Figure 662: PLS Not Ready State Transition Conditions .....	596
Figure 663: PLS Ready State Transition Conditions .....	597
Figure 664: FQ Processing State Transition Conditions .....	597
Figure 665: FQ Complete State Transition Conditions.....	597
Figure 666: EPF Processing Port Disabled State Transition Conditions .....	598
Figure 667: EPF Complete Port Disabled State Transition Conditions .....	598
Figure 668: EPF Processing Port Enabled State Transition Conditions.....	598
Figure 669: EPF Complete Port Enabled State Transition Conditions .....	598
Figure 670: Controller Resource Allocation.....	601
Figure 671: Configuration A - Two IP Interfaces .....	605
Figure 672: Configuration B – Multiple IP Interfaces without a CDC .....	605
Figure 673: Configuration C – Multiple IP interfaces with a CDC .....	605
Figure 674: mDNS Protocol Field.....	606
Figure 675: mDNS Subtype .....	606
Figure 676: mDNS Domain .....	606
Figure 677: Registration and Discovery Example .....	610
Figure 678: Kickstart Discovery Request and Response Sequence .....	615
Figure 679: Zoning DB Abstract Representation.....	615
Figure 680: ZoneDBActive Abstract Representation.....	616
Figure 681: ZoneDBConfig Abstract Representation .....	616



Figure 682: ZoneGroup Detailed Representation.....	616
Figure 683: Zone Detailed Representation .....	617
Figure 684: Zone Member Types .....	618
Figure 685: {NQN, Role} Zone Member Format.....	618
Figure 686: {(NQN+IP+Protocol), Role} Zone Member Format.....	619
Figure 687: {(NQN+IP+Protocol+IP Protocol Port), Role} Zone Member Format.....	619
Figure 688: {(NQN+IP+Protocol+IP Protocol Port+PortID), Role} Zone Member Format .....	620
Figure 689: {(NQN+PortID), Role} Zone Member Format .....	621
Figure 690: {(NQN+Protocol+PortID+ADRFAM), Role} Zone Member Format.....	621
Figure 691: {(IP+Protocol), Role} Zone Member Format.....	622
Figure 692: {(IP+Protocol+IP Protocol Port), Role} Zone Member Format.....	623
Figure 693: {ZoneAlias} Zone Member Format .....	623
Figure 694: Zone Property Types.....	623
Figure 695: Fabric Enforced Zone Property Format.....	624
Figure 696: ZoneAlias Detailed Representation.....	624
Figure 697: GAZ for Push Model DDC.....	625
Figure 698: FZL Data for GAZ.....	625
Figure 699: FZR Data for GAZ.....	626
Figure 700: AAZ for Push Model DDC .....	627
Figure 701: FZL Data for AAZ.....	627
Figure 702: FZS Data for AAZ.....	627
Figure 703: RAZ for Push Model DDC .....	628
Figure 704: FZL Data for RAZ.....	628
Figure 705: GAZ for Pull Model DDC .....	629
Figure 706: GAZ Operation Specific Parameters for Pull Model DDC Request Log Page.....	629
Figure 707: FZS Data for Pull Model GAZ .....	629
Figure 708: AAZ for Pull Model DDC .....	631
Figure 709: AAZ Operation Specific Parameters for Pull Model DDC Request Log Page .....	631
Figure 710: FZR Data for Pull Model AAZ.....	632
Figure 711: FZS Data for Pull Model AAZ.....	632
Figure 712: RAZ for Pull Model DDC .....	633
Figure 713: RAZ Operation Specific Parameters for Pull Model DDC Request Log Page .....	633
Figure 714: FZS Data for Pull Model RAZ.....	633
Figure 715: Pull Model DDC Zoning Operations Status Values .....	634
Figure 716: Log Page Request Operation.....	634
Figure 717: Log Page Request Operation Specific Parameters for Pull Model DDC Request Log Page.....	635
Figure 718: Example reference for retrieving Underlying Namespaces and Underlying Ports.....	636
Figure 719: Example reference for retrieving Underlying Namespaces and Underlying Ports.....	637
Figure 720: Example steps for retrieving underlying resources and configuring an Exported NVM Subsystem .....	638
Figure 721: Example steps for restricting access to an Exported NVM Subsystem to specific hosts.....	638
Figure 722: Example steps for revoking specific hosts access to an Exported NVM Subsystem .....	639
Figure 723: Example steps for removing an Exported NVM Subsystem.....	639
Figure 724: Example steps for clearing all Exported NVM Subsystems.....	639
Figure 725: Example of TLS secure channel establishment .....	640
Figure 726: Unique Discovery Service NQN retrieval for bidirectional authentication .....	642
Figure 727: Example of authentication transaction for NVMe/TCP .....	643
Figure 728: Mapping of authentication messages to the Authentication Send command .....	643
Figure 729: Mapping of authentication messages to the Authentication Receive command.....	644
Figure 730: Example of TLS secure channel concatenation with NVMe/TCP.....	645
Figure 731: AUTH_Negotiate message format .....	646
Figure 732: Secure channel protocol identifiers .....	646
Figure 733: Authentication protocol identifiers .....	647
Figure 734: AUTH_Failure1 and AUTH_Failure2 message format .....	647
Figure 735: AUTH_Failure reason codes.....	647
Figure 736: AUTH_Failure reason code explanations.....	648
Figure 737: Example of DH-HMAC-CHAP authentication transaction .....	649
Figure 738: Mathematical notations for DH-HMAC-CHAP.....	649
Figure 739: Authentication protocol descriptor for DH-HMAC-CHAP.....	651
Figure 740: DH-HMAC-CHAP hash function identifiers .....	651
Figure 741: DH-HMAC-CHAP Diffie-Hellman group identifiers .....	651
Figure 742: DH-HMAC-CHAP_Challenge message format .....	652
Figure 743: DH-HMAC-CHAP_Reply message format .....	653

Figure 744: DH-HMAC-CHAP_Success1 message format .....	655
Figure 745: DH-HMAC-CHAP_Success2 message format .....	657
Figure 746: DH-HMAC-CHAP Configurations .....	659
Figure 747: NVM Subsystem AUTHREQ Settings for DH-HMAC-CHAP .....	659
Figure 748: DH-HMAC-CHAP Host Behavior .....	660
Figure 749: DH-HMAC-CHAP NVM Subsystem Behavior .....	660
Figure 750: DH-HMAC-CHAP with TLS Concatenation Configurations .....	660
Figure 751: NVM Subsystem AUTHREQ Settings for DH-HMAC-CHAP with TLS Concatenation .....	661
Figure 752: DH-HMAC-CHAP with TLS Concatenation Host Behavior .....	661
Figure 753: DH-HMAC-CHAP with TLS Concatenation NVM Subsystem Behavior .....	662
Figure 754: Example of DH-HMAC-CHAP authentication transaction with AVE .....	664
Figure 755: DH-HMAC-CHAP_Access-Request PDU format .....	665
Figure 756: DH-HMAC-CHAP_Access-Result PDU format .....	667
Figure 757: PRP List Describing I/O Submission Queue .....	677
Figure 758: PRP List Describing Data to Compare .....	678
Figure 759: Write after Write .....	681
Figure 760: Non-Idempotent Command .....	682
Figure 761: Retried Command Does Not Succeed .....	682
Figure 762: Retried Command Affects Another Host .....	683

# 1 Introduction

## 1.1 Overview

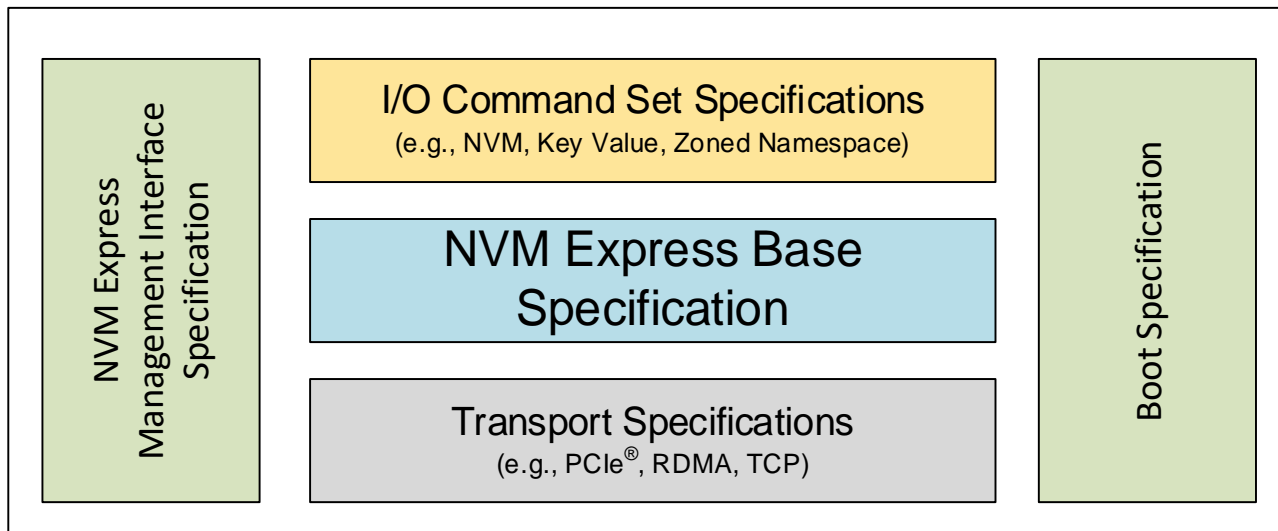
The NVM Express® (NVMe®) interface allows host software to communicate with a non-volatile memory subsystem (NVM subsystem). This interface is optimized for all storage solutions, attached using a variety of transports including PCI Express®, Ethernet, InfiniBand™, and Fibre Channel. The mapping of extensions 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).

For an overview of changes from revision 2.0 to revision 2.1, refer to <https://nvmexpress.org/changes> for a document that describes the new features, including mandatory requirements for a controller to comply with revision 2.1.

### 1.1.1 NVMe Express® Specification Family

Figure 1 shows the relationship of the NVM Express specifications to each other within the NVMe® family of specifications.

**Figure 1: NVMe Family of Specifications**



The NVMe Express specification family structure shown in Figure 1 is intended to show the applicability of NVMe Express specifications to each other, not a hierarchy, protocol stack, or system architecture.

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

The NVMe Express Management Interface (NVMe-MI) Specification defines an optional management interface for all NVMe Express Subsystems.

NVMe Express I/O Command Set specifications define data structures, features, log pages, commands, and status values that extend the NVMe Express Base Specification.

NVMe Express Transport specifications define the binding of the NVMe protocol including controller properties to a specific transport.

The NVMe Express Boot Specification defines constructs and guidelines for booting from NVMe Express interfaces.

## 1.2 Scope

This specification defines a set of properties and commands that comprise the interface required for communication with a controller in an NVM subsystem. These properties are to be implemented by an instance of a controller using a specific NVMe Transport. This specification also defines common aspects of the NVMe I/O Command Sets that may be supported by a controller.

There are three types of controllers with different capabilities (refer to section 3.1.3):

- a) I/O controllers;
- b) Discovery controllers; and
- c) Administrative controllers.

In this document the generic term controller is often used instead of enumerating specific controller types when applicable controller types may be determined from the context.

## 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 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 specification defines requirements and behaviors that are implementation agnostic. The implementation of these requirements and behaviors are outside the scope of this specification. For example, an NVM subsystem that follows this specification may be implemented by an SSD that attaches directly to a fabric, a device that translates between a fabric and a PCIe NVMe SSD, or software running on a general-purpose server.

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

This specification does not contain any information on caching algorithms or techniques.

The implementation or use of other published specifications referred to in this specification, even if required for compliance with the specification, are outside the scope of this specification (e.g., PCI, PCI Express, and PCI-X). This includes published specifications for fabrics and other technologies referred to by this document or any NVMe Transport binding specification.

## 1.4 Conventions

### 1.4.1 Keywords

Several keywords are used to differentiate between different levels of requirements.

#### 1.4.1.1 mandatory

A keyword indicating items to be implemented as defined by this specification.

#### 1.4.1.2 may

A keyword that indicates flexibility of choice with no implied preference.

#### 1.4.1.3 obsolete

A keyword indicating functionality that was defined in a previous version of the NVM Express specification and that has been removed from this specification.

**1.4.1.4 optional**

A keyword that describes features that are not required by this specification. However, if any optional feature defined by the specification is implemented, the feature shall be implemented in the way defined by the specification.

**1.4.1.5 R**

“R” is used as an abbreviation for “reserved” when the figure or table does not provide sufficient space for the full word “reserved”.

**1.4.1.6 reserved**

A keyword referring to bits, bytes, words, fields, and opcode values that are set-aside for future standardization. Their use and interpretation may be specified by future extensions to this or other specifications. A reserved bit, byte, word, field, property, or register shall be cleared to 0h, or in accordance with a future extension to this specification. The recipient of a command or a register write is not required to check reserved bits, bytes, words, or fields. Receipt of reserved coded values in defined fields in commands shall be reported as an error. Writing a reserved coded value into a controller property field produces undefined results.

**1.4.1.7 shall**

A keyword indicating a mandatory requirement. Designers are required to implement all such mandatory requirements to ensure interoperability with other products that conform to the specification.

**1.4.1.8 should**

A keyword indicating flexibility of choice with a strongly preferred alternative. Equivalent to the phrase “it is recommended”.

**1.4.2 Numerical Descriptions**

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

Size values are shown in binary units or decimal units. The symbols used to represent these values are as shown in Figure 2.

**Figure 2: Decimal and Binary Units**

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

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

Some parameters are defined as an ASCII string. ASCII strings shall contain only code values (i.e., byte values or octet values) 20h through 7Eh. For the string “Copyright”, the character “C” is the first byte, the character “o” is the second byte, etc. ASCII strings are left justified. If padding is necessary, then the string

shall be padded with spaces (i.e., ASCII character 20h) to the right unless the string is specified as null-terminated.

Some parameters are defined as a UTF-8 string. UTF-8 strings shall contain only byte values (i.e., octet values) 20h through 7Eh, 80h through BFh, and C2h through F4h (refer to sections 1 to 3 of RFC 3629). For the string “Copyright”, the character “C” is the first byte, the character “o” is the second byte, etc. UTF-8 strings are left justified. If padding is necessary, then the string shall be padded with spaces (i.e., ASCII character 20h, Unicode character U+0020) to the right unless the string is specified as null-terminated.

If padding is necessary for a field that contains a null-terminated string then the field should be padded with nulls (i.e., ASCII character 00h, Unicode character U+0000) to the right of the string.

A hexadecimal ASCII string is an ASCII string that uses a subset of the code values: “0” to “9”, “A” to “F” uppercase, and “a” to “f” lowercase.

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

Binary (i.e., base 2) numbers are written with a lower case “b” suffix (e.g., 1001b, 10b). Binary numbers larger than four digits are written with an underscore character dividing each group of four digits (e.g., 1000\_0101\_0010b).

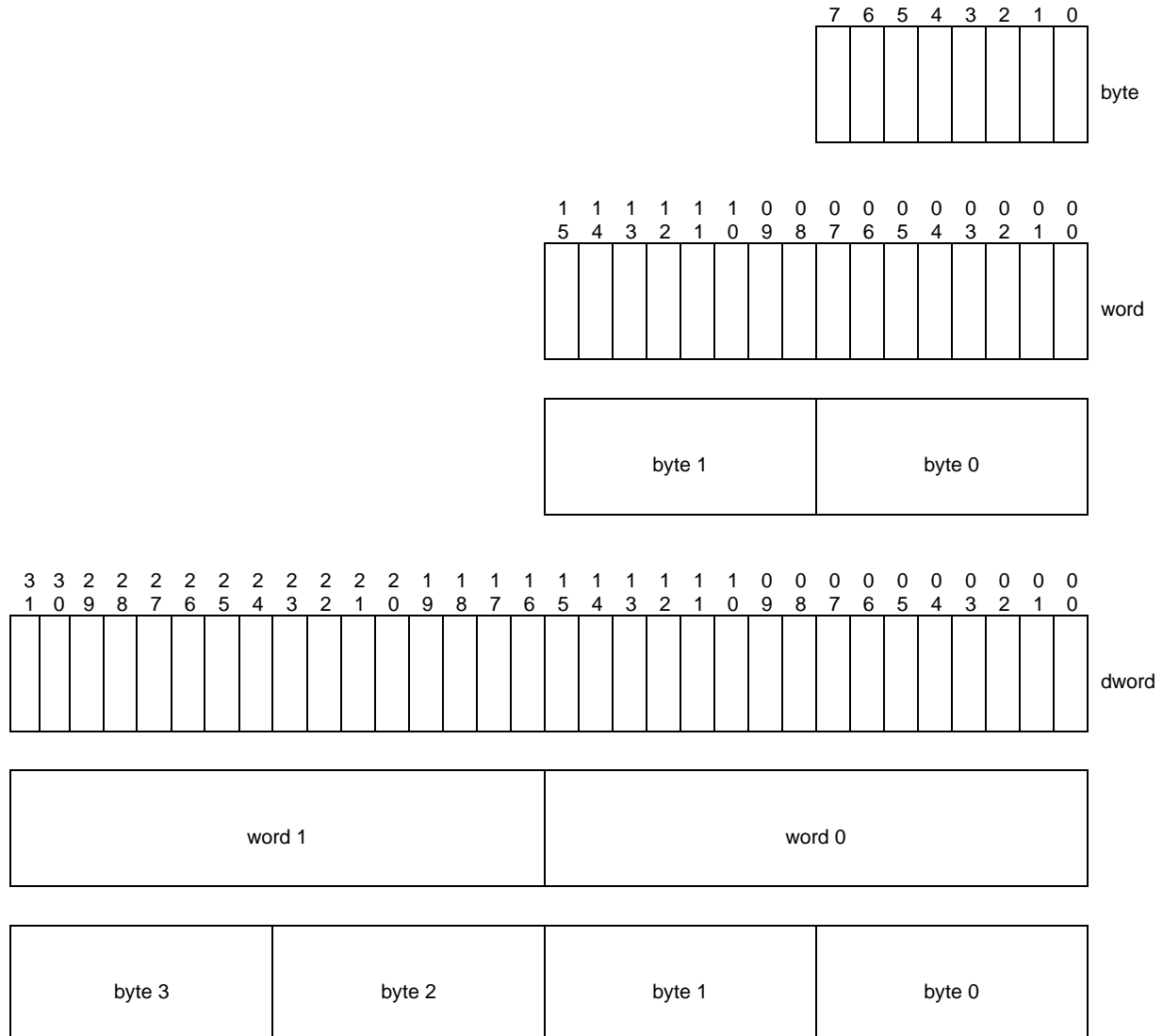
All other numbers are decimal (i.e., base 10). A decimal number is represented in this specification by any sequence of digits consisting of only the Western-Arabic numerals 0 to 9 not immediately followed by a lower-case b or a lower-case h (e.g., 175). This specification uses the following conventions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

- a) the decimal separator (i.e., separating the integer and fractional portions of the number) is a period;
- b) the thousands separator (i.e., separating groups of three decimal digits in a portion of the number) is a comma;
- c) the thousands separator is used in only the integer portion of a number and not the fractional portion of a number; and
- d) the decimal representation for a year does not include a comma (e.g., 2019 instead of 2,019).

### 1.4.3 Byte, Word, and Dword Relationships

Figure 3 illustrates the relationship between bytes, words and dwords. A qword (quadruple word) is a unit of data that is four times the size of a word; it is not illustrated due to space constraints. Unless otherwise specified, this specification specifies data in a little endian format.

**Figure 3: Byte, Word, and Dword Relationships**



## 1.5 Definitions

### 1.5.1 admin label

An admin label is an administratively configured ASCII or UTF-8 string (refer to section 1.4.2) that may be used to help identify specific NVMe entities (i.e., Hosts, NVM subsystems and namespaces). An admin label is capable of describing the entity's physical location, DNS name or other information.

### 1.5.2 admin label ASCII

An ASCII string. Refer to section 1.4.2 for ASCII string requirements. Refer to section 1.5.1 for admin label usage.

### **1.5.3 admin label UTF-8**

A UTF-8 string. Refer to section 1.4.2 for UTF-8 string requirements. Refer to section 1.5.1 for admin label usage.

### **1.5.4 Admin Queue**

The Admin Queue is the Submission Queue and Completion Queue with identifier 0. The Admin Submission Queue and corresponding Admin Completion Queue are used to submit administrative commands and receive completions for those administrative commands, respectively.

The Admin Submission Queue is uniquely associated with the Admin Completion Queue.

### **1.5.5 Administrative controller**

A controller that exposes capabilities that allow a host to manage an NVM subsystem. An Administrative controller does not implement I/O Queues, provide access to data or metadata associated with user data on a non-volatile storage medium, or support namespaces attached to the Administrative controller (i.e., there are never any active NSIDs).

### **1.5.6 allocated namespace**

A namespace that is associated with an allocated NSID.

### **1.5.7 Allowed Host List**

A list of hosts (identified by Host NQN and Host Identifier) present in each Exported NVM Subsystem that are granted access to the Exported NVM Subsystem via an Exported Port.

### **1.5.8 arbitration burst**

The maximum number of commands that may be fetched by an arbitration mechanism at one time from a Submission Queue.

### **1.5.9 arbitration mechanism**

The method used to determine which Submission Queue is selected next to fetch commands for execution by the controller. Refer to section 3.4.4.

### **1.5.10 association**

An exclusive communication relationship between a particular controller and a particular host that encompasses the Admin Queue and all I/O Queues of that controller.

### **1.5.11 audit**

The process of accessing media to determine correct operation of a sanitize operation. Refer to section 8.1.24 and to ISO/IEC 27040.

### **1.5.12 authentication commands**

Used to refer to Fabrics Authentication Send or Authentication Receive commands.

### **1.5.13 cache**

A data storage area used by the NVM subsystem, that is not accessible to a host, and that may contain a subset of user data stored in the non-volatile storage media or may contain user data that is not committed to non-volatile storage media.

### **1.5.14 candidate command**

A candidate command is a submitted command which has been transferred into the controller and the controller deems ready for processing.



### **1.5.15 capsule**

An NVMe unit of information exchange used in NVMe over Fabrics. A capsule contains a command or response and may optionally contain command/response data and SGLs.

### **1.5.16 Centralized Discovery controller (CDC)**

A Discovery controller that reports discovery information registered by Direct Discovery controllers and hosts.

### **1.5.17 Channel**

A Channel represents a communication path between the controller and one or more Media Units in an NVM subsystem.

### **1.5.18 command completion**

A command is completed when the controller has completed processing the command, has updated status information in the completion queue entry, and has posted the completion queue entry to the associated Completion Queue.

### **1.5.19 command submission**

For memory-based transport model (e.g., PCIe) implementations, a command is submitted when a Submission Queue Tail Doorbell write has completed that moves the Submission Queue Tail Pointer value past the Submission Queue slot in which the command was placed.

For message-based transport model (e.g., NVMe over Fabrics) implementations, a command is submitted when a host adds a capsule to a Submission Queue.

### **1.5.20 controller**

A controller is the interface between a host and an NVM subsystem. There are three types of controllers:

- a) I/O controllers;
- b) Discovery controllers; and
- c) Administrative controllers.

A controller executes commands submitted by a host on a Submission Queue and posts a completion on a Completion Queue. All controllers implement one Admin Submission Queue and one Admin Completion Queue. Depending on the controller type, a controller may also implement one or more I/O Submission Queues and I/O Completion Queues. When PCI Express is used as the transport, then a controller is a PCI Express function.

### **1.5.21 Controller Reset**

Host modification of the CC property that clears CC.EN from '1' to '0' (refer to section 3.7.2).

### **1.5.22 Directive**

A method of information exchange between a host and either an NVM subsystem or a controller. Information may be transmitted using the Directive Send and Directive Receive commands. A subset of I/O commands may include a Directive Type field and a Directive Specific field to communicate more information that is specific to the associated I/O command. Refer to section 8.1.8.

### **1.5.23 Direct Discovery controller (DDC)**

A Discovery controller that is capable of registering discovery information with a Centralized Discovery controller.

#### **1.5.24 Discovery controller**

A controller that exposes capabilities that allow a host to retrieve a Discovery Log Page. A Discovery controller does not implement I/O Queues or provide access to a non-volatile storage medium. Refer to section 3.1.3.3.

#### **1.5.25 discovery information**

Information about a host or NVM subsystem that is used for discovery (e.g., NVMe Transport address, NQN, etc.).

#### **1.5.26 Discovery Service**

An NVM subsystem that supports Discovery controllers only. A Discovery Service shall not support a controller that exposes namespaces.

#### **1.5.27 dispersed namespace**

A shared namespace that may be concurrently accessed by controllers in two or more NVM subsystems (refer to section 8.1.9).

#### **1.5.28 dynamic controller**

The controller is allocated on demand with no state (e.g., Feature settings) preserved from prior associations.

#### **1.5.29 Domain**

A domain is the smallest indivisible unit that shares state (e.g., power state, capacity information).

#### **1.5.30 embedded management controller**

An embedded management controller is a Management Controller (refer to the NVM Express Management Interface Specification) that provides an external management interface (e.g., Redfish®), typically implemented via commands to the Management Endpoint.

#### **1.5.31 emulated controller**

An NVM Express controller that is defined in software. An emulated controller may or may not have an underlying physical NVMe controller (e.g., physical PCIe function).

#### **1.5.32 Endurance Group**

A portion of non-volatile storage in the NVM subsystem whose endurance is managed as a group. Refer to section 3.2.3.

#### **1.5.33 Entry Key**

A set of discovery information entry fields that allow for the unique identification of each discovery information entry registered with the CDC or DDC. Refer to the Entry Key Type (EKTYPE) field.

#### **1.5.34 Exported Namespace**

A namespace in an Exported NVM Subsystem.

#### **1.5.35 Exported NVM Resources**

NVM resources created to enable remote access to physical NVM resources that includes:

- a) Exported NVM Subsystems;
- b) Exported Namespaces; and
- c) Exported Ports.

### **1.5.36 Exported NVM Subsystem**

A logical NVM subsystem that exports underlying NVM resources and that:

- a) contains zero or more Exported Namespaces;
- b) contains zero or more controllers;
- c) contains zero or more Exported Ports; and
- d) may contain an Allowed Host List.

### **1.5.37 Exported Port**

A port used to export an NVMe subsystem over a specific fabrics transport and represented by an Exported Port ID.

### **1.5.38 Exported Port ID**

A port identifier used to specify an Exported Port.

### **1.5.39 fabric (network fabric)**

A network topology in which nodes pass data to each other.

### **1.5.40 Fabric Zoning**

A technique to specify access control configurations between hosts and NVM subsystems.

### **1.5.41 firmware/boot partition image update command sequence**

The sequence of one or more Firmware Image Download commands that download a firmware image or a boot partition image followed by a Firmware Commit command that commits that downloaded image to a firmware slot or a boot partition.

### **1.5.42 firmware slot**

A firmware slot is a location in a domain used to store a firmware image. The domain stores from one to seven firmware images. Controllers in the same domain share the same firmware slots.

### **1.5.43 host**

An entity that interfaces to an NVM subsystem through one or more controllers and submits commands to Submission Queues and retrieves command completions from Completion Queues.

### **1.5.44 host-accessible memory**

Memory that the host is able to access (e.g., host memory, Controller Memory Buffer (CMB), Persistent Memory Region (PMR)).

### **1.5.45 host management agent**

A host management agent is a part of the host that provides an external management interface (e.g., Redfish) to external managers, typically via Admin commands to the controller (refer to the NVM Express Management Interface Specification).

### **1.5.46 host memory**

Memory that may be read and written by both a host and a controller and that is not exposed by a controller (i.e., Controller Memory Buffer or Persistent Memory Region). Host memory may be implemented inside or outside a host (e.g., a memory region exposed by a device that is neither the host nor controller).

### **1.5.47 idempotent command**

A command that produces the same end state in the NVM subsystem and returns the same results if that command is resubmitted one or more times with no intervening commands. Refer to section 9.6.3.1.

**1.5.48 Identify Controller data structures**

All controller data structures that are able to be retrieved via the Identify command:

- Identify Controller data structure (i.e., CNS 01h); and
- each of the I/O Command Set specific Identify Controller data structure (i.e., CNS 06h).

**1.5.49 Identify Namespace data structures**

All namespace data structures that are able to be retrieved via the Identify command:

- Identify Namespace data structures (i.e., CNS 00h, CNS 09h, and CNS 11h);
- I/O Command Set Independent Identify Namespace data structures (i.e., CNS 08h and CNS 1Fh); and
- I/O Command Set specific Identify Namespace data structures (i.e., CNS 05h, CNS 0Ah, and CNS 1Bh).

**1.5.50 I/O command**

An I/O command is a command submitted to an I/O Submission Queue.

**1.5.51 I/O Completion Queue**

An I/O Completion Queue is a Completion Queue that is used to indicate command completions and is associated with one or more I/O Submission Queues.

**1.5.52 I/O controller**

A controller that implements I/O queues and is intended to be used to access a non-volatile storage medium.

**1.5.53 I/O Submission Queue**

An I/O Submission Queue is a Submission Queue that is used to submit I/O commands for execution by the controller (e.g., Read command and Write command for the NVM Command Set).

**1.5.54 Media Unit**

A Media Unit represents a component of the underlying media in an NVM subsystem. Endurance Groups are composed of Media Units.

**1.5.55 memory-based controller**

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

**1.5.56 message-based controller**

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

**1.5.57 metadata**

Metadata is contextual information related to formatted user data (e.g., a particular LBA of data as defined in the NVM Command Set Specification). The host may include metadata to be stored by the NVM subsystem if storage space is provided by the controller. Refer to the applicable I/O Command Set specification for details.

**1.5.58 MMC**

A Migration Management Controller. Refer to section 8.1.12.

**1.5.59 MMH**

A Migration Management Host. Refer to section 8.1.12.

#### **1.5.60 MMHD**

A Migration Management Host associated with a Migration Management Controller in a Destination NVM Subsystem. Refer to section 8.1.12.

#### **1.5.61 MMHS**

A Migration Management Host associated with a Migration Management Controller in a Source NVM Subsystem. Refer to section 8.1.12.

#### **1.5.62 namespace**

A set of resources that may be directly accessed by a host (e.g., formatted non-volatile storage).

#### **1.5.63 namespace ID (NSID)**

An identifier used by a controller to provide access to a namespace or the name of the field in the SQE that contains the namespace identifier (refer to Figure 92). Refer to section 3.2.1 for the definitions of valid NSID, invalid NSID, active NSID, inactive NSID, allocated NSID, and unallocated NSID.

#### **1.5.64 NVM**

NVM is an acronym for non-volatile memory.

#### **1.5.65 NVM Set**

A portion of NVM from an Endurance Group. Refer to section 3.2.2.

#### **1.5.66 NVM subsystem**

An NVM subsystem includes one or more domains, one or more controllers, zero or more namespaces, and one or more ports. An NVM subsystem may include a non-volatile storage medium and an interface between the controller(s) in the NVM subsystem and non-volatile storage medium.

#### **1.5.67 NVM subsystem port**

An NVMe over Fabrics protocol interface between an NVM subsystem and a fabric. An NVM subsystem port is a collection of one or more physical fabric interfaces that together act as a single interface.

#### **1.5.68 NVMe over Fabrics**

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

#### **1.5.69 NVMe Transport**

A protocol layer that provides reliable delivery of data, commands, and responses between a host and an NVM subsystem. The NVMe Transport layer is layered on top of the fabric. It is independent of the fabric physical interconnect and low-level fabric protocol layers.

#### **1.5.70 NVMe Transport binding specification**

A specification of reliable delivery of data, commands, and responses between a host and an NVM subsystem for an NVMe Transport. The binding may exclude or restrict functionality based on the NVMe Transport's capabilities.

#### **1.5.71 participating NVM subsystem**

An NVM subsystem that participates in (i.e., contains controllers that provide access to) a dispersed namespace.

**1.5.72 physical fabric interface (physical ports)**

A physical connection between an NVM subsystem and a fabric.

**1.5.73 Placement Handle**

A namespace scoped handle that maps to an Endurance Group scoped Reclaim Unit Handle which references a Reclaim Unit in each Reclaim Group.

**1.5.74 Placement Identifier**

A data structure that specifies a Reclaim Group Identifier and a Placement Handle that references a Reclaim Unit. Refer to Figure 282 and Figure 283.

**1.5.75 Port ID**

An identifier that is associated with an NVM subsystem port. Refer to section 2.2.2.

**1.5.76 Ports List**

A list of ports that may be used to export an NVM subsystem. Entries in the Ports List are in the format specified by Underlying Fabrics Transport Entry data structure (refer to Figure 336).

**1.5.77 Power Loss Acknowledge (PLA)**

The transport-specific variable that is used by the controller to inform the host of the controller's current Power Loss Signaling processing (refer to section 8.2.5).

**1.5.78 Power Loss Notification (PLN)**

The transport-specific variable that is used to inform the controller that a main power loss event is expected to occur (refer to section 8.2.5).

**1.5.79 primary controller**

An NVM Express controller that supports the Virtualization Management command. An NVM subsystem may contain multiple primary controllers. Secondary controller(s) in an NVM subsystem depend on a primary controller for dynamic resource management (refer to section 8.2.6).

A PCI Express SR-IOV Physical Function that supports the NVM Express interface and the Virtualization Enhancements capability is an example of a primary controller (refer to section 8.2.6.4).

**1.5.80 private namespace**

A namespace that is only able to be attached to one controller at a time. Refer to the Namespace Multi-path I/O and Namespace Sharing Capabilities (NMIC) field in Figure 319.

**1.5.81 property**

The generalization of memory mapped controller registers defined for NVMe over PCIe. Properties are used to configure low level controller attributes and obtain low level controller status. Refer to section 3.1.4.

**1.5.82 Reclaim Group (RG)**

An entity that contains one or more Reclaim Units. Refer to section 3.2.4.

**1.5.83 Reclaim Unit (RU)**

A logical representation of non-volatile storage within a Reclaim Group that is able to be physically erased by the controller without disturbing any other Reclaim Units. Refer to section 3.2.4.

**1.5.84 Reclaim Unit Handle (RUH)**

A controller resource that references a Reclaim Unit in each Reclaim Group. Refer to section 3.2.4.

#### **1.5.85 rotational media**

Media that stores data on rotating platters (refer to section 8.1.23).

#### **1.5.86 Runtime D3 (Power Removed)**

In Runtime D3 (RTD3) main power is removed from the controller. Auxiliary power may or may not be provided. For PCI Express, RTD3 is the D3<sub>cold</sub> power state (refer to section 8.1.17.4).

#### **1.5.87 sanitize operation**

Process by which all user data in the NVM subsystem is altered such that recovery of the previous user data from any cache or the non-volatile storage media is infeasible for a given level of effort (refer to IEEE 2883™-2022).

#### **1.5.88 sanitization target**

The target of a sanitize operation (i.e., an NVM subsystem).

#### **1.5.89 secondary controller**

An NVM Express controller that depends on a primary controller in an NVM subsystem for management of some controller resources (refer to section 8.2.6).

A PCI Express SR-IOV Virtual Function that supports the NVM Express interface and receives resources from a primary controller is an example of a secondary controller (refer to section 8.2.6.4).

#### **1.5.90 shared namespace**

A namespace that may be attached to two or more controllers in an NVM subsystem concurrently. Refer to the Namespace Multi-path I/O and Namespace Sharing Capabilities (NMIC) field in Figure 319.

#### **1.5.91 specified namespace**

The namespace that is associated with the value specified by the Namespace Identifier (NSID) field in a command as defined by the Common Command Format (refer to Figure 92).

#### **1.5.92 spindown**

The process of changing a spindle from an operational power state to a non-operational power state, for an Endurance Group that stores data on rotational media (refer to section 8.1.23).

#### **1.5.93 spinup**

The process of changing a spindle from a non-operational power state to an operational power state, for an Endurance Group associated with rotational media (refer to section 8.1.23).

#### **1.5.94 static controller**

The controller is pre-existing with a specific Controller ID and its state (e.g., Feature settings) is preserved from prior associations.

#### **1.5.95 Underlying Namespace**

A namespace (defined in section 1.5.62) accessible through physical or virtual functions in an Underlying NVM Subsystem that may be used to associate with an Exported NVM Subsystem. Underlying Namespaces are identified by the Underlying Namespace Entry data structure (refer to Figure 334).

#### **1.5.96 Underlying Namespace List**

A list of namespaces (refer to section 5.1.13.4.1) in all underlying NVM subsystems that may be used to create an Exported Namespace.

**1.5.97 Underlying NVM Subsystem**

Defined as NVM subsystem.

**1.5.98 Underlying Port**

A port through which an NVMe subsystem is attached to a transport (e.g., Ethernet, InfiniBand, Fibre Channel) (refer to section 1.5.72).

**1.5.99 user data**

Data stored in a namespace that is composed of data that the host may store and later retrieve including metadata if supported.

**1.6 I/O Command Set specific definitions used in this specification**

The following terms used in this specification are defined in each I/O Command Set specification.

**1.6.1 Endurance Group Host Read Command**

An I/O Command Set specific command that results in the controller reading user data, but may or may not return the data to the host.

**1.6.2 Format Index**

A value used to index into the I/O Command Set Specific Format table (i.e., the User Data Format number).

**1.6.3 SMART Data Units Read Command**

An I/O Command Set specific command that results in the controller reading user data, but may or may not return the data to the host.

**1.6.4 SMART Host Read Command**

An I/O Command Set specific command that results in the controller reading user data, but may or may not return the data to the host.

**1.6.5 User Data Format**

An I/O Command Set specific format that describes the layout of the data on the NVM media.

**1.6.6 User Data Out Command**

An I/O Command Set specific command that results in the controller writing user data, but may or may not transfer user data from the host to the controller.

**1.7 NVM Command Set specific definitions used in this specification**

The following terms used in this specification are defined in the NVM Command Set Specification. These terms are used throughout the document as examples for a specific I/O Command Set.

**1.7.1 logical block**

The smallest addressable data unit for Read and Write commands.

**1.7.2 logical block address (LBA)**

The address of a logical block, referred to commonly as LBA.

**1.8 References**

CNSA 1.0, "USE OF PUBLIC STANDARDS FOR SECURE INFORMATION SHARING", CNSSP 15 ANNEX B "NSA-APPROVED COMMERCIAL NATIONAL SECURITY ALGORITHM (CNSA) SUITE", 20 October 2016. Available from <https://www.cnss.gov/CNSS/issuances/Policies.cfm>.



IEEE Std 2883™-2022, IEEE Standard for Sanitizing Storage. Available from <https://standards.ieee.org>.

INCITS 502-2019, Information Technology – SCSI Primary Commands - 5 (SPC-5). Available from <https://webstore.ansi.org>.

INCITS 556-2020, Information Technology – Non-Volatile Memory Express - 2 (FC-NVMe-2). Available from <https://webstore.ansi.org>.

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

ISO/IEC 27040:2024 Information technology – Security techniques – Storage security. Available from <https://www.iso.org>.

JEDEC JESD218B-02: Solid State Drive (SSD) Requirements and Endurance Test Method standard. Available from <https://www.jedec.org>.

NVM Express Boot Specification, Revision 1.1. Available from <https://www.nvmexpress.org>.

NVM Express Management Interface Specification, Revision 2.0. Available from <https://www.nvmexpress.org>.

NVM Express NVM Command Set Specification, Revision 1.1. Available from <https://www.nvmexpress.org>.

NVM Express Zoned Namespace Command Set Specification, Revision 1.2. Available from <https://www.nvmexpress.org>.

NVM Express Key Value Command Set Specification, Revision 1.1. Available from <https://www.nvmexpress.org>.

NVM Express NVMe over PCIe Transport Specification, Revision 1.1. Available from <https://www.nvmexpress.org>.

NVM Express RDMA Transport Specification, Revision 1.1. Available from <https://www.nvmexpress.org>.

NVM Express TCP Transport Specification, Revision 1.1. Available from <https://www.nvmexpress.org>.

PCI-SIG PCI Express® Base Specification, Revision 6.2. Available from <https://www.pcisig.com>.

RFC 1952, P. Deutsch, “GZIP file format specification version 4.3”, May 1996. Available from <https://www.rfc-editor.org/info/rfc1952>.

RFC 1994, W. Simpson, “PPP Challenge Handshake Authentication Protocol (CHAP)”, August 1996. Available from <https://www.rfc-editor.org/info/rfc1994>.

RFC 2104, H. Krawczyk, M. Bellare, R. Canetti, “HMAC: Keyed-Hashing for Message Authentication”, February 1997. Available from <https://www.rfc-editor.org/info/rfc2104>.

RFC 2631, E. Rescorla, “Diffie-Hellman Key Agreement Method”, June 1999. Available from <https://www.rfc-editor.org/info/rfc2631>.

RFC 3629, F. Yergeau, “UTF-8, a transformation format of ISO 10646”, November 2003. Available from <https://www.rfc-editor.org/info/rfc3629>.

RFC 3986, T. Berners-Lee, R. Fielding, L. Masinter, “Uniform Resource Identifier (URI): Generic Syntax”, January 2005. Available from <https://www.rfc-editor.org/info/rfc3986>.

RFC 4086, D. Eastlake 3rd, J. Schiller, S. Crocker, “Randomness Requirements for Security”, June 2005. Available from <https://www.rfc-editor.org/info/rfc4086>.

RFC 4088, D. Black, K. McCloghrie, J. Schoenwaelder, “Uniform Resource Identifier (URI) Scheme for the Simple Network Management Protocol (SNMP)”, June 2005. Available from <https://www.rfc-editor.org/info/rfc4088>.

RFC 4301, S. Kent, K. Seo, “Security Architecture for the Internet Protocol”, December 2005. Available from <https://www.rfc-editor.org/info/rfc4301>.

RFC 4648, S. Josefsson, "The Base16, Base32, and Base64 Data Encodings", October 2006. Available from <https://www.rfc-editor.org/info/rfc4648>.

RFC 5869, H. Krawczyk, P. Eronen, "HMAC-based Extract-and-Expand Key Derivation Function (HKDF)", May 2010. Available from <https://www.rfc-editor.org/info/rfc5869>.

RFC 6234, D. Eastlake 3rd, and T. Hansen, "US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF)", May 2011. Available from <https://www.rfc-editor.org/info/rfc6234>.

RFC 6520, R. Seggelmann, M. Tuexen, M. Williams, "Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) Heartbeat Extension", February 2012. Available from <https://www.rfc-editor.org/info/rfc6520>.

RFC 7296, C. Kaufman, P. Hoffman, Y. Nir, P. Eronen, T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", October 2014. Available from <https://www.rfc-editor.org/info/rfc7296>.

RFC 7919, D. Gillmor, "Negotiated Finite Field Diffie-Hellman Ephemeral Parameters for Transport Layer Security (TLS)", August 2016. Available from <https://www.rfc-editor.org/info/rfc7919>.

RFC 8446, E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.3", August 2018. Available from <https://www.rfc-editor.org/info/rfc8446>.

RFC 9562, K. Davis, B. Peabody, and P. Leach, "Universally Unique Identifiers, May 2024". Available from <https://www.rfc-editor.org/info/rfc9562>.

UEFI Specification Version 2.10, August 2022. Available from <https://uefi.org>.

Advanced Configuration and Power Interface (ACPI) Specification, Version 6.5, August 2022. Available from <https://www.uefi.org>.

TCG Storage Architecture Core Specification, Version 2.01 Revision 1.00. Available from <https://www.trustedcomputinggroup.org>.

TCG Storage Interface Interactions Specification (SIIS), Version 1.11 Revision 1.18. Available from <https://www.trustedcomputinggroup.org>.

TCG Storage Security Subsystem Class: Key Per I/O Version 1.00 Revision 1.41. Available from <https://trustedcomputinggroup.org>.

## 1.9 References Under Development

None.

## 2 Theory of Operation

The NVM Express scalable interface is designed to address the needs of storage systems that utilize PCI Express based solid state drives or fabric connected devices. The interface provides optimized command submission and completion paths. It includes support for parallel operation by supporting up to 65,535 I/O Queues with up to 65,535 outstanding commands per I/O Queue. Additionally, support has been added for many Enterprise capabilities like end-to-end data protection (compatible with SCSI Protection Information, commonly known as T10 DIF, and SNIA DIX standards), enhanced error reporting, and virtualization.

The interface has the following key attributes:

- Does not require uncacheable / MMIO register reads in the command submission or completion path;
- A maximum of one MMIO register write or one 64B message is necessary in the command submission path;
- Support for up to 65,535 I/O Queues, with each I/O Queue supporting up to 65,535 outstanding commands;
- Priority associated with each I/O Queue with well-defined arbitration mechanism;
- All information to complete a 4 KiB read request is included in the 64B command itself, ensuring efficient small I/O operation;
- Efficient and streamlined command set;
- Support for MSI/MSI-X and interrupt aggregation;
- Support for multiple namespaces;
- Efficient support for I/O virtualization architectures like SR-IOV;
- Robust error reporting and management capabilities; and
- Support for multi-path I/O and namespace sharing.

This specification defines a streamlined set of properties that are used to configure low level controller attributes and obtain low level controller status. These properties have a transport specific mechanism for defining access (e.g., memory-based transports use registers, whereas message-based transports use the Property Get and Property Set commands). The following are examples of functionality defined in properties:

- Indication of controller capabilities;
- Status for controller failures (command status is provided in a CQE);
- Admin Queue configuration (I/O Queue configuration processed via Admin commands); and
- Doorbell registers for a scalable number of Submission and Completion Queues.

There are two defined models for communication between the host and the NVM subsystem, a memory-based transport model and a message-based transport model. All NVM 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.

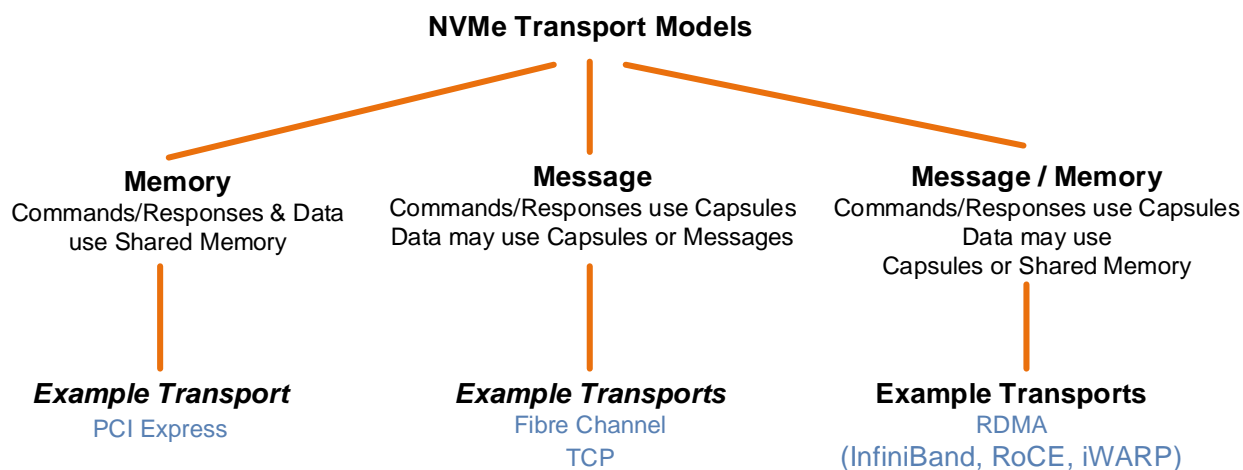
A memory-based transport model is one in which commands, responses, and data are transferred between a host and an NVM subsystem by performing explicit memory read and write operations (e.g., over PCIe).

A message-based transport model is one in which messages containing command capsules and response capsules are sent between a host and an NVM subsystem (e.g., over a fabric). The two subtypes of message-based transport models are differentiated by how data is sent between a host and an NVM subsystem. In the message-only transport model data is only sent between a host and an NVM subsystem using capsules or messages. The message/memory 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 NVM subsystem. Data may optionally be included in command capsules and response capsules. Both the message-only transport model and the message/memory transport model are

referenced as message-based transport models throughout this specification when the description is applicable to both subtypes.

An NVM subsystem is made up of a single domain or multiple domains as described in section 3.2.5. An NVM subsystem may optionally include a non-volatile storage medium, and an interface between the controller(s) of the NVM subsystem and the non-volatile storage medium. Controllers expose this non-volatile storage medium to hosts through namespaces. An NVM subsystem is not required to have the same namespaces attached to all controllers. An NVM subsystem that supports a Discovery controller does not support any other controller type. A Discovery Service is an NVM subsystem that supports Discovery controllers only (refer to section 3.1).

**Figure 4: Taxonomy of Transport Models**



The capabilities and settings that apply to an NVM Express controller are indicated in the Controller Capabilities (CAP) property and the Identify Controller data structure (refer to Figure 312).

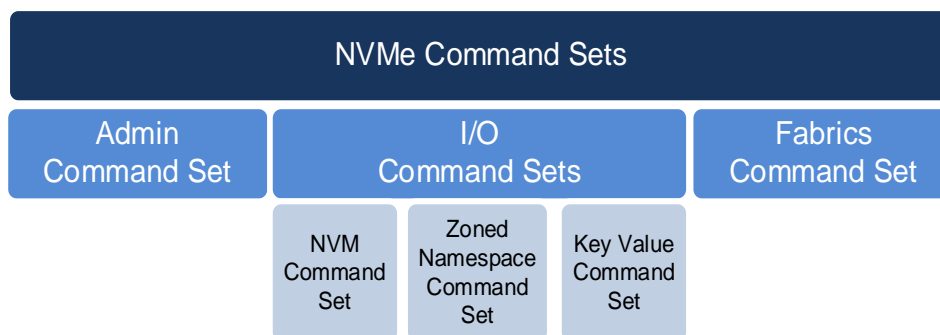
A namespace is a set of resources (e.g., formatted non-volatile storage) that may be accessed by a host. A namespace has an associated namespace identifier that a host uses to access that namespace. The set of resources may consist of non-volatile storage and/or other resources.

Associated with each namespace is an I/O Command Set that operates on that namespace. An NVM Express controller may support multiple namespaces. Namespaces may be created and deleted using the Namespace Management command and Capacity Management command. The Identify Namespace data structures (refer to section 1.5.49) indicate capabilities and settings that are specific to a particular namespace.

The NVM Express interface is based on a paired Submission and Completion Queue mechanism. Commands are placed by host software into a Submission Queue. Completions are placed into the associated Completion Queue by the controller.

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

**Figure 5: Types of NVMe Command Sets**



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

An I/O Command Set is used with an I/O queue pair. This specification defines common I/O commands. I/O Command Sets are defined in I/O Command Set specifications. The example I/O Command Sets shown in Figure 5 are the NVM Command Set, the Key Value Command Set, and the Zoned Namespace Command Set. Other I/O Command Sets include the Computational Programs Command Set and the SLM Command Set.

The Fabrics Command Set is NVMe over Fabrics specific. Fabrics Command Set commands are used for operations specific to NVMe over Fabrics including establishing a connection, NVMe in-band authentication, and to get or set a property. All Fabrics commands may be submitted on the Admin Submission Queue and some Fabrics commands may also be submitted on an I/O Submission Queue. Unlike Admin and I/O commands, Fabrics commands are processed by a controller regardless of whether the controller is enabled (i.e., regardless of the state of CC.EN).

## 2.1 Memory-Based Transport Model (PCIe)

In the memory-based model, Submission and Completion Queues are allocated in memory.

Host software creates queues, up to the maximum supported by the controller. Typically, the number of command queues created is based on the system configuration and anticipated workload. For example, on a four core processor based system, there may be a queue pair per core to avoid locking and ensure data structures are created in the appropriate processor core's cache. Figure 6 provides a graphical representation of the queue pair mechanism, showing a 1:1 mapping between Submission Queues and Completion Queues. Figure 7 shows an example where multiple I/O Submission Queues utilize the same I/O Completion Queue on Core B. Figure 6 and Figure 7 show that there is always a 1:1 mapping between the Admin Submission Queue and Admin Completion Queue.

Figure 6: Queue Pair Example, 1:1 Mapping

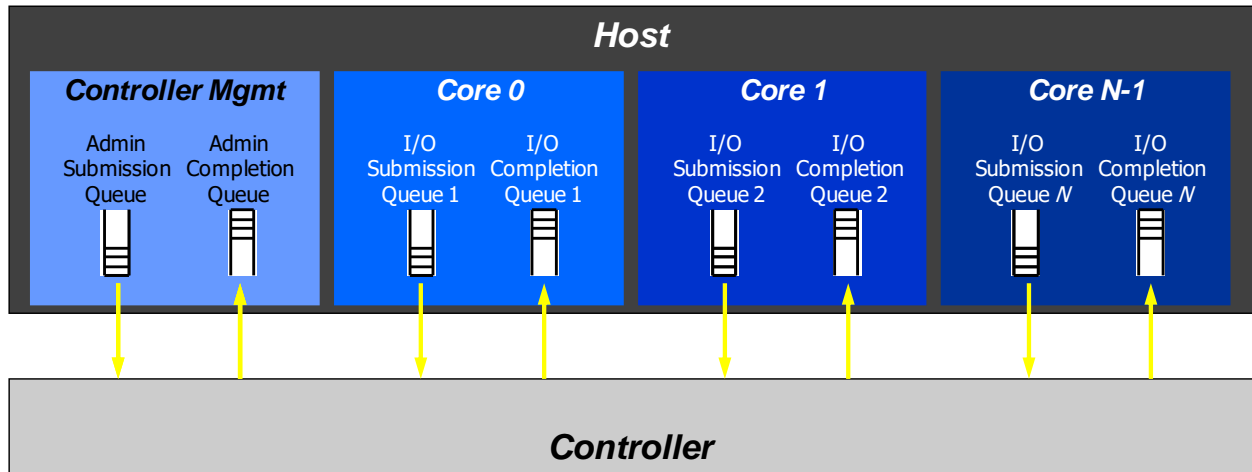
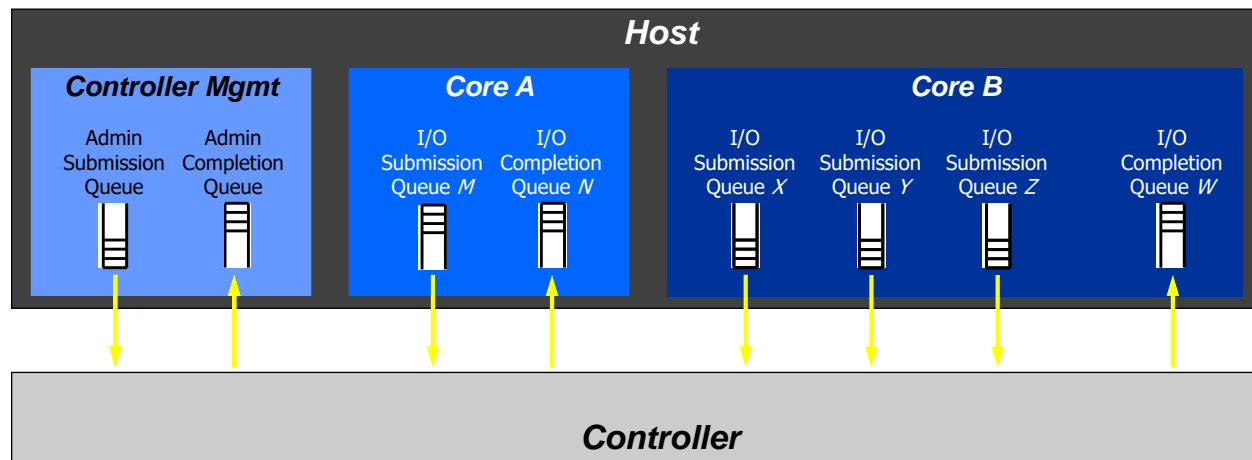


Figure 7: Queue Pair Example, *n*:1 Mapping



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

Each submission queue entry is a command. Commands are 64 bytes in size. The physical memory locations in memory to use for data transfers are specified using Physical Region Page (PRP) entries or Scatter Gather Lists (SGL). Each command may include two PRP entries or one Scatter Gather List segment. If more than two PRP entries are necessary to describe the data buffer, then a pointer to a PRP List that describes a list of PRP entries is provided. If more than one SGL segment is necessary to describe the data buffer, then the SGL segment provides a pointer to the next SGL segment.

A Completion Queue (CQ) is a circular buffer with a fixed slot size used to post status for completed commands. A completed command is uniquely identified by a combination of the associated SQ identifier and command identifier that is assigned by host software. In the memory-based transport model multiple Submission Queues may be associated with a single Completion Queue. A configuration with a single Completion Queue may be used where a single worker thread processes all command completions via one Completion Queue even when those commands originated from multiple Submission Queues. The CQ Head pointer is updated by host software after processing completion queue entries indicating the last free

CQ slot. A Phase Tag (P) bit is defined in the completion queue entry to indicate whether an entry has been newly posted without the host consulting a register (refer to section 4.2.4). The Phase Tag bit enables the host to determine whether entries are new or not.

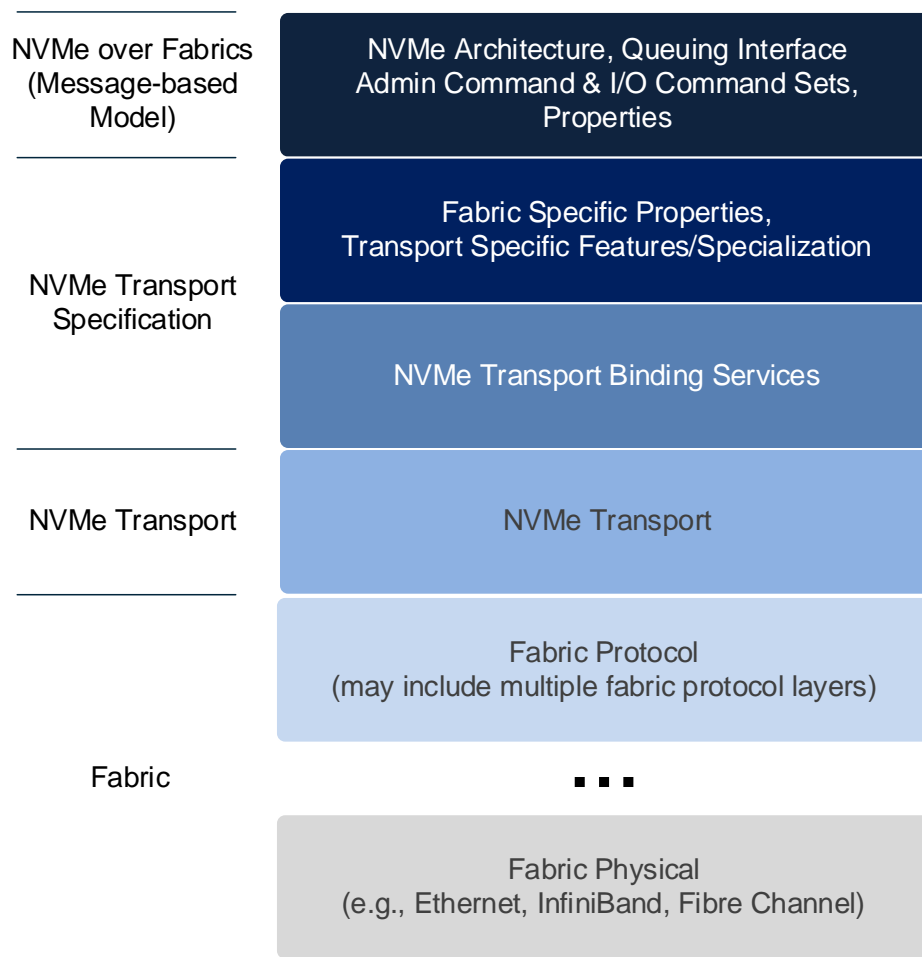
## 2.2 Message-Based Transport Model (Fabrics)

The message-based transport model used for NVMe over Fabrics has the following differences from the memory-based transport model:

- There is a one-to-one mapping between I/O Submission Queues and I/O Completion Queues. NVMe over Fabrics does not support multiple I/O Submission Queues being mapped to a single I/O Completion Queue;
- NVMe over Fabrics does not define an interrupt mechanism that allows a controller to generate a host interrupt. It is the responsibility of the host fabric interface (e.g., Host Bus Adapter) to generate host interrupts;
- NVMe over Fabrics uses different mechanisms for I/O Submission Queue and I/O Completion Queue creation and deletion (refer to section 3.5);
- NVMe over Fabrics does not support transferring metadata from a separate buffer (e.g., does not support the Metadata Pointer field, refer to Figure 92);
- NVMe over Fabrics does not support PRPs but requires use of SGLs for Admin, I/O, and Fabrics commands. This differs from the memory-based transport model where SGLs are not supported for Admin commands and are optional for I/O commands;
- NVMe over Fabrics does not support Completion Queue flow control (refer to section 3.3.1.2.1). This requires that the host ensures there are available Completion Queue slots before submitting new commands; and
- NVMe over Fabrics allows Submission Queue flow control to be disabled if the host and controller agree to disable Submission Queue flow control. If Submission Queue flow control is disabled, the host is required to ensure that there are available Submission Queue slots before submitting new commands.

### 2.2.1 Fabrics and Transports

NVMe over Fabrics utilizes the protocol layering shown in Figure 8. This specification defines core aspects of the architecture that are independent of the NVMe Transport. An NVMe Transport binding specification is used to describe any NVMe Transport specific specialization as well as how the services required by the NVMe interface are mapped onto the corresponding NVMe Transport. The native fabric communication services and other functionality used by the NVMe interface and NVMe Transports (e.g., the Fabric Protocol and Fabric Physical layers in Figure 8) are outside the scope of the NVMe family of specifications.

**Figure 8: NVMe over Fabrics Layering**

### 2.2.2 NVM Subsystem Ports for Fabrics

An NVM subsystem presents a collection of one to (64Ki - 16) controllers which are used to access namespaces. The controllers may be associated with hosts through one to 64Ki NVM subsystem ports.

An NVM subsystem port is a protocol interface between an NVM subsystem and a fabric. An NVM subsystem port is a collection of one or more physical fabric interfaces that together act as a single protocol interface. When link aggregation (e.g., Ethernet) is used, the physical ports for the group of aggregated links constitute a single NVM subsystem port.

An NVM subsystem contains one or more NVM subsystem ports.

Each NVM subsystem port has a 16-bit port identifier (Port ID). An NVM subsystem port is identified by the NVM Subsystem NVMe Qualified Name (NQN) and Port ID. The NVM subsystem ports of an NVM subsystem may support different NVMe Transports. An NVM subsystem port may support multiple NVMe Transports if more than one NVMe Transport binding specifications exist for the underlying fabric (e.g., an NVM subsystem port identified by a Port ID may support both iWARP and RoCE). An NVM subsystem implementation may bind specific controllers to specific NVM subsystem ports or allow the flexible allocation of controllers between NVM subsystem ports, however, once connected, each specific controller is bound to a single NVM subsystem port.



A controller is associated with exactly one host at a time. NVMe over Fabrics allows multiple hosts to connect to different controllers in the NVM subsystem through the same NVM subsystem port. All other aspects of NVMe over Fabrics multi-path I/O and namespace sharing (refer to section 2.4.1) are equivalent to that of the memory-based transport model.

### 2.2.3 Discovery Service

NVMe over Fabrics defines a discovery mechanism that a host uses to determine the NVM subsystems that expose namespaces that the host may access. The Discovery Service provides a host with the following capabilities:

- The ability to discover a list of NVM subsystems with namespaces that are accessible to the host;
- The ability to discover multiple paths to an NVM subsystem;
- The ability to discover controllers that are statically configured;
- The optional ability to establish explicit persistent connections to the Discovery controller; and
- The optional ability to receive Asynchronous Event Notifications from the Discovery controller.

A Discovery Service is an NVM subsystem that supports only Discovery controllers (refer to section 3.1.3.3), and shall not support any other controller type.

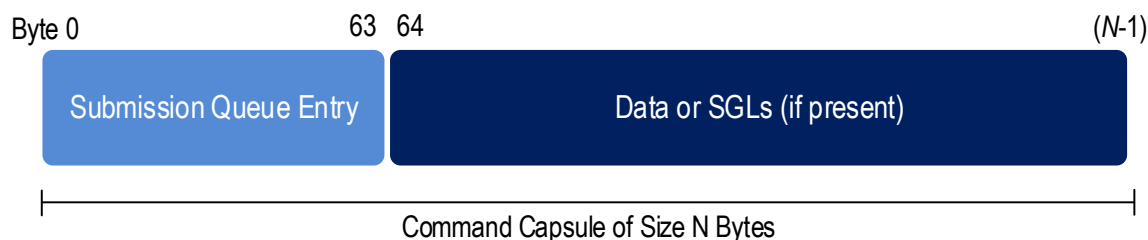
The method that a host uses to obtain the information necessary to connect to the initial Discovery Service is implementation specific. This information may be determined using a host configuration file, a hypervisor or OS property, or some other mechanism.

### 2.2.4 Capsules and Data Transfer

A capsule is an NVMe unit of information exchange used in NVMe over Fabrics. A capsule may be classified as a command capsule or a response capsule. A command capsule contains a command (formatted as a submission queue entry) and may optionally include SGLs or data. A response capsule contains a response (formatted as a completion queue entry) and may optionally include data. Data refers to any data transferred at an NVMe layer between a host and an NVM subsystem (e.g., logical block data or a data structure associated with a command). A capsule is independent of any underlying NVMe Transport unit (e.g., packet, message, or frame and associated headers and footers) and may consist of multiple such units.

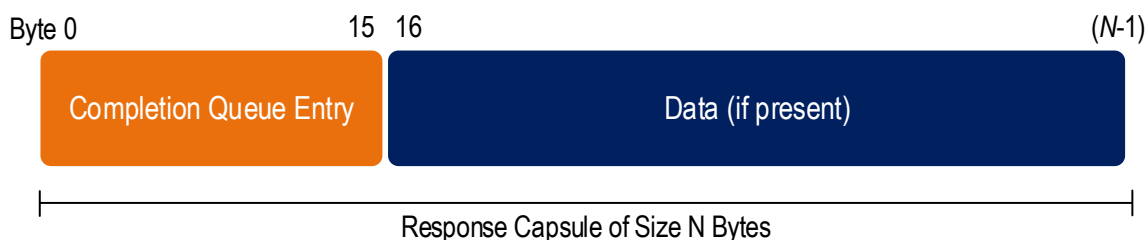
Command capsules are transferred from a host to an NVM subsystem. The SQE contains an Admin command, an I/O command, or a Fabrics command. The minimum size of a command capsule is NVMe Transport binding specific, but shall be at least 64B in size. The maximum size of a command capsule is NVMe Transport binding specific. The format of a command capsule is shown in Figure 9.

**Figure 9: Command Capsule Format**



Response capsules are transferred from an NVM subsystem to a host. The CQE is associated with a previously issued Admin command, I/O command, or Fabrics command. The size of a response capsule is NVMe Transport binding specific, but shall be at least 16B in size. The maximum size of a response capsule is NVMe Transport binding specific. The format of a response capsule is shown in Figure 10.

**Figure 10: Response Capsule Format**



NVMe Transports using the message-only transport model and message/memory transport model require all SGLs sent from the host to the controller be transferred within the command. NVMe Transports may optionally support the transfer of a portion or all data within the command and response capsules.

NVMe over Fabrics requires SGLs for all commands (Fabrics, Admin, and I/O). An SGL may specify the placement of data within a capsule or the information required to transfer data using an NVMe Transport specific data transfer mechanism (e.g., via memory transfers as in RDMA). Each NVMe Transport binding specification defines the SGLs used by a particular NVMe Transport and any capsule SGL and data placement restrictions.

### 2.2.5 Authentication

NVMe over Fabrics supports both fabric secure channel that includes authentication (refer to section 8.3.4.1) and NVMe in-band authentication. An NVM subsystem may require a host to use fabric secure channel, NVMe in-band authentication, or both. The Discovery Service indicates if fabric secure channel shall be used for an NVM subsystem. The Connect response indicates if NVMe in-band authentication shall be used with that controller.

A controller associated with an NVM subsystem that requires a fabric secure channel shall not accept any commands (i.e., Fabrics commands, Admin commands, or I/O commands) on an NVMe Transport until a secure channel is established. Following a Connect command, a controller that requires NVMe in-band authentication shall not accept any commands on the queue created by that Connect command other than authentication commands until NVMe in-band authentication has completed. Refer to section 8.3.4.

## 2.3 NVM Storage Model

### 2.3.1 Storage Entities

The NVM storage model includes the following entities:

- NVM subsystems (refer to 1.5.66);
- Domains (refer to section 3.2.5);
- Endurance Groups (refer to section 3.2.3);
- Reclaim Groups and Reclaim Units (refer to section 3.2.4);
- NVM Sets (refer to section 3.2.2); and
- Namespaces (refer to section 3.2.1).

As illustrated below,

- each domain is contained in a single NVM subsystem;
- each Endurance Group is contained in a single domain and may contain either:
  - one or more NVM Sets; or
  - one or more Reclaim Groups;
- each NVM Set is contained in a single Endurance Group and each namespace is contained in a single NVM Set. Each Media Unit is contained in a single Endurance Group; and
- each Reclaim Group is contained in a single Endurance Group, each Reclaim Unit is contained in a Reclaim Group, and each namespace is contained in an Endurance Group within one or more Reclaim Units of the Reclaim Groups in that Endurance Group.

Each Endurance Group is composed of storage media, which are termed Media Units (refer to section 8.1.4.2) or Reclaim Units (refer to section 3.2.4). Reclaim Unit Handles reference a Reclaim Unit in each Reclaim group for writing user data. For clarity, Media Units, Reclaim Unit Handles, and Reclaim Units are not shown in the examples in this section that follow.

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

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

**Figure 11: Simple NVM Storage Hierarchy with NVM Sets**

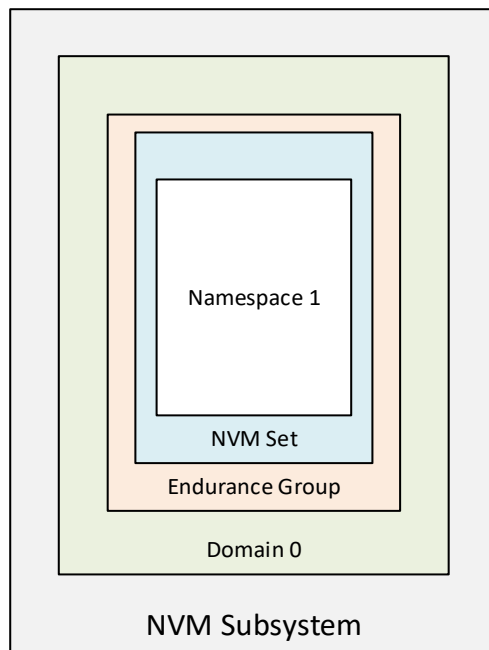


Figure 12 shows the hierarchical relationships in a simple NVM subsystem, which has:

- one domain;
- one Endurance Group;
- one Reclaim Group; and
- one namespace with user data written to the single Reclaim Group.

The placement (i.e., which Reclaim Group) of user data for a namespace is directed by each host write command to that namespace (refer to section 8.1.10).

**Figure 12: Simple NVM Storage Hierarchy with One Reclaim Group**

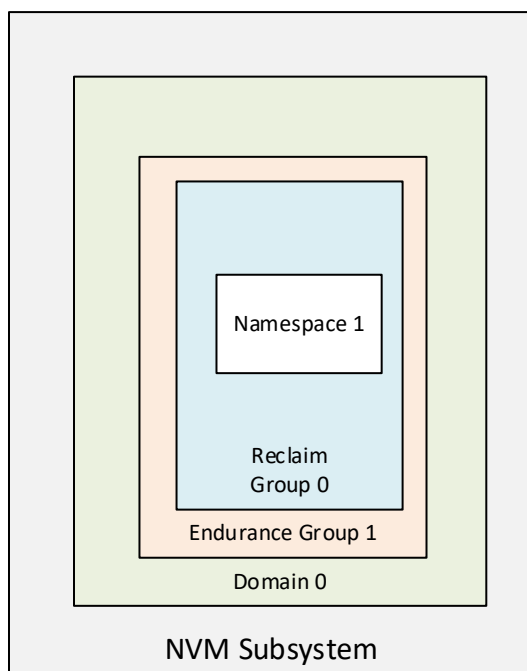


Figure 13 shows the hierarchical relationships in a simple NVM subsystem, which has:

- one domain;
- one Endurance Group;
- four Reclaim Groups; and
- one namespace with user data written to each Reclaim Group.

The placement (i.e., which Reclaim Group) of user data for a namespace is directed by each host write command to that namespace (refer to section 8.1.10).

**Figure 13: Simple NVM Storage Hierarchy with Multiple Reclaim Groups**

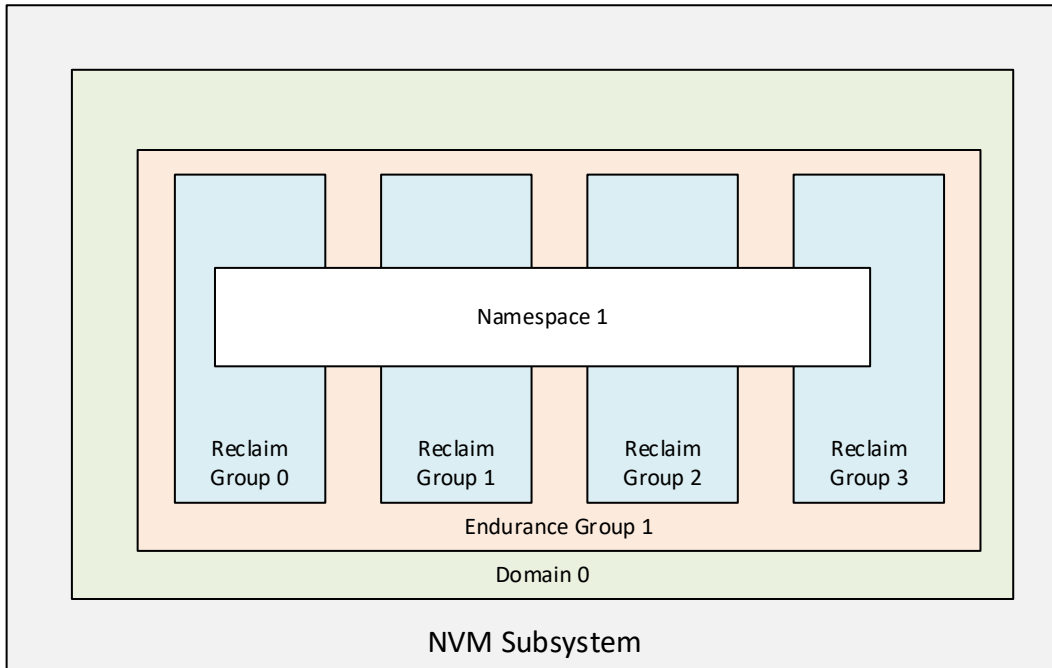
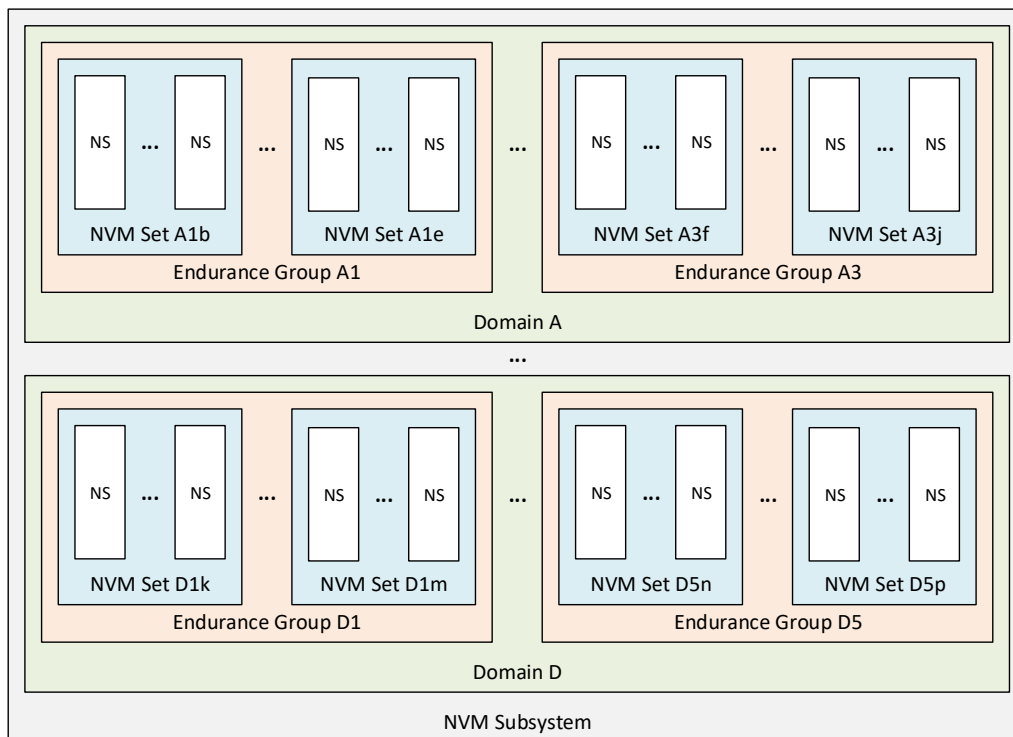


Figure 14 shows the relationships of these entities in a complex NVM subsystem, which has:

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

**Figure 14: Complex NVM Storage Hierarchy with NVM Sets**



Entity naming key (Abc):

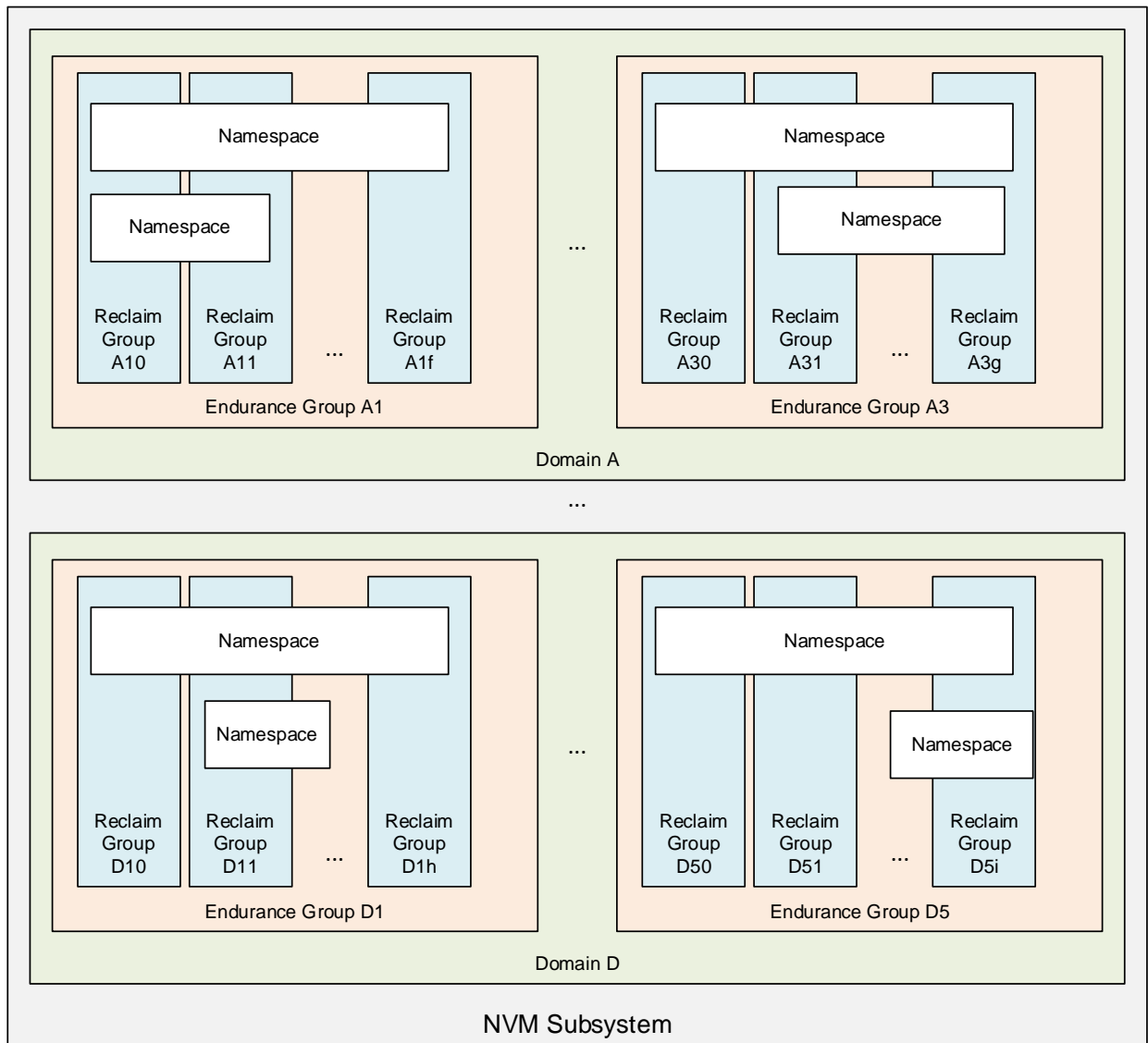
- A: Domain (capital letter)
- b: Endurance Group (digit)
- c: NVM Set (lower case letter)

Figure 15 shows the relationships in a complex NVM subsystem, which has:

- multiple domains;
- multiple Endurance Groups per domain;
- multiple Reclaim Groups per Endurance Group; and
- multiple namespaces per Endurance Group.

The placement (i.e., which Reclaim Group) of user data for a namespace is directed by each host write command to that namespace (refer to section 8.1.10).

**Figure 15: Complex NVM Storage Hierarchy with Multiple Reclaim Groups**



Entity naming key (Abc):

- A: Domain (capital letter)
- b: Endurance Group (digit)
- c: Reclaim Group (digit or lower-case letter for maximum number)

The support of Endurance Groups, Reclaim Groups within an Endurance Group, or NVM Sets within an Endurance Group is optional, but the storage model supports these concepts. An NVM subsystem may be shipped by the vendor with storage entities configured, or an NVM subsystem may be configured or re-configured by the customer. Typical changes to the configuration are creation and deletion of namespaces.

An NVM subsystem that does not support multiple NVM Sets does not require reporting of NVM Sets. An NVM subsystem that does not support multiple Endurance Groups does not require reporting of Endurance Groups.

### 2.3.2 I/O Command Sets

I/O commands perform operations on namespaces, and each namespace is associated with exactly one I/O command set. For example, commands in the NVM Command Set access data represented in a namespace as logical blocks, and commands in the Key Value Command Set access data represented in a namespace as key-value pairs.

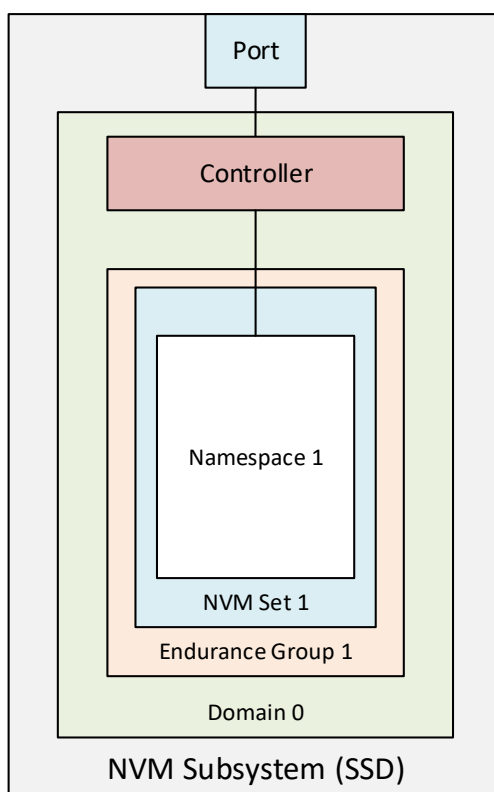
The association of a namespace to an I/O command set is specified when the namespace is created and is fixed for the lifetime of that namespace.

A controller may support one or more I/O command sets. Namespaces that are associated with the I/O command sets that are supported and enabled on a controller may be attached to that controller. A host issues commands to a namespace and those commands are interpreted based on the I/O command set associated with that namespace.

### 2.3.3 NVM Subsystem Examples

Figure 16 illustrates a simple NVM subsystem that has a single instance of each storage entity.

**Figure 16: Single-Namespace NVM Subsystem**

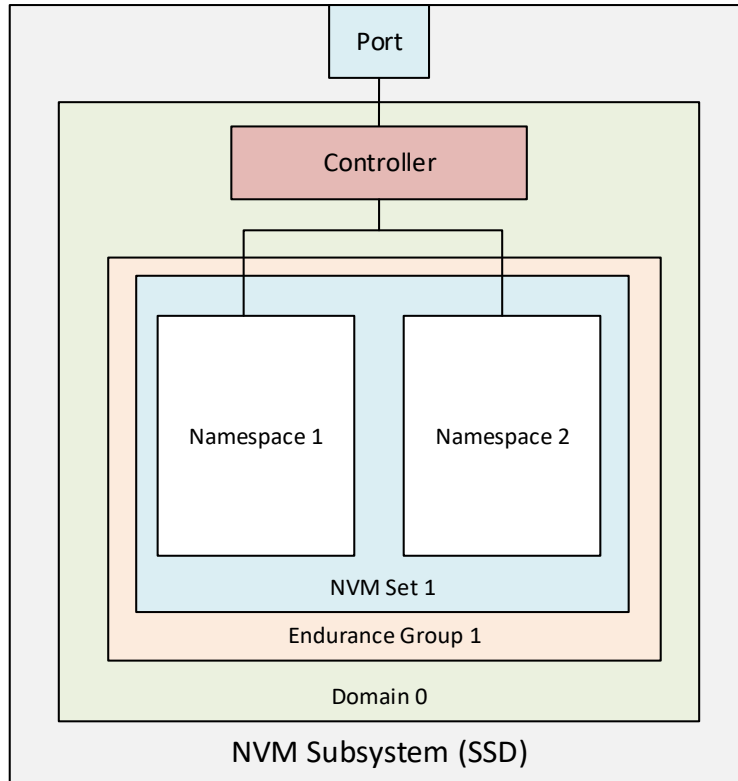


- The NVM subsystem consists of a single port and a single domain.
- The domain contains a controller and storage media.
- All of the storage media are contained in one Endurance Group.
- All of the storage media in that Endurance Group are organized into one NVM Set.
- That NVM Set contains a single namespace.



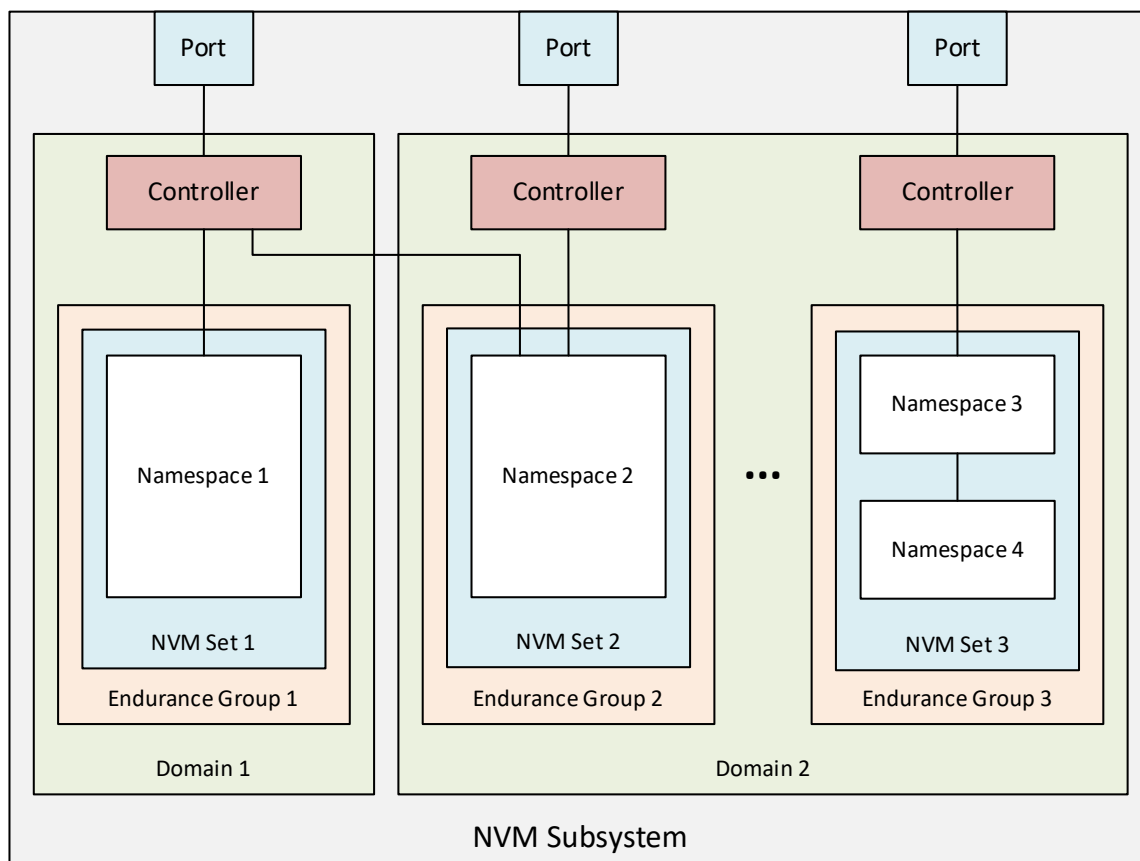
Figure 17 shows an NVM subsystem with two namespaces.

**Figure 17: Two-Namespace NVM Subsystem**



An NVM subsystem may have multiple domains, multiple namespaces, multiple controllers, and multiple ports, as shown in Figure 18.

**Figure 18: Complex NVM Subsystem**



## 2.4 Extended Capabilities Theory

### 2.4.1 Multi-Path I/O and Namespace Sharing

This section provides an overview of multi-path I/O and namespace sharing. Multi-path I/O refers to two or more completely independent paths between a single host and a namespace while namespace sharing refers to the ability for two or more hosts to access a common shared namespace using different NVM Express controllers. Both multi-path I/O and namespace sharing require that the NVM subsystem contain two or more controllers. NVM subsystems that support Multi-Path I/O and Namespace Sharing may also support asymmetric controller behavior (refer to section 2.4.2). Concurrent access to a shared namespace by two or more hosts requires some form of coordination between hosts. The procedure used to coordinate these hosts is outside the scope of this specification.

Figure 19 shows an NVM subsystem that contains a single NVM Express controller implemented over PCI Express and a single PCI Express port. Since this is a single Function PCI Express device, the NVM Express controller shall be associated with PCI Function 0. A controller may support multiple namespaces. The controller in Figure 19 supports two namespaces labeled NS A and NS B. Associated with each controller namespace is a namespace ID, labeled as NSID 1 and NSID 2, that is used by the controller to reference a specific namespace. The namespace ID is distinct from the namespace itself and is the handle a host and controller use to specify a particular namespace in a command. The selection of a controller's namespace IDs is outside the scope of this specification. In this example, NSID 1 is associated with

namespace A and NSID 2 is associated with namespace B. Both namespaces are private to the controller and this configuration supports neither multi-path I/O nor namespace sharing.

**Figure 19: NVM Express Controller with Two Namespaces**

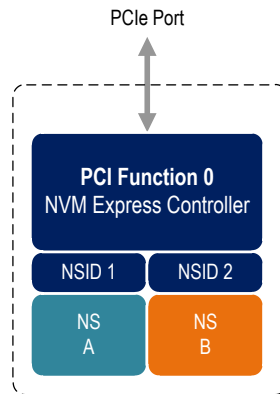
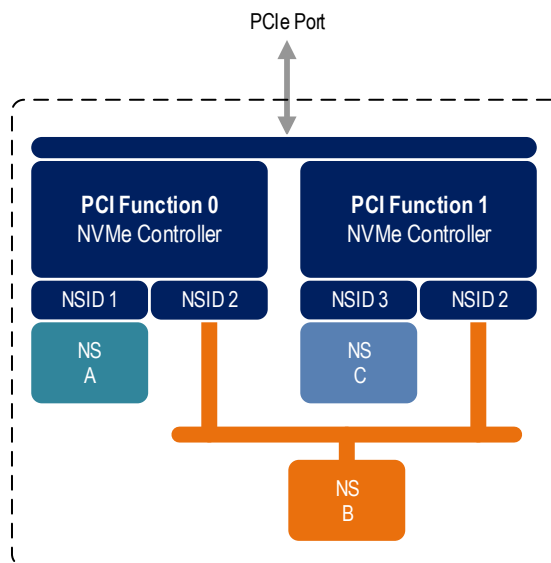


Figure 20 shows a multi-Function NVM subsystem with a single PCI Express port containing two controllers implementing NVMe over PCIe. One controller is associated with PCI Function 0 and the other controller is associated with PCI Function 1. Each controller supports a single private namespace and access to shared namespace B. The namespace ID shall be the same in all controllers that have access to a particular shared namespace. In this example, both controllers use NSID 2 to access shared namespace B.

**Figure 20: NVM Subsystem with Two Controllers and One Port**



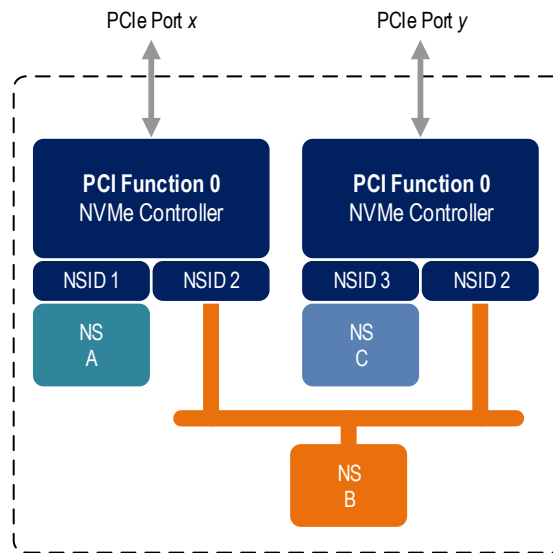
There is one or more Identify Controller data structures for each controller and one or more Identify Namespace data structures (refer to section 1.5.49) for each namespace (refer to Figure 310). Controllers with access to a shared namespace return the Identify Namespace data structure associated with that shared namespace (i.e., the same data structure contents are returned by all controllers with access to the same shared namespace). There is a globally unique identifier (refer to section 4.7.1) associated with the namespace itself and may be used to determine when there are multiple paths to the same shared namespace.

Controllers associated with a shared namespace may operate on the namespace concurrently. Operations performed by individual controllers are atomic to the shared namespace at the write atomicity level of the

controller to which the command was submitted (refer to section 3.4.3). The write atomicity level is not required to be the same across controllers that share a namespace. If there are any ordering requirements between commands issued to different controllers that access a shared namespace, then host software or an associated application, is required to enforce these ordering requirements.

Figure 21 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 4.4 for Feature behavior on reset. The functional behavior of this example is otherwise the same as that illustrated in Figure 20.

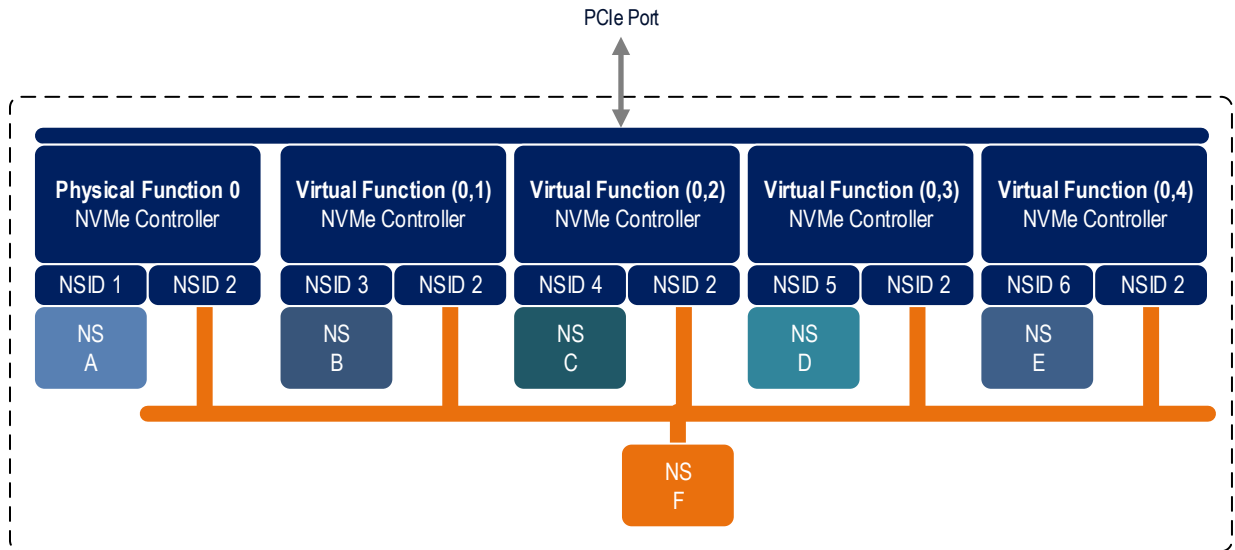
**Figure 21: NVM Subsystem with Two Controllers and Two Ports**



The two ports shown in Figure 21 may be associated with the same Root Complex or with different Root Complexes and may be used to implement both multi-path I/O and I/O sharing architectures. System-level architectural aspects and use of multiple ports in a PCI Express fabric are beyond the scope of this specification.

Figure 22 illustrates an NVM subsystem that supports Single Root I/O Virtualization (SR-IOV) and has one Physical Function and four Virtual Functions. An NVMe Express controller implementing NVMe over PCIe is associated with each Function with each controller having a private namespace and access to a namespace shared by all controllers, labeled NS F. The behavior of the controllers in this example parallels that of the other examples in this section. Refer to section 8.2.6.4 for more information on SR-IOV.

**Figure 22: PCI Express Device Supporting Single Root I/O Virtualization (SR-IOV)**



Examples provided in this section are meant to illustrate concepts and are not intended to enumerate all possible configurations. For example, an NVM subsystem may contain multiple PCI Express ports with each port supporting SR-IOV.

### 2.4.2 Asymmetric Controller Behavior

Asymmetric controller behavior occurs in NVM subsystems where namespace access characteristics (e.g., performance) may vary based on:

- the internal configuration of the NVM subsystem; or
- which controller is used to access a namespace (e.g., Fabrics).

NVM subsystems that provide asymmetric controller behavior may support Asymmetric Namespace Access Reporting as described in section 8.1.1.

## 3 NVM Express Architecture

### 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 Memory-Based Controller Architecture (PCIe)

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

#### 3.1.2 Message-Based Controller Architecture (Fabrics)

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 controller that 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 546).

In an NVM subsystem that supports the dynamic controller model, 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.

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.

While an association exists between a host and a controller, only that host may establish connections with I/O Queues of that controller. To establish a new connection with I/O Queues of that controller, the host sends subsequent Connect commands using the same NVM subsystem port, NVMe Transport type, and NVMe Transport address and specifies the:

- same Host NQN;
- same NVM Subsystem NQN;
- same Controller ID; and
- either the:
  - same Host Identifier; or
  - a Host Identifier value of 0h, if supported (refer to section 5.1.12.3.1).

An association between a host and controller is terminated if:

- the controller is shutdown as described in section 3.6.2;
- a Controller Level Reset occurs;
- the NVMe Transport connection is lost between the host and controller for the Admin Queue; or
- an NVMe Transport connection is lost between the host and controller for any I/O Queue and the host or controller does not support individual I/O Queue deletion (refer to section 3.3.2.4).

There is no explicit NVMe command that breaks the NVMe Transport association between a host and controller. The Disconnect command (refer to section 6.4) provides a method to delete an I/O Queue (refer to section 3.3.2.4). While a controller is associated with a host, that controller is busy, and no other associations may be made with that controller.

To use the dynamic controller model, the host specifies a controller identifier of FFFFh when using the Fabrics Connect command (refer to section 6.3) to establish an association with an NVM subsystem.

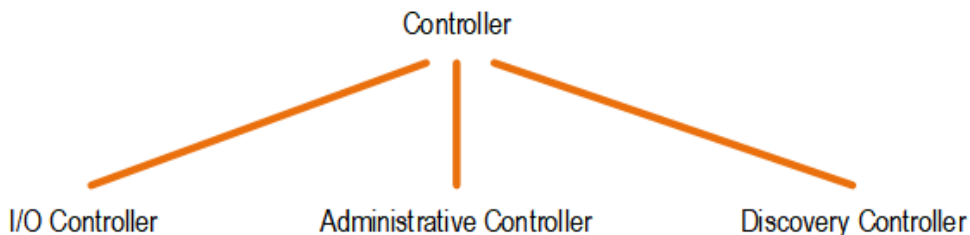
When using the static controller model with a fabric connected controller, the state that persists across associations is any state that persists across a Controller Level Reset. Additionally, different controllers may present different Feature settings or namespace attachments to the same host. The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

While allocation of static controllers to hosts are expected to be durable (so that hosts can expect to form associations to the same controllers repeatedly (e.g., after each host reboot)), the NVM subsystem may remove the host allocation of a controller that is not in use at any time for implementation specific reasons (e.g., controller resource reclamation, subsystem reconfiguration).

### 3.1.3 Controller Types

As shown in Figure 23, there are three types of controllers. An I/O controller (refer to section 3.1.3.1) is a controller that supports commands that provide access to user data stored on an NVM subsystem’s non-volatile storage medium and may support commands that provide management capabilities. An Administrative controller (refer to section 3.1.3.2) is a controller that supports commands that provide management capabilities, but does not support I/O commands that access to user data stored on an NVM subsystem’s non-volatile storage medium. A Discovery controller (refer to section 3.1.3.3) is a controller used in NVMe over Fabrics to provide access to a Discovery log page.

**Figure 23: Controller Types**



The Controller Type (CNTRLTYPE) field in the Identify Controller data structure indicates a controller’s type. Regardless of controller type, all controllers implement one Admin Submission Queue and one Admin Completion Queue. Depending on the controller type, a controller may also support one or more I/O Submission Queues and I/O Completion Queues.

When using a memory-based transport implementation (e.g., PCIe), host software submits commands to a controller through pre-allocated Submission Queues. A controller is alerted to newly submitted commands through SQ Tail Doorbell register writes. The difference between the previous doorbell register value and the current register write indicates the number of commands that were submitted.

A controller fetches commands from the Submission Queue(s) and processes them. Except for fused operations, there are no ordering restrictions for processing of commands within or across Submission Queues. Host software should not submit commands to a Submission Queue that may not be re-ordered

arbitrarily by the controller. Data associated with the processing of a command may or may not be committed to the NVM subsystem non-volatile storage medium in the order that commands are submitted.

Host software submits commands of higher priorities to the appropriate Submission Queues. Priority is associated with the Submission Queue itself, thus the priority of the command is based on the Submission Queue to which that command was submitted. The controller arbitrates across the Submission Queues based on fairness and priority according to the arbitration scheme specified in section 3.4.4.

Upon completion of the command execution by the NVM subsystem, the controller presents completion queue entries to the host through the appropriate Completion Queues. Transport specific methods (e.g., PCIe interrupts) are used to notify the host of completion queue entries to process (refer to the appropriate Transport specification).

There are no ordering restrictions for completions to the host. Each completion queue entry identifies the Submission Queue Identifier and Command Identifier of the associated command. Host software uses this information to correlate the completions with the commands submitted to the Submission Queue(s).

Host software is responsible for creating I/O Submission Queues and I/O Completion Queues prior to using those queue pairs to submit commands to the controller. I/O Submission Queues and I/O Completion Queues are created using the Create I/O Submission Queue command (refer to section 5.2.2) and the Create I/O Completion Queue command (refer to section 5.2.1).

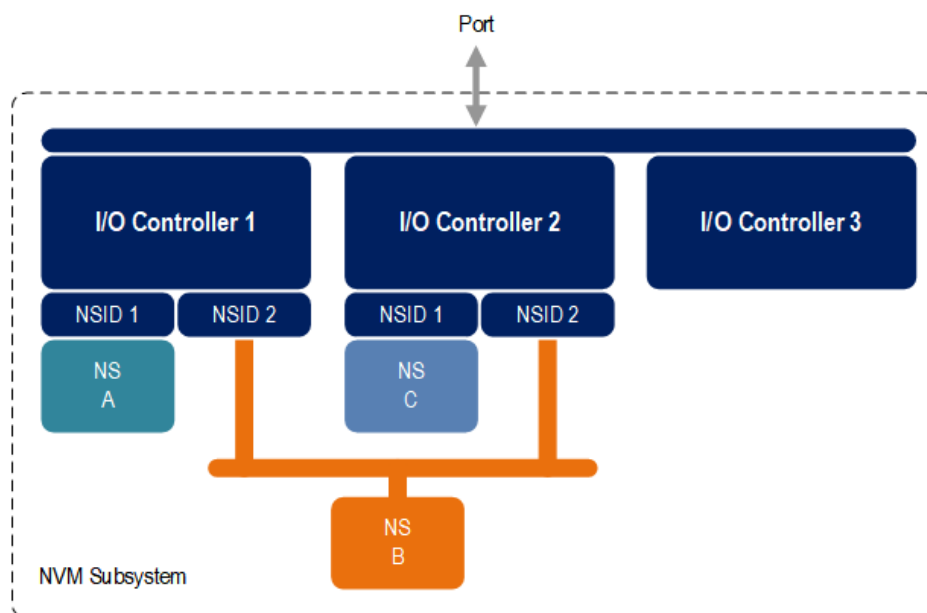
### 3.1.3.1 I/O Controller

An I/O controller is a controller that supports commands that provide access to user data stored on an NVM subsystem's non-volatile storage medium using an I/O command set and may support commands that provide management capabilities.

An I/O controller may simultaneously support multiple I/O Command Sets. The I/O Command Sets that the controller supports and which of these I/O Command Sets the controller simultaneously supports is reported in the Identify I/O Command Set data structure (refer to section 5.1.13.2.19). The contents of the Identify I/O Command Set data structure are not required to be the same for all controllers in an NVM subsystem.

Figure 24 shows an NVM subsystem with three I/O controllers. I/O controller one has two attached namespaces, private namespace A and shared namespace B. I/O controller two also has two attached namespaces, private namespace C and shared namespace B. I/O controller three has no attached namespaces. At some later point in time shared namespace B may be attached to I/O controller three.

**Figure 24: NVM Subsystem with Three I/O Controllers**





### 3.1.3.2 Administrative Controller

An Administrative controller is a controller whose intended purpose is to provide NVM subsystem management capabilities. While an I/O controller may support these same management capabilities, an Administrative controller has fewer mandatory capabilities. Unlike an I/O controller, an Administrative controller does not support I/O commands that access to user data stored on an NVM subsystem's non-volatile storage medium. NVMe Transports may support a transport specific mechanism to allow an Administrative controller to load a dedicated NVMe management driver instead of a generic NVMe driver (refer to the applicable NVMe Transport binding specification for details).

Examples of management capabilities that may be supported by an Administrative controller include the following.

- Ability to efficiently poll NVM subsystem health status via NVMe-MI using the NVMe-MI Send command and the NVMe-MI Receive command (refer to the NVM Subsystem Health Status Poll section in the NVM Express Management Interface Specification);
- Ability to manage an NVMe enclosure via NVMe-MI using the NVMe-MI Send command and the NVMe-MI Receive command;
- Ability to manage NVM subsystem namespaces using the Namespace Attachment command and the Namespace Management command;
- Ability to perform virtualization management using the Virtualization Management command;
- Ability to reset an entire NVM subsystem using the NVM Subsystem Reset (NSSR) register if supported; and
- Ability to shutdown an entire NVM subsystem using the NVM Subsystem Shutdown (NSSD) property, if supported.

An Administrative controller shall not support I/O Queues. Namespaces shall not be attached to an Administrative controller.

An Administrative controller is required to support the mandatory Admin commands listed in Figure 28. An Administrative controller may support one or more I/O Command Sets. If an Administrative controller supports an I/O Command Set, then only I/O Command Set specific Admin commands may be supported since an Administrative controller only has an Admin Queue and no I/O Queues.

Figure 25 shows an NVM subsystem with one Administrative controller and two I/O controllers within an NVM subsystem that contains a non-volatile storage medium and namespaces. I/O controller one has two attached namespaces, private namespace A and shared namespace B. I/O controller two also has two attached namespaces, private namespace C and shared namespace B. An Administrative controller has no attached namespaces. The Administrative controller in this example may be used for tasks such as NVM subsystem namespace management and efficiently polling NVM subsystem health status via NVMe-MI. While this example shows a single Administrative controller, an NVM subsystem may support zero or more Administrative controllers.

**Figure 25: NVM Subsystem with One Administrative and Two I/O Controllers**

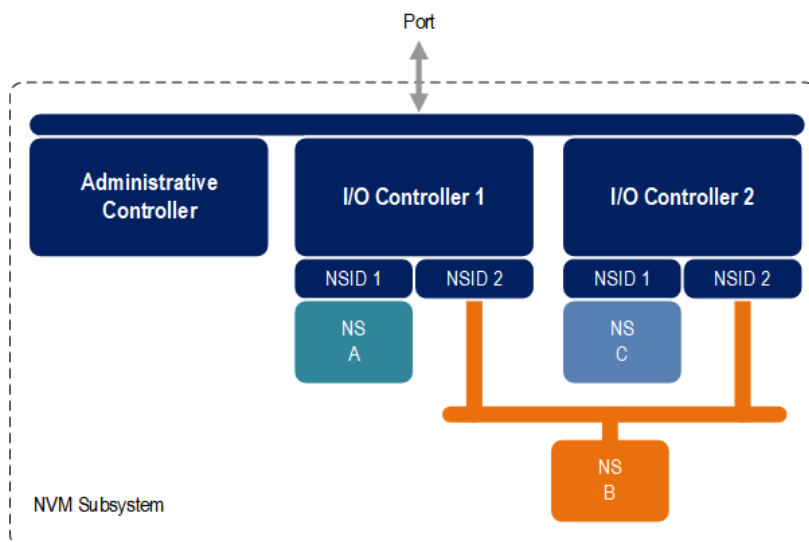
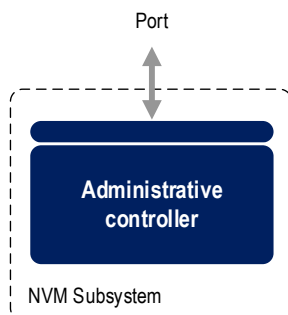


Figure 26 shows an NVM subsystem with one Administrative controller within an NVM subsystem that contains no non-volatile storage medium or namespaces. The Administrative controller in this example may be used to manage an NVMe enclosure using NVMe-MI. Since the Administrative controller is used for a very specific dedicated purpose, the implementer of such an Administrative controller may choose to implement only the mandatory capabilities along with the NVMe-MI Send and NVMe-MI Receive commands.

**Figure 26: NVM Subsystem with One Administrative Controller**



### 3.1.3.3 Discovery Controller

A Discovery controller is a controller used in NVMe over Fabrics. A Discovery controller enables a host to discover other NVM subsystems or other Discovery subsystems. A Discovery controller only implements features related to Discovering other subsystems and does not implement I/O Queues, I/O commands, or expose namespaces. The features supported by the Discovery controller are defined in section 3.1.3.6.

If the Discovery subsystem provides a unique Discovery Service NQN (i.e., the NVM Subsystem NVMe Qualified Name (SUBNQN) field in that Discovery subsystem's Identify Controller data structure contains a unique Discovery Service NQN value), then that Discovery subsystem shall support both the unique Discovery Service NQN and the well-known Discovery Service NQN (i.e., nqn.2014-08.org.nvmeexpress.discovery) being specified in the Connect command (refer to section 6.3) from the host.

If the Discovery subsystem does not provide a unique Discovery Service NQN (i.e., the SUBNQN field in that Discovery subsystem's Identify Controller data structure contains the well-known Discovery Service NQN), then that Discovery subsystem shall support the well-known Discovery Service NQN being specified in the Connect command from the host.

In the Connect command to a Discovery subsystem that provides a unique Discovery Service NQN, the host may use either of the following:

- the well-known Discovery Service NQN; or
- the unique Discovery Service NQN of that Discovery subsystem.

In the Connect command to a Discovery subsystem that does not provide a unique Discovery Service NQN, the host uses the well-known Discovery Service NQN.

The method that a host uses to obtain the fabric information necessary to connect to a Discovery controller using the well-known Discovery Service NQN or the unique NQN via the NVMe Transport may be:

- a) implementation specific;
- b) fabric specific;
- c) known in advance (e.g., a well-known address);
- d) administratively configured; or
- e) for IP-based fabrics, Automated Discovery of Discovery Controllers for IP-based Fabrics (refer to section 8.3.1) may be used.

The Discovery log page provided by a Discovery controller contains one or more entries. Each entry specifies information necessary for the host to connect to an NVM subsystem. An entry may be associated with an NVM subsystem that exposes namespaces or a referral to another Discovery Service. There are no ordering requirements for log page entries within the Discovery log page.

Discovery controller(s) may provide different log page contents depending on the Host NQN provided (e.g., different NVM subsystems may be accessible to different hosts). The set of Discovery Log Page Entries should include all applicable addresses on the same fabric as the Discovery Service and may include addresses on other fabrics.

Discovery controllers that support explicit persistent connections shall support both the Asynchronous Event Request command and the Keep Alive command (refer to sections 5.1.2 and 5.1.14 respectively). A host requests an explicit persistent connection to a Discovery controller and Asynchronous Event Notifications from the Discovery controller on that persistent connection by specifying a non-zero Keep Alive Timer value in the Connect command. If the Connect command specifies a non-zero Keep Alive Timer value and the Discovery controller does not support Asynchronous Events, then the Discovery controller shall return a status value of Connect Invalid Parameters (refer to Figure 548) for the Connect command. Discovery controllers shall indicate support for Discovery Log Change Notifications in the Identify Controller data structure (refer to Figure 312).

Discovery controllers that do not support explicit persistent connections shall not support Keep Alive commands and may use a fixed Discovery controller activity timeout value (e.g., 2 minutes). If no commands are received by such a Discovery controller within that time period, the controller may perform the actions for Keep Alive Timer expiration defined in section 3.9.5.

A Discovery controller shall not support the Disconnect command.

A Discovery log page with multiple Discovery Log Page Entries for the same NVM subsystem indicates that there are multiple fabric paths to the NVM subsystem, and/or that multiple static controllers may share a fabric path. The host may use this information to form multiple associations to controllers within an NVM subsystem.

Multiple Discovery Log Page Entries for the same NVM subsystem with different Port ID values indicates that the resulting NVMe Transport connections are independent with respect to NVM subsystem port hardware failures. A host that uses a single association should pick a record to attach to an NVM subsystem. A host that uses multiple associations should choose different ports.

A transport specific method may exist to indicate changes to a Discovery controller.

Controller IDs in the range FFF0h to FFFFh are not allocated as valid Controller IDs on completion of a Connect command, as described in section 6.3. Figure 27 defines these Controller IDs.

**Figure 27: Controller IDs FFF0h to FFFFh**

Controller ID	Definition
FFF0h to FFFCh	Reserved. Use of this value in a Connect command results in a status code of Connect Invalid Parameters being returned, as described in section 6.3.
FFFDh	This value in the Controller ID (CNTLID) field of the Registered Controller data structure or Registered Controller Extended data structure for a dispersed namespace indicates that the controller is not contained in the same participating NVM subsystem as the controller processing the command (refer to section 8.1.9.6). Use of this value in a Connect command results in a status code of Connect Invalid Parameters being returned, as described in section 6.3.
FFFEh	This value is sent in a Connect command to specify that any available static controller is allowed to be allocated.
FFFFh	This value is sent in a Connect command to specify that any available dynamic controller is allowed to be allocated.

The Controller ID values returned in the Discovery Log Page Entries indicate whether an NVM subsystem supports the dynamic or static controller model. The controller ID value of FFFFh is used for NVM subsystems that support the dynamic controller model indicating that any available controller may be returned. The Controller ID value of FFFEh is used for NVM subsystems that support the static controller model indicating that any available controller may be returned. An NVM subsystem supports the dynamic controller model if Discovery Log Page Entries use the Controller ID value of FFFFh. An NVM subsystem supports the static controller model if Discovery Log Page Entries use a Controller ID value that is less than FFFFh. The Identify Controller data structure also indicates whether an NVM subsystem is dynamic or static.

If an NVM subsystem implements the dynamic controller model, then multiple Discovery Log Page Entries (refer to Figure 294) with the Controller ID set to FFFFh may be returned for that NVM subsystem (e.g., to indicate multiple NVM subsystem ports) in the Discovery log page. If an NVM subsystem implements the static controller model, then multiple Discovery Log Page Entries that indicate different Controller ID values may be returned for that NVM subsystem in the Discovery log page. If an NVM subsystem that implements the static controller model includes any Discovery Log Page Entries that indicate a Controller ID of FFFEh, then the host should remember the Controller ID returned from the Fabrics Connect command and re-use the allocated Controller ID for future associations to that particular controller.

### 3.1.3.3.1 Discovery Controller Asynchronous Event Configuration

Discovery controllers that support Asynchronous Event Notifications shall implement the Get Features and Set Features commands. A Discovery controller shall enable Asynchronous Discovery Log Event Notifications, if a non-zero Keep Alive Timeout (KATO) value is received in the Connect command (refer to section 6.3) sent to that controller.

Figure 391 defines Discovery controller Asynchronous Event Notifications.

### 3.1.3.3.2 Discovery Controller Asynchronous Event Information – Requests and Notifications

If a Discovery controller detects an event about which a host has requested notification, then the Discovery controller shall send an Asynchronous Event with the:

- Asynchronous Event Type field set to Notice (i.e., 2h);
- Log Page Identifier field set to either Discovery (i.e., 70h) or Host Discovery (i.e., 71h) depending on which log page has changed; and
- Asynchronous Event Information field set as defined in Figure 151.

As a result of a Discovery controller updating Discovery log page(s), that Discovery controller shall send a Discovery Log Page Change Asynchronous Event notification (i.e., the Asynchronous Event Information field set to F0h) to each entity that has requested asynchronous event notifications of this type (refer to Figure 151).

As a result of a Discovery controller updating Host Discovery log page(s), that Discovery controller shall send a Host Discovery Log Page Change Asynchronous Event notification (i.e., the Asynchronous Event Information field set to F1h) to each entity that has requested asynchronous event notifications of this type (refer to Figure 151).

### 3.1.3.3.3 Discovery Controller Initialization

The initialization process for Discovery controllers is described in section 3.5.2.1.

### 3.1.3.4 Command Support Requirements

Figure 28 defines commands that are mandatory, optional, and prohibited for an I/O controller, Administrative controller, and Discovery controller. I/O Command Set specific command support requirements are described within individual I/O Command Set specifications. Since an Administrative controller does not support I/O queues, I/O Command Set specific commands that are not Admin commands are not supported by an Administrative controller.

A host may utilize the Commands Supported and Effects log page to determine optional commands that are supported by a controller.

**Figure 28: Admin Command Support Requirements**

Command	Combined Opcode Value	Controller Support Requirements <sup>1</sup>			Reference
		I/O	Administrative	Discovery	
Delete I/O Submission Queue	00h	M <sup>9</sup>	P	P	5.2.4
Create I/O Submission Queue	01h	M <sup>9</sup>	P	P	5.2.2
Get Log Page	02h	M	M	M	5.1.12
Delete I/O Completion Queue	04h	M <sup>9</sup>	P	P	5.2.3
Create I/O Completion Queue	05h	M <sup>9</sup>	P	P	5.2.1
Identify	06h	M	M	M	5.1.13
Abort	08h	M	O	O	5.1.1
Set Features	09h	M	O <sup>4</sup>	Note 6	5.1.25
Get Features	0Ah	M	O <sup>4</sup>	Note 6	5.1.11
Asynchronous Event Request	0Ch	M	O <sup>5</sup>	Note 6	5.1.2
Namespace Management	0Dh	O	O	P	5.1.21
Firmware Commit	10h	O	O	P	5.1.8
Firmware Image Download	11h	O	O	P	5.1.9
Device Self-test	14h	O	O	P	5.1.4
Namespace Attachment	15h	O	O	P	5.1.20
Keep Alive	18h	M <sup>2</sup>	M <sup>2</sup>	Note 6	5.1.14
Directive Send	19h	O	O	P	5.1.7
Directive Receive	1Ah	O	O	P	5.1.6
Virtualization Management	1Ch	O	O	P	5.2.6
NVMe-MI Send	1Dh	O	O	P	5.1.19
NVMe-MI Receive	1Eh	O	O	P	5.1.16
Capacity Management	20h	O	O	P	5.1.3
Discovery Information Management	21h	P	P	M <sup>7</sup>	5.3.3
Fabric Zoning Receive	22h	P	P	M <sup>7</sup>	5.3.5
Lockdown	24h	O	O	P	5.1.15
Fabric Zoning Lookup	25h	P	P	M <sup>7</sup>	5.3.4
Clear Exported NVM Resource Configuration	28h	P	O	P	5.3.1

Figure 28: Admin Command Support Requirements

Command	Combined Opcode Value	Controller Support Requirements <sup>1</sup>			Reference
		I/O	Administrative	Discovery	
Fabric Zoning Send	29h	P	P	M <sup>7</sup>	5.3.6
Create Exported NVM Subsystem	2Ah	P	O	P	5.3.2
Manage Exported NVM Subsystem	2Dh	P	O	P	5.3.8
Manage Exported Namespace	31h	P	O	P	5.3.7
Manage Exported Port	35h	P	O	P	5.3.9
Send Discovery Log Page	39h	P	P	M <sup>7</sup>	5.3.10
Track Send	3Dh	O	O	P	5.1.26
Track Receive	3Eh	O	O	P	5.1.27
Migration Send	41h	O	O	P	5.1.16
Migration Receive	42h	O	O	P	5.1.17
Controller Data Queue	45h	O	O	P	5.1.4
Doorbell Buffer Config	7Ch	O	O	P	5.2.5
<b>I/O Command Set specific Admin commands</b>					
Format NVM	80h	O	O	P	5.1.10
Security Send	81h	O	O	P	5.1.24
Security Receive	82h	O	O	P	5.1.23
Sanitize	84h	O <sup>3</sup>	O <sup>3</sup>	P	5.1.22
I/O Command Set specific Admin command	85h	Note 8	P	P	Note 8
	86h				
	88h				
	89h				
<b>Vendor Specific</b>					
Vendor Specific		O	O	O	
Notes:					
1. O/M/P definition: O = Optional, M = Mandatory, P = Prohibited					
2. Prohibited if the Keep Alive Timer feature is not supported (refer to section 3.9).					
3. Prohibited for an Exported NVM Subsystem (refer to section 8.3.3).					
4. Mandatory if any of the features in Figure 32 are implemented.					
5. Mandatory if Telemetry Log, Firmware Commit, or SMART/Health Critical Warnings are supported.					
6. For Discovery controllers that support explicit persistent connections, this command is mandatory. For Discovery controllers that do not support explicit persistent connections, this command is prohibited.					
7. Mandatory for CDCs and optional for Discovery controllers that are not a CDC.					
8. Refer to the applicable I/O Command Set specification.					
9. Mandatory for NVMe over PCIe. This command is not supported for NVMe over Fabrics.					

Figure 29: Fabrics Command Support Requirements

Command	Opcode Value	Fabrics Command Type	Controller Support Requirements <sup>1, 2</sup>			Reference
			I/O	Administrative	Discovery	
Property Set	7Fh	00h	M	M	M	6.6
Connect		01h	M	M	M	6.3
Property Get		04h	M	M	M	6.5
Authentication Send		05h	O	O	O	6.2
Authentication Receive		06h	O	O	O	6.1
Disconnect		08h	O	P	P	6.4

**Figure 29: Fabrics Command Support Requirements**

Command	Opcode Value	Fabrics Command Type	Controller Support Requirements <sup>1, 2</sup>			Reference
			I/O	Administrative	Discovery	
<b>Vendor Specific</b>						
Vendor Specific	7Fh	C0h to FFh	O	O	O	
Notes:						
1. O/M/P definition: O = Optional, M = Mandatory, P = Prohibited						
2. For NVMe over PCIe implementations, all Fabrics commands are prohibited. For NVMe over Fabrics implementations, the commands are as noted in the table.						

**Figure 30: Common I/O Command Support Requirements**

Command	Combined Opcode Value	Controller Support Requirements <sup>1</sup>			Reference
		I/O	Administrative	Discovery	
Flush	00h	M	P	P	7.2
Reservation Register	0Dh	O <sup>2</sup>	P	P	7.6
Reservation Report	0Eh	O <sup>2</sup>	P	P	7.8
Reservation Acquire	11h	O <sup>2</sup>	P	P	7.5
I/O Management Receive	12h	O <sup>3</sup>	P	P	7.3
Reservation Release	15h	O <sup>2</sup>	P	P	7.7
Cancel	18h	O	P	P	7.1
I/O Management Send	1Dh	O <sup>3</sup>	P	P	7.4
<b>Vendor Specific</b>					
Vendor Specific		O	O	O	
Notes:					
1. O/M/P definition: O = Optional, M = Mandatory, P = Prohibited					
2. Mandatory if reservations are supported as indicated in the Identify Controller data structure.					
3. Mandatory for controllers that support the Flexible Data Placement capability (refer to section 8.1.10). Refer to the FDPS bit in Figure 312.					

### 3.1.3.5 Log Page Support Requirements

Figure 31 defines log pages that are mandatory, optional, and prohibited for an I/O controller, Administrative controller, and Discovery controller. I/O Command Set specific log page support requirements are described within individual I/O Command Set specifications.

**Figure 31: Log Page Support Requirements**

Log Page Name	Log Page Identifier	Controller Support Requirements <sup>1</sup>			Reference
		I/O	Administrative	Discovery	
Supported Log Pages	00h	M <sup>3</sup>	M <sup>3</sup>	M	5.1.12.1.1
Error Information	01h	M	M	O	5.1.12.1.2
SMART / Health Information (Controller scope)	02h	M	O	P	5.1.12.1.3
SMART / Health Information (Namespace scope)	02h	O	O	P	
Firmware Slot Information	03h	M	O	P	5.1.12.1.4
Changed Attached Namespace List	04h	O	O	P	5.1.12.1.5
Commands Supported and Effects	05h	M <sup>3</sup>	M	M	5.1.12.1.6
Device Self-test	06h	O	O	P	5.1.12.1.7

Figure 31: Log Page Support Requirements

Log Page Name	Log Page Identifier	Controller Support Requirements <sup>1</sup>			Reference
		I/O	Administrative	Discovery	
Telemetry Host-Initiated	07h	O	O	P	5.1.12.1.8
Telemetry Controller-Initiated	08h	O	O <sup>8</sup>	P	5.1.12.1.9
Endurance Group Information	09h	O		P	5.1.12.1.10
Predictable Latency Per NVM Set	0Ah	O	O <sup>8</sup>	P	5.1.12.1.11
Predictable Latency Event Aggregate	0Bh	O	O <sup>8</sup>	P	5.1.12.1.12
Asymmetric Namespace Access	0Ch	O	P	P	5.1.12.1.13
Persistent Event	0Dh	O	O	P	5.1.12.1.14
LBA Status Information	0Eh	Refer to the NVM Command Set Specification			
Endurance Group Event Aggregate	0Fh	O	O	P	5.1.12.1.15
Media Unit Status	10h	O <sup>2</sup>	P	P	5.1.12.1.16
Supported Capacity Configuration List	11h	O <sup>2</sup>	P	P	5.1.12.1.17
Feature Identifiers Supported and Effects	12h	M <sup>3</sup>	M <sup>3,9</sup>	M <sup>9</sup>	5.1.12.1.18
NVMe-MI Commands Supported and Effects	13h	M <sup>3,7</sup>	M <sup>3,7</sup>	O	5.1.12.1.19
Command and Feature Lockdown	14h	O	O	P	5.1.12.1.20
Boot Partition	15h	O	O	P	5.1.12.1.21
Rotational Media Information	16h	O	P	P	5.1.12.1.22
Dispersed Namespace Participating NVM Subsystems	17h	O	O	P	5.1.12.1.23
Management Address List	18h	O	O	O	5.1.12.1.24
Physical Interface Receiver Eye Opening Measurement	19h	O <sup>4</sup>	O <sup>4</sup>	P	Note 11
Reachability Groups	1Ah	M <sup>6</sup>	P	P	5.1.12.1.25
Reachability Associations	1Bh	M <sup>6</sup>	P	P	5.1.12.1.26
Changed Allocated Namespace List	1Ch	O	O	P	5.1.12.1.27
FDP Configurations	20h	O <sup>5</sup>	P	P	5.1.12.1.28
Reclaim Unit Handle Usage	21h	O <sup>5</sup>	P	P	5.1.12.1.29
FDP Statistics	22h	O <sup>5</sup>	P	P	5.1.12.1.30
FDP Events	23h	O <sup>5</sup>	P	P	5.1.12.1.31
Discovery	70h	P	P	M	5.1.12.3.1
Host Discovery	71h	P	P	O	5.1.12.3.2
AVE Discovery	72h	P	P	O	5.1.12.3.3
Pull Model DDC Request	73h	P	P	M <sup>10</sup>	5.1.12.3.4



**Figure 31: Log Page Support Requirements**

Log Page Name	Log Page Identifier	Controller Support Requirements <sup>1</sup>			Reference
		I/O	Administrative	Discovery	
Reservation Notification	80h	O	P	P	5.1.12.1.32
Sanitize Status	81h	O	O <sup>8</sup>	P	5.1.12.1.33
Program List	82h	Refer to the Computational Programs Command Set Specification			
Downloadable Program Types List	83h				
Memory Range Set List	84h				
	85h to BEh	Refer to the applicable I/O Command Set specification			
Changed Zone List	BFh	Refer to the Zoned Namespace Command Set Specification			
Vendor Specific	C0h to FFh				
Notes:					
1. O/M/P definition: O = Optional, M = Mandatory, P = Prohibited					
2. Mandatory for controllers that support Fixed Capacity Management (refer to section 8.1.4.2).					
3. Optional for NVM Express revision 1.4 and earlier.					
4. If this log page is not described for a specific physical interface (refer to the applicable NVM Express transport specification), then this log page is prohibited for that transport.					
5. Mandatory for controllers that support the Flexible Data Placement capability (refer to section 8.1.10). Refer to the FDPS bit in Figure 312.					
6. Optional for controllers that do not support Reachability Reporting (refer to section 8.1.19).					
7. Optional if the NVMe-MI Send command and the NVMe-MI Receive command are not supported (refer to Figure 28).					
8. Prohibited for an Exported NVM subsystem (refer to section 8.3.3).					
9. Optional if the Set Features command is not supported (refer to Figure 28).					
10. Mandatory for CDCs and prohibited for Discovery controllers that are not a CDC.					
11. Refer to the applicable NVM Express transport specification.					

### 3.1.3.6 Feature Support Requirements

Figure 32 defines features that are mandatory, optional, and prohibited for an I/O controller, Administrative controller, and Discovery controller. If any feature is supported, then the Set Features command and the Get Features command shall be supported. I/O Command Set specific feature support requirements are described within individual I/O Command Set specifications.

**Figure 32: Feature Support Requirements**

Feature Name	Feature Identifier	Controller Support Requirements <sup>1</sup>			Reference
		I/O	Administrative	Discovery	
Arbitration	01h	M	P	P	5.1.25.1.1
Power Management	02h	M	O	P	5.1.25.1.2
LBA Range Type	03h	Refer to the NVM Command Set Specification			
Temperature Threshold	04h	M	O	P	5.1.25.1.3
Error Recovery	05h	Refer to the NVM Command Set Specification			
Volatile Write Cache	06h	O	P	P	5.1.25.1.4
Number of Queues	07h	M	P	P	5.1.25.2.1
Interrupt Coalescing	08h	Note 2	Note 2	P	5.1.25.2.2
Interrupt Vector Configuration	09h	Note 2	Note 2	P	5.1.25.2.3
Write Atomicity Normal	0Ah	Refer to the NVM Command Set Specification			
Asynchronous Event Configuration	0Bh	M	O <sup>8</sup>	M <sup>10</sup>	5.1.25.1.5
Autonomous Power State Transition	0Ch	O	O	P	5.1.25.1.6
Host Memory Buffer	0Dh	O	O	P	5.1.25.2.4
Timestamp	0Eh	O	O	P	5.1.25.1.7
Keep Alive Timer	0Fh	M <sup>7</sup>	M <sup>7</sup>	M <sup>10</sup>	5.1.25.1.8

Figure 32: Feature Support Requirements

Feature Name	Feature Identifier	Controller Support Requirements <sup>1</sup>			Reference
		I/O	Administrative	Discovery	
Host Controlled Thermal Management	10h	O	O	P	5.1.25.1.9
Non-Operational Power State Config	11h	O	O	P	5.1.25.1.10
Read Recovery Level Config	12h	O	O	P	5.1.25.1.11
Predictable Latency Mode Config	13h	O	O <sup>9</sup>	P	5.1.25.1.12
Predictable Latency Mode Window	14h	O	O <sup>9</sup>	P	5.1.25.1.13
LBA Status Information Report Interval	15h	Refer to the NVM Command Set Specification			
Host Behavior Support	16h	O	O	P	5.1.25.1.14
Sanitize Config	17h	O	O <sup>9</sup>	P	5.1.25.1.15
Endurance Group Event Configuration	18h	O	O <sup>9</sup>	P	5.1.25.1.16
I/O Command Set Profile	19h	O	P	P	5.1.25.1.17
Spinup Control	1Ah	O	P	P	5.1.25.1.18
Power Loss Signaling Config	1Bh	O	O	P	5.1.25.1.19
Performance Characteristics	1Ch	Refer to the NVM Command Set Specification			
Flexible Data Placement	1Dh	O <sup>6</sup>	P	P	5.1.25.1.20
Flexible Data Placement Events	1Eh	O <sup>6</sup>	P	P	5.1.25.1.21
Namespace Admin Label	1Fh	O	O	P	5.1.25.1.22
Key Value Configuration	20h	Refer to the Key Value Command Set Specification			
Controller Data Queue	21h	O	O	P	5.1.25.1.23
Embedded Management Controller Address	78h	O	O	O	5.1.25.1.24
Host Management Agent Address	79h	O	O	O	5.1.25.1.25
Enhanced Controller Metadata	7Dh	O <sup>5</sup>	O <sup>5</sup>	O	5.1.25.1.26.1
Controller Metadata	7Eh	O <sup>5</sup>	O <sup>5</sup>	O	5.1.25.1.26.2
Namespace Metadata	7Fh	O <sup>5</sup>	O <sup>5</sup>	O	5.1.25.1.26.3
Software Progress Marker	80h	O	O	P	5.1.25.1.27
Host Identifier	81h	O <sup>3</sup>	O	P	5.1.25.1.28
Reservation Notification Mask	82h	O <sup>4</sup>	P	P	5.1.25.1.29
Reservation Persistence	83h	O <sup>4</sup>	P	P	5.1.25.1.30

**Figure 32: Feature Support Requirements**

Feature Name	Feature Identifier	Controller Support Requirements <sup>1</sup>			Reference
		I/O	Administrative	Discovery	
Namespace Write Protection Config	84h	O	O	P	5.1.25.1.31
Boot Partition Write Protection Config	85h	O	O	P	5.1.25.1.32
Notes:					
<ol style="list-style-type: none"> <li>1. O/M/P definition: O = Optional, M = Mandatory, P = Prohibited</li> <li>2. Mandatory for NVMe over PCIe. This feature is not supported for NVMe over Fabrics.</li> <li>3. Mandatory if reservations are supported as indicated in the Identify Controller data structure.</li> <li>4. Mandatory if reservations are supported by the namespace as indicated by a non-zero value in the Reservation Capabilities (RESCAP) field in the Identify Namespace data structure.</li> <li>5. Optional for NVM subsystems that do not implement a Management Endpoint. For NVM subsystems that implement any Management Endpoint refer to the NVM Express Management Interface Specification.</li> <li>6. Mandatory for controllers that support the Flexible Data Placement capability (refer to section 8.1.10). Refer to the FDPS bit in Figure 312.</li> <li>7. Optional if not required by the NVMe Transport (refer to section 3.9).</li> <li>8. Mandatory if Telemetry Log, Firmware Commit or SMART/Health Critical Warnings are supported.</li> <li>9. Prohibited for an Exported NVM Subsystem (refer to section 8.3.3).</li> <li>10. For Discovery controllers that support explicit persistent connections, this feature is mandatory. For Discovery controllers that do not support explicit persistent connections, this feature is prohibited.</li> </ol>					

### 3.1.4 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.

For message-based controllers, properties may be read with the Property Get command and may be written with the Property Set command.

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

Figure 33 describes the property map for the controller.

Accesses that target any portion of two or more properties are not supported.

Software should not rely on 0h being returned.

**Figure 33: Property Definition**

Offset (OFST)	Size (in bytes)	I/O Controller <sup>1</sup>	Administrative Controller <sup>1</sup>	Discovery Controller <sup>1</sup>	Name
0h	8	M	M	M	<b>CAP:</b> Controller Capabilities
8h	4	M	M	M	<b>VS:</b> Version
Ch	4	M <sup>2</sup>	M <sup>2</sup>	R	<b>INTMS:</b> Interrupt Mask Set
Fh	4	M <sup>2</sup>	M <sup>2</sup>	R	<b>INTMC:</b> Interrupt Mask Clear
14h	4	M	M	M	<b>CC:</b> Controller Configuration
18h		R	R	R	Reserved
1Ch	4	M	M	M	<b>CSTS:</b> Controller Status
20h	4	O	O	R	<b>NSSR:</b> NVM Subsystem Reset
24h	4	M <sup>2</sup>	M <sup>2</sup>	R	<b>AQA:</b> Admin Queue Attributes
28h	8	M <sup>2</sup>	M <sup>2</sup>	R	<b>ASQ:</b> Admin Submission Queue Base Address

Figure 33: Property Definition

Offset (OFST)	Size (in bytes)	I/O Controller <sup>1</sup>	Administrative Controller <sup>1</sup>	Discovery Controller <sup>1</sup>	Name
30h	8	M <sup>2</sup>	M <sup>2</sup>	R	<b>ACQ</b> : Admin Completion Queue Base Address
38h	4	O <sup>3</sup>	O <sup>3</sup>	R	<b>CMBLOC</b> : Controller Memory Buffer Location
3Ch	4	O <sup>3</sup>	O <sup>3</sup>	R	<b>CMBSZ</b> : Controller Memory Buffer Size
40h	4	O <sup>3</sup>	O <sup>3</sup>	R	<b>BPINFO</b> : Boot Partition Information
44h	4	O <sup>3</sup>	O <sup>3</sup>	R	<b>BPRSEL</b> : Boot Partition Read Select
48h	8	O <sup>3</sup>	O <sup>3</sup>	R	<b>BPMBL</b> : Boot Partition Memory Buffer Location
50h	8	O <sup>3</sup>	O <sup>3</sup>	R	<b>CMBMSC</b> : Controller Memory Buffer Memory Space Control
58h	4	O <sup>3</sup>	O <sup>3</sup>	R	<b>CMBSTS</b> : Controller Memory Buffer Status
5Ch	4	O <sup>3</sup>	O <sup>3</sup>	R	<b>CMBEBS</b> : Controller Memory Buffer Elasticity Buffer Size
60h	4	O <sup>3</sup>	O <sup>3</sup>	R	<b>CMBSWTP</b> : Controller Memory Buffer Sustained Write Throughput
64h	4	O	O	R	<b>NSSD</b> : NVM Subsystem Shutdown
68h	4	M	M	R	<b>CRTO</b> : Controller Ready Timeouts
6Ch		R	R	R	Reserved
E00h	4	O <sup>3</sup>	O <sup>3</sup>	R	<b>PMRCAP</b> : Persistent Memory Capabilities
E04h	4	O <sup>3</sup>	O <sup>3</sup>	R	<b>PMRCTL</b> : Persistent Memory Region Control
E08h	4	O <sup>3</sup>	O <sup>3</sup>	R	<b>PMRSTS</b> : Persistent Memory Region Status
E0Ch	4	O <sup>3</sup>	O <sup>3</sup>	R	<b>PMREBS</b> : Persistent Memory Region Elasticity Buffer Size
E10h	4	O <sup>3</sup>	O <sup>3</sup>	R	<b>PMRSWTP</b> : Persistent Memory Region Sustained Write Throughput
E14h	4	O <sup>3</sup>	O <sup>3</sup>	R	<b>PMRMSCCL</b> : Persistent Memory Region Controller Memory Space Control Lower
E18h	4	O <sup>3</sup>	O <sup>3</sup>	R	<b>PMRMSCU</b> : Persistent Memory Region Controller Memory Space Control Upper
E1Ch		R	R	R	Reserved
1000h		Transport Specific: <ul style="list-style-type: none"> <li>Refer to Figure 34 for Memory-Based transport implementations.</li> <li>Refer to Figure 35 for Message-Based transport implementations.</li> </ul>			
Notes:					
1. O/M/R definition: O = Optional, M = Mandatory, R = Reserved					
2. Mandatory for memory-based controllers. For message-based controllers this property is reserved.					
3. Optional for memory-based controllers. For message-based controllers this property is reserved.					
4. Determined by the transport (e.g., the offset calculation formula Offset (1000h + ((2y) * (4 << CAP.DSTRD)))) for the memory-based PCIe transport).					

**Figure 34: Memory-Based Property Definition**

Offset (OFST)	Size (in bytes)	I/O Controller <sup>1</sup>	Admin. Controller <sup>1</sup>	Discovery Controller <sup>1</sup>	Name
1000h	Variable <sup>2</sup>	T	T	T	Transport Specific (e.g., PCIe doorbell registers as specified in the NVMe over PCIe Transport Specification)
1000h + Variable <sup>2</sup>		O	O	O	Vendor Specific

Notes:

- O/T definition: O = Optional, T = Transport Specific
- Determined by the transport (e.g., the offset calculation formula  $\text{Offset} (1000h + ((2y) * (4 \ll \text{CAP.DSTRD})))$  for the PCIe transport).

**Figure 35: Message-Based Property Definition**

Offset (OFST)	Size (in bytes)	I/O Controller <sup>1</sup>	Admin. Controller <sup>1</sup>	Discovery Controller <sup>1</sup>	Name
1000h	300h	R	R	R	Reserved for Fabrics
1300h		O	O	O	Vendor Specific

Notes:

- O/R definition: O = Optional, R = Reserved

The following conventions are used to describe controller properties for all transport models. Hardware shall return '0' for all bits that are marked as reserved, and host software shall write all reserved bits and properties with the value of 0h.

The following terms and abbreviations are used:

<b>RO</b>	Read Only
<b>RW</b>	Read Write
<b>RWC</b>	Read/Write '1' to clear
<b>RWS</b>	Read/Write '1' to set
<b>Impl Spec</b>	Implementation Specific – the controller has the freedom to choose its implementation.
<b>HwInit</b>	The default state is dependent on NVM Express controller and system configuration.
<b>Reset</b>	This column indicates the value of the field after a Controller Level Reset as defined in section 3.7.2.

For some fields, it is implementation specific as to whether the field is RW, RWC, or RO; this is typically shown as RW/RO or RWC/RO to indicate that if the functionality is not supported that the field is read only.

When a field is referred to in the document, the convention used is "Property Symbol.Field Symbol". For example, the PCI command register Parity Error Response Enable bit is referred to by the name CMD.PEE. If the field is an array of bits, the field is referred to as "Property Symbol.Field Symbol (array offset to element)". When a sub-field is referred to in the document, the convention used is "Property Symbol.Field Symbol.Sub Field Symbol". For example, when the Controller Ready With Media Support sub-field of the Controller Ready Modes Supported field within the Controller Capability property, the sub-field is referred to by the name CAP.CRMS.CRWMS.

### 3.1.4.1 Offset 0h: CAP – Controller Capabilities

This property indicates basic capabilities of the controller to host software.

Figure 36: Offset 0h: CAP – Controller Capabilities

Bits	Type	Reset	Description						
63:62	RO	0h	Reserved						
61	RO	Impl Spec	<p><b>NVM Subsystem Shutdown Enhancements Supported (NSSES):</b> This bit indicates whether the controller supports enhancements to the NVM Subsystem Shutdown feature.</p> <p>If the controller supports the enhancements to the NVM Subsystem Shutdown feature as defined in section 3.6.3, then this bit shall be set to '1' and the NSSS bit shall be set to '1'. If a controller compliant with a revision of the NVM Express Base Specification later than revision 2.0 sets the NSSS bit to '1', then that controller shall set this bit to '1'.</p> <p>If this bit is cleared to '0', then the controller does not support the enhancements to the NVM Subsystem Shutdown feature as defined in section 3.6.3.</p> <p>If the NSSRS bit is cleared to '0' or the NSSS bit is cleared to '0', then this bit shall be cleared to '0'.</p>						
60:59	RO	Impl Spec	<p><b>Controller Ready Modes Supported (CRMS):</b> This field indicates the ready capabilities of the controller. Refer to sections 3.5.3 and 3.5.4 for more detail.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td> <p><b>Controller Ready Independent of Media Support (CRIMS):</b> If this bit is set to '1', then the controller supports the Controller Ready Independent of Media mode.</p> <p>If this bit is cleared to '0', then the controller does not support Controller Ready Independent of Media mode.</p> </td> </tr> <tr> <td>0</td> <td> <p><b>Controller Ready With Media Support (CRWMS):</b> If this bit is set to '1', then the controller supports the Controller Ready With Media mode.</p> <p>If this bit is cleared to '0', then the controller does not support Controller Ready With Media mode.</p> <p>This bit shall be set to '1' on controllers compliant with NVM Express Base Specification, Revision 2.0 and later.</p> </td> </tr> </tbody> </table>	Bits	Description	1	<p><b>Controller Ready Independent of Media Support (CRIMS):</b> If this bit is set to '1', then the controller supports the Controller Ready Independent of Media mode.</p> <p>If this bit is cleared to '0', then the controller does not support Controller Ready Independent of Media mode.</p>	0	<p><b>Controller Ready With Media Support (CRWMS):</b> If this bit is set to '1', then the controller supports the Controller Ready With Media mode.</p> <p>If this bit is cleared to '0', then the controller does not support Controller Ready With Media mode.</p> <p>This bit shall be set to '1' on controllers compliant with NVM Express Base Specification, Revision 2.0 and later.</p>
Bits	Description								
1	<p><b>Controller Ready Independent of Media Support (CRIMS):</b> If this bit is set to '1', then the controller supports the Controller Ready Independent of Media mode.</p> <p>If this bit is cleared to '0', then the controller does not support Controller Ready Independent of Media mode.</p>								
0	<p><b>Controller Ready With Media Support (CRWMS):</b> If this bit is set to '1', then the controller supports the Controller Ready With Media mode.</p> <p>If this bit is cleared to '0', then the controller does not support Controller Ready With Media mode.</p> <p>This bit shall be set to '1' on controllers compliant with NVM Express Base Specification, Revision 2.0 and later.</p>								
58	RO	Impl Spec	<p><b>NVM Subsystem Shutdown Supported (NSSS):</b> This bit indicates whether the controller supports the NVM Subsystem Shutdown feature.</p> <p>If this bit is set to '1', then the controller supports the NVM Subsystem Shutdown feature. If the NSSES bit is set to '1', then this bit shall be set to '1'.</p> <p>If this bit is cleared to '0', then the controller does not support the NVM Subsystem Shutdown feature. If the NSSRS bit is cleared to '0', then this bit shall be cleared to '0'.</p> <p>Refer to section 3.6.3 for a description of the NVM Subsystem Shutdown feature and the behavioral enhancements associated with the NSSES bit being set to '1'.</p>						
57	RO	Impl Spec	<p><b>Controller Memory Buffer Supported (CMBS):</b> If this bit is set to '1', then this bit indicates that the controller supports the Controller Memory Buffer, and that addresses supplied by the host are permitted to reference the Controller Memory Buffer only if the host has enabled the Controller Memory Buffer's controller memory space.</p> <p>If the controller supports the Controller Memory Buffer, then this bit shall be set to '1'.</p>						
56	RO	Impl Spec	<p><b>Persistent Memory Region Supported (PMRS):</b> This bit indicates whether the Persistent Memory Region is supported. This bit is set to '1' if the Persistent Memory Region is supported. This bit is cleared to '0' if the Persistent Memory Region is not supported.</p>						

**Figure 36: Offset 0h: CAP – Controller Capabilities**

Bits	Type	Reset	Description										
55:52	RO	Impl Spec	<p><b>Memory Page Size Maximum (MPSMAX):</b> This field indicates the maximum host memory page size that the controller supports. The maximum memory page size is <math>(2^{(12 + MPSMAX)})</math>. The host shall not configure a memory page size in CC.MPS that is larger than this value.</p> <p>For Discovery controllers this field shall be cleared to 0h.</p>										
51:48	RO	Impl Spec	<p><b>Memory Page Size Minimum (MPSMIN):</b> This field indicates the minimum host memory page size that the controller supports. The minimum memory page size is <math>(2^{(12 + MPSMIN)})</math>. The host shall not configure a memory page size in CC.MPS that is smaller than this value.</p> <p>For Discovery controllers this shall be cleared to 0h.</p>										
47:46	RO	Impl Spec	<p><b>Controller Power Scope (CPS):</b> This field indicates scope of controlling the main power for this controller.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Power Scope</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Not Reported</td> </tr> <tr> <td>01b</td> <td>Controller scope</td> </tr> <tr> <td>10b</td> <td>Domain scope (i.e., the NVM subsystem supports multiple domains (refer to section 3.2.5).</td> </tr> <tr> <td>11b</td> <td>NVM subsystem scope (i.e., the NVM subsystem does not support multiple domains).</td> </tr> </tbody> </table> <p>If the NSSS bit is set to '1', then this field shall not be cleared to 00b.</p>	Value	Power Scope	00b	Not Reported	01b	Controller scope	10b	Domain scope (i.e., the NVM subsystem supports multiple domains (refer to section 3.2.5).	11b	NVM subsystem scope (i.e., the NVM subsystem does not support multiple domains).
Value	Power Scope												
00b	Not Reported												
01b	Controller scope												
10b	Domain scope (i.e., the NVM subsystem supports multiple domains (refer to section 3.2.5).												
11b	NVM subsystem scope (i.e., the NVM subsystem does not support multiple domains).												
45	RO	Impl Spec	<p><b>Boot Partition Support (BPS):</b> This bit indicates whether the controller supports Boot Partitions. If this bit is set to '1', then the controller supports Boot Partitions. If this bit is cleared to '0', then the controller does not support Boot Partitions. Refer to section 8.1.3.</p>										
44:37	RO	Impl Spec	<p><b>Command Sets Supported (CSS):</b> This field indicates the I/O Command Set(s) that the controller supports.</p> <p>For Discovery controllers, this field should have only the NCSS bit set to 1.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7</td> <td> <p><b>No I/O Command Set Support (NOIOCSS):</b> This bit indicates whether no I/O Command Set is supported (i.e., only the Admin Command Set is supported).</p> <p>This bit shall be set to '1' if no I/O Command Set is supported.</p> </td> </tr> <tr> <td>6</td> <td> <p><b>I/O Command Set Support (IOCSS):</b> This bit indicates whether the controller supports one or more I/O Command Sets and supports the Identify I/O Command Set data structure (refer to section 5.1.13.2.19). Controllers that support I/O Command Sets other than the NVM Command Set shall set this bit to '1'. Controllers that only support the NVM Command Set may set this bit to '1'.</p> </td> </tr> <tr> <td>5:1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td> <p><b>NVM Command Set Support (NCSS):</b> This bit indicates whether the controller supports the NVM Command Set or a Discovery controller.</p> </td> </tr> </tbody> </table>	Bits	Description	7	<p><b>No I/O Command Set Support (NOIOCSS):</b> This bit indicates whether no I/O Command Set is supported (i.e., only the Admin Command Set is supported).</p> <p>This bit shall be set to '1' if no I/O Command Set is supported.</p>	6	<p><b>I/O Command Set Support (IOCSS):</b> This bit indicates whether the controller supports one or more I/O Command Sets and supports the Identify I/O Command Set data structure (refer to section 5.1.13.2.19). Controllers that support I/O Command Sets other than the NVM Command Set shall set this bit to '1'. Controllers that only support the NVM Command Set may set this bit to '1'.</p>	5:1	Reserved	0	<p><b>NVM Command Set Support (NCSS):</b> This bit indicates whether the controller supports the NVM Command Set or a Discovery controller.</p>
Bits	Description												
7	<p><b>No I/O Command Set Support (NOIOCSS):</b> This bit indicates whether no I/O Command Set is supported (i.e., only the Admin Command Set is supported).</p> <p>This bit shall be set to '1' if no I/O Command Set is supported.</p>												
6	<p><b>I/O Command Set Support (IOCSS):</b> This bit indicates whether the controller supports one or more I/O Command Sets and supports the Identify I/O Command Set data structure (refer to section 5.1.13.2.19). Controllers that support I/O Command Sets other than the NVM Command Set shall set this bit to '1'. Controllers that only support the NVM Command Set may set this bit to '1'.</p>												
5:1	Reserved												
0	<p><b>NVM Command Set Support (NCSS):</b> This bit indicates whether the controller supports the NVM Command Set or a Discovery controller.</p>												
36	RO	Impl Spec	<p><b>NVM Subsystem Reset Supported (NSSRS):</b> This bit indicates whether the controller supports the NVM Subsystem Reset feature defined in section 3.7.1. This bit is set to '1' if the controller supports the NVM Subsystem Reset feature. This bit is cleared to '0' if the controller does not support the NVM Subsystem Reset feature.</p> <p>For Discovery controllers, this field shall be cleared to 0h.</p>										

Figure 36: Offset 0h: CAP – Controller Capabilities

Bits	Type	Reset	Description
35:32	RO	Impl Spec	<p><b>Doorbell Stride (DSTRD):</b> Each Submission Queue and Completion Queue Doorbell property is 32-bits in size. This property indicates the stride between doorbell properties. The stride is specified as <math>(2 \wedge (2 + DSTRD))</math> in bytes. A value of 0h indicates a stride of 4 bytes, where the doorbell properties are packed without reserved space between each property. Refer to section 8.2.2.</p> <p>For NVMe over Fabrics I/O controllers, this property shall be cleared to a fixed value of 0h.</p>
31:24	RO	Impl Spec	<p><b>Timeout (TO):</b> This is the worst-case time that host software should wait for the CSTS.RDY bit to transition from:</p> <ul style="list-style-type: none"> <li>a) '0' to '1' after the CC.EN bit transitions from '0' to '1'; or</li> <li>b) '1' to '0' after the CC.EN bit transitions from '1' to '0'.</li> </ul> <p>This worst-case time may be experienced after events such as an abrupt shutdown, loss of main power without shutting down the controller, or activation of a new firmware image; typical times are expected to be much shorter.</p> <p>This field is in 500 millisecond units. The maximum value of this field is FFh, which indicates a 127.5 second timeout.</p> <p>If the Controller Ready Independent of Media Enable (CC.CRIME) bit is cleared to '0' and the worst-case time for the CSTS.RDY bit to change state is due to enabling the controller after the CC.EN bit transitions from '0' to '1', then this field shall be set to:</p> <ul style="list-style-type: none"> <li>a) the value in the Controller Ready With Media Timeout (CRTO.CRWMT) field; or</li> <li>b) FFh if the value in the CRTO.CRWMT field is greater than FFh.</li> </ul> <p>If the Controller Ready Independent of Media Enable (CC.CRIME) bit is set to '1' and the worst-case time for the CSTS.RDY bit to change state is due to enabling the controller after the CC.EN bit transitions from '0' to '1', then this field shall be set to:</p> <ul style="list-style-type: none"> <li>a) the value in the Controller Ready Independent of Media Timeout (CRTO.CRIMT); or</li> <li>b) FFh if the value in the CRTO.CRIMT field is greater than FFh.</li> </ul> <p>Controllers that support the CRTO register (refer to Figure 57) are able to indicate larger timeouts for enabling the controller. Host software should use the value in the CRTO.CRWMT field or the CRTO.CRIMT field depending on the controller ready mode indicated by the CC.CRIME bit to determine the worst-case timeout for the CSTS.RDY bit to transition from '0' to '1' after the CC.EN bit transitions from '0' to '1'. Host software that is based on revisions earlier than NVM Express Base Specification, Revision 2.0 is not required to wait for more than 127.5 seconds for the CSTS.RDY bit to transition.</p> <p>Refer to sections 3.5.3 and 3.5.4 for more information.</p>
23:19	RO	0h	Reserved



**Figure 36: Offset 0h: CAP – Controller Capabilities**

Bits	Type	Reset	Description						
18:17	RO	Impl Spec	<p><b>Arbitration Mechanism Supported (AMS):</b> This field is bit significant and indicates the optional arbitration mechanisms supported by the controller. If a bit is set to '1', then the corresponding arbitration mechanism is supported by the controller. Refer to section 3.4.4 for arbitration details.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td><b>Vendor Specific (VS):</b> Vendor Specific arbitration mechanism.</td> </tr> <tr> <td>0</td> <td><b>Weighted Round Robin with Urgent Priority Class (WRRUPC):</b> Weighted Round Robin with Urgent Priority Class arbitration mechanism.</td> </tr> </tbody> </table> <p>The round robin arbitration mechanism is not listed since all controllers shall support this arbitration mechanism.</p> <p>For Discovery controllers, this property shall be cleared to 0h.</p>	Bits	Description	1	<b>Vendor Specific (VS):</b> Vendor Specific arbitration mechanism.	0	<b>Weighted Round Robin with Urgent Priority Class (WRRUPC):</b> Weighted Round Robin with Urgent Priority Class arbitration mechanism.
			Bits	Description					
1	<b>Vendor Specific (VS):</b> Vendor Specific arbitration mechanism.								
0	<b>Weighted Round Robin with Urgent Priority Class (WRRUPC):</b> Weighted Round Robin with Urgent Priority Class arbitration mechanism.								
16	RO	Impl Spec	<p><b>Contiguous Queues Required (CQR):</b> 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 controllers using a message-based transport, this property shall be set to a value of 1.</p>						
15:00	RO	Impl Spec	<p><b>Maximum Queue Entries Supported (MQES):</b> This field indicates the maximum individual queue size that the controller supports. For NVMe over PCIe implementations, this value applies to the I/O Submission Queues and I/O Completion Queues that the host creates. For NVMe over Fabrics implementations, this value applies to only the I/O Submission Queues that the host creates. This is a 0's based value. The minimum value is 1h, indicating two entries.</p>						

**3.1.4.2 Offset 8h: VS – Version**

This property is Read Only (RO) and indicates the version of this specification that the controller supports, as defined in Figure 37.

**Figure 37: Specification Version Descriptor**

Bits	Description
31:16	<b>Major Version (MJR):</b> An integer value indicating the major version number of this specification which is supported by the controller.
15:08	<b>Minor Version (MNR):</b> An integer value indicating the minor version number of this specification which is supported by the controller.
07:00	<b>Tertiary Version (TER):</b> An integer value indicating the tertiary version number of this specification which is supported by the controller. If this field is cleared to 0h, then this specification does not have a tertiary version number.

The reset value for each field is described in Figure 38:

**Figure 38: NVM Express Base Specification Version Property Reset Values**

Specification Version <sup>1</sup>	MJR Field	MNR Field	TER Field
1.0	1h	0h	0h

**Figure 38: NVM Express Base Specification Version Property  
Reset Values**

Specification Version 1	MJR Field	MNR Field	TER Field
1.1	1h	1h	0h
1.2	1h	2h	0h
1.2.1	1h	2h	1h
1.3	1h	3h	0h
1.4	1h	4h	0h
2.0	2h	0h	0h
2.1	2h	1h	0h

Notes:  
1. The specification version listed includes lettered versions (e.g., 1.4 includes 1.4, 1.4a through 1.4c, etc.).

### 3.1.4.3 Offset Ch: INTMS – Interrupt Mask Set

This property is used to mask interrupts when using pin-based interrupts, single message MSI, or multiple message MSI. When using MSI-X, the interrupt mask table defined as part of MSI-X should be used to mask interrupts. Host software shall not access this property when configured for MSI-X; any accesses when configured for MSI-X is undefined. For interrupt behavior requirements, refer to the Interrupts section of the NVMe over PCIe Transport Specification.

**Figure 39: Offset Ch: INTMS – Interrupt Mask Set**

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

### 3.1.4.4 Offset 10h: INTMC – Interrupt Mask Clear

This property is used to unmask interrupts when using pin-based interrupts, single message MSI, or multiple message MSI. When using MSI-X, the interrupt mask table defined as part of MSI-X should be used to unmask interrupts. Host software shall not access this property when configured for MSI-X; any accesses when configured for MSI-X is undefined. For interrupt behavior requirements, refer to the Interrupts section of the NVMe over PCIe Transport Specification.

**Figure 40: Offset 10h: INTMC – Interrupt Mask Clear**

Bits	Type	Reset	Description
31:00	RWC	0h	<b>Interrupt Vector Mask Clear (IVMC):</b> This field is bit significant. If a '1' is written to a bit, then the corresponding interrupt vector is unmasked. Writing a '0' to a bit has no effect. When read, this field returns the current interrupt mask value within the controller (not the value of this property). If a bit has a value of a '1', then the corresponding interrupt vector is masked. If a bit has a value of '0', then the corresponding interrupt vector is not masked.

### 3.1.4.5 Offset 14h: CC – Controller Configuration

This property modifies settings for the controller. Host software shall set the Arbitration Mechanism Selected (CC.AMS), the Memory Page Size (CC.MPS), and the I/O Command Set Selected (CC.CSS) to valid values

prior to enabling the controller by setting CC.EN to '1'. Attempting to create an I/O queue before initializing the I/O Completion Queue Entry Size (CC.IOCQES) and the I/O Submission Queue Entry Size (CC.IOSQES) shall cause a controller to abort a Create I/O Completion Queue command or a Create I/O Submission Queue command with a status code of Invalid Queue Size.

**Figure 41: Offset 14h: CC – Controller Configuration**

Bits	Type	Reset	Description						
31:25	RO	0h	Reserved						
24	RW/RO	0b	<p><b>Controller Ready Independent of Media Enable (CRIME):</b> This bit controls the controller ready mode. The controller ready mode is determined by the state of this bit at the time the controller is enabled by transitioning the CC.EN bit from '0' to '1'.</p> <p>If the CAP.CRMS field is set to 11b, then this bit is RW. If the CAP.CRMS field is not set to 11b, then this bit is RO and shall be cleared to '0'. Refer to sections 3.5.3 and 3.5.4 for more detail.</p> <p>Changing the value of this field may cause a change in the time reported in the CAP.TO field. Refer to the definition of CAP.TO for more details.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td><b>Controller Ready With Media Mode:</b> Enabling the controller (i.e., CC.EN transitions from '0' to '1') when this bit is cleared to '0' enables Controller Ready With Media mode.</td> </tr> <tr> <td>1b</td> <td><b>Controller Ready Independent Of Media Mode:</b> Enabling the controller when this bit is set to '1' enables Controller Ready Independent of Media mode.</td> </tr> </tbody> </table>	Value	Definition	0b	<b>Controller Ready With Media Mode:</b> Enabling the controller (i.e., CC.EN transitions from '0' to '1') when this bit is cleared to '0' enables Controller Ready With Media mode.	1b	<b>Controller Ready Independent Of Media Mode:</b> Enabling the controller when this bit is set to '1' enables Controller Ready Independent of Media mode.
Value	Definition								
0b	<b>Controller Ready With Media Mode:</b> Enabling the controller (i.e., CC.EN transitions from '0' to '1') when this bit is cleared to '0' enables Controller Ready With Media mode.								
1b	<b>Controller Ready Independent Of Media Mode:</b> Enabling the controller when this bit is set to '1' enables Controller Ready Independent of Media mode.								
23:20	RW/RO	0h	<p><b>I/O Completion Queue Entry Size (IOCQES):</b> This field defines the I/O completion queue entry size that is used for the selected I/O Command Set(s). The required and maximum values for this field are specified in the CQES field in the Identify Controller data structure in Figure 312 for each I/O Command Set. The value is in bytes and is specified as a power of two (<math>2^n</math>).</p> <p>If any I/O Completion Queues exist, then write operations that change the value in this field produce undefined results.</p> <p>If the controller does not support I/O queues, then this field shall be read-only with a value of 0h.</p> <p>For Discovery controllers, this field is reserved.</p>						
19:16	RW/RO	0h	<p><b>I/O Submission Queue Entry Size (IOSQES):</b> This field defines the I/O submission queue entry size that is used for the selected I/O Command Set(s). The required and maximum values for this field are specified in the SQES field in the Identify Controller data structure in Figure 312 for each I/O Command Set. The value is in bytes and is specified as a power of two (<math>2^n</math>).</p> <p>If any I/O Submission Queues exist, then write operations that change the value in this field produce undefined results.</p> <p>If the controller does not support I/O queues, then this field shall be read-only with a value of 0h.</p> <p>For Discovery controllers, this field is reserved.</p>						

Figure 41: Offset 14h: CC – Controller Configuration

Bits	Type	Reset	Description										
15:14	RW	00b	<p><b>Shutdown Notification (SHN):</b> This field is used to initiate a controller shutdown when a power down condition is expected. For a normal controller shutdown, it is expected that the controller is given time to process the controller shutdown. For an abrupt shutdown, the host may not wait for the controller shutdown to complete before power is lost.</p> <p>The controller shutdown notification values are defined as:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>No notification and no effect</td> </tr> <tr> <td>01b</td> <td>Normal shutdown notification</td> </tr> <tr> <td>10b</td> <td>Abrupt shutdown notification</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table> <p>This field should be written by host software prior to any power down condition and prior to any change of the PCI power management state. It is recommended that this field also be written prior to a warm reset (refer to the PCI Express Base Specification). To determine when the controller shutdown processing is complete, refer to the definition of the CSTS.ST bit and the definition of the CSTS.SHST field. Refer to section 3.6 for additional shutdown processing details.</p> <p>Other fields in this property (including the EN bit) may be modified as part of updating this field to 01b or 10b to initiate a controller shutdown. If the EN bit is cleared to '0' such that the EN bit transitions from '1' to '0', then both a Controller Reset and a controller shutdown occur.</p> <p>If an NVM Subsystem Shutdown is reported as in progress or is reported as complete (i.e., CSTS.ST is set to '1', and CSTS.SHST is set to 01b or 10b), then writes to this field modify the field value but have no effect. Refer to section 3.6.3 for details.</p>	Value	Definition	00b	No notification and no effect	01b	Normal shutdown notification	10b	Abrupt shutdown notification	11b	Reserved
Value	Definition												
00b	No notification and no effect												
01b	Normal shutdown notification												
10b	Abrupt shutdown notification												
11b	Reserved												
13:11	RW	000b	<p><b>Arbitration Mechanism Selected (AMS):</b> This field selects the arbitration mechanism to be used. This value shall only be changed when CC.EN is cleared to '0'. Host software shall only set this field to supported arbitration mechanisms indicated in CAP.AMS. If this field is set to an unsupported value, then the behavior is undefined.</p> <p>For Discovery controllers, this value shall be cleared to 0h.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>Round Robin</td> </tr> <tr> <td>001b</td> <td>Weighted Round Robin with Urgent Priority Class</td> </tr> <tr> <td>010b to 110b</td> <td>Reserved</td> </tr> <tr> <td>111b</td> <td>Vendor Specific</td> </tr> </tbody> </table>	Value	Definition	000b	Round Robin	001b	Weighted Round Robin with Urgent Priority Class	010b to 110b	Reserved	111b	Vendor Specific
Value	Definition												
000b	Round Robin												
001b	Weighted Round Robin with Urgent Priority Class												
010b to 110b	Reserved												
111b	Vendor Specific												
10:07	RW	0h	<p><b>Memory Page Size (MPS):</b> This field indicates the host memory page size. The memory page size is <math>2^{(12 + MPS)}</math>. Thus, the minimum host memory page size is 4 KiB and the maximum host memory page size is 128 MiB. The value set by host software shall be a supported value as indicated by the CAP.MPSMAX and CAP.MPSMIN fields. This field describes the value used for PRP entry size. This field shall only be modified when CC.EN is cleared to '0'.</p> <p>For Discovery controllers this property shall be cleared to 0h.</p>										

**Figure 41: Offset 14h: CC – Controller Configuration**

Bits	Type	Reset	Description																											
06:04	RW	000b	<p><b>I/O Command Set Selected (CSS):</b> This field specifies the I/O Command Set or Sets that are selected. This field shall only be changed when the controller is disabled (i.e., CC.EN is cleared to '0'). The I/O Command Set or Sets that are selected shall be used for all I/O Submission Queues.</p> <table border="1"> <thead> <tr> <th>Value</th> <th colspan="2">Definition</th> </tr> </thead> <tbody> <tr> <td rowspan="3">000b</td> <td><b>CAP.CSS.NCSS Bit</b></td> <td><b>Definition</b></td> </tr> <tr> <td>1b</td> <td>NVM Command Set</td> </tr> <tr> <td>0b</td> <td>Reserved</td> </tr> <tr> <td>001b to 101b</td> <td colspan="2">Reserved</td> </tr> <tr> <td rowspan="3">110b</td> <td><b>CAP.CSS.IOCSS Bit</b></td> <td><b>Definition</b></td> </tr> <tr> <td>1b</td> <td>All Supported I/O Command Sets The I/O Command Sets that are supported are reported in the Identify I/O Command Set data structure (refer to section 5.1.13.2.19).</td> </tr> <tr> <td>0b</td> <td>Reserved</td> </tr> <tr> <td rowspan="3">111b</td> <td><b>CAP.CSS.NOIOCSS Bit</b></td> <td><b>Definition</b></td> </tr> <tr> <td>1b</td> <td>Admin Command Set only I/O Command Set and I/O Command Set specific Admin commands are not supported. Any I/O Command Set specific Admin command submitted on the Admin Submission Queue is aborted with a status code of Invalid Command Opcode.</td> </tr> <tr> <td>0b</td> <td>Reserved</td> </tr> </tbody> </table> <p>For Discovery controllers, this property shall be cleared to 000b.</p>	Value	Definition		000b	<b>CAP.CSS.NCSS Bit</b>	<b>Definition</b>	1b	NVM Command Set	0b	Reserved	001b to 101b	Reserved		110b	<b>CAP.CSS.IOCSS Bit</b>	<b>Definition</b>	1b	All Supported I/O Command Sets The I/O Command Sets that are supported are reported in the Identify I/O Command Set data structure (refer to section 5.1.13.2.19).	0b	Reserved	111b	<b>CAP.CSS.NOIOCSS Bit</b>	<b>Definition</b>	1b	Admin Command Set only I/O Command Set and I/O Command Set specific Admin commands are not supported. Any I/O Command Set specific Admin command submitted on the Admin Submission Queue is aborted with a status code of Invalid Command Opcode.	0b	Reserved
Value	Definition																													
000b	<b>CAP.CSS.NCSS Bit</b>	<b>Definition</b>																												
	1b	NVM Command Set																												
	0b	Reserved																												
001b to 101b	Reserved																													
110b	<b>CAP.CSS.IOCSS Bit</b>	<b>Definition</b>																												
	1b	All Supported I/O Command Sets The I/O Command Sets that are supported are reported in the Identify I/O Command Set data structure (refer to section 5.1.13.2.19).																												
	0b	Reserved																												
111b	<b>CAP.CSS.NOIOCSS Bit</b>	<b>Definition</b>																												
	1b	Admin Command Set only I/O Command Set and I/O Command Set specific Admin commands are not supported. Any I/O Command Set specific Admin command submitted on the Admin Submission Queue is aborted with a status code of Invalid Command Opcode.																												
	0b	Reserved																												
03:01	RO	000b	Reserved																											

Figure 41: Offset 14h: CC – Controller Configuration

Bits	Type	Reset	Description
00	RW	0b	<p><b>Enable (EN):</b> When set to '1', then the controller shall process commands. When cleared to '0', then the controller shall not process commands nor post completion queue entries to Completion Queues. When the host modifies CC to clear this bit from '1' to '0', the controller is reset (i.e., a Controller Reset, refer to section 3.7.2). That reset deletes all I/O Submission Queues and I/O Completion Queues, resets the Admin Submission Queue and the Admin Completion Queue, and brings the hardware to an idle state. That reset does not affect transport specific state (e.g., PCI Express registers including MMIO MSI-X registers), nor the Admin Queue properties (AQA, ASQ, or ACQ). Refer to section 3.7.2 for the effects of that reset on all controller properties. Internal controller state (e.g., Feature values defined in section 5.1.25 that are not persistent across power states) are reset to their default values. The controller shall ensure that there is no impact (e.g., data loss) caused by that Controller Reset to the results of commands that have had corresponding completion queue entries posted to an I/O Completion Queue prior to that Controller Reset.</p> <p>When this bit is cleared to '0', the CSTS.RDY bit is cleared to '0' by the controller once the controller is ready to be re-enabled. When this bit is set to '1', the controller sets the CSTS.RDY bit to '1' when it is ready to process commands. The CSTS.RDY bit may be set to '1' before namespace(s) are ready to be accessed.</p> <p>Setting this bit from a '0' to a '1' when the CSTS.RDY bit is a '1' or clearing this bit from a '1' to a '0' when the CSTS.RDY bit is cleared to '0' has undefined results. The Admin Queue properties (AQA, ASQ, and ACQ) are only allowed to be modified when this bit is cleared to '0'.</p> <p>If an NVM Subsystem Shutdown is reported as in progress or is reported as completed (i.e., the CSTS.ST bit is set to '1', and the CSTS.SHST field is set to 01b or 10b), then:</p> <ul style="list-style-type: none"> <li>• setting this bit from '0' to '1' modifies the field value but has no effect (e.g., the controller does not respond by setting the CSTS.RDY bit to '1'); and</li> <li>• clearing this bit from '1' to '0' resets the controller as defined by this field.</li> </ul> <p>Refer to section 3.6.3 for details on NVM Subsystem Shutdown functionality.</p>

## 3.1.4.6 Offset 1Ch: CSTS – Controller Status

Figure 42: Offset 1Ch: CSTS – Controller Status

Bits	Type	Reset <sup>1</sup>	Description
31:07	RO	0h	Reserved
06	RO	Hwlnit	<p><b>Shutdown Type (ST):</b> If CSTS.SHST is set to a non-zero value, then this bit indicates the type of shutdown reported by CSTS.SHST.</p> <p>If this bit is set to '1', then CSTS.SHST is reporting the state of an NVM Subsystem Shutdown and this bit remains set to '1' until an NVM Subsystem Reset occurs.</p> <p>If this bit is cleared to '0', then CSTS.SHST is reporting the state of a controller shutdown.</p> <p>An NVM Subsystem Reset shall clear this bit to '0'. All other Controller Level Resets shall not change the value of this bit.</p> <p>If CSTS.SHST is cleared to 00b, then this bit should be ignored by the host.</p>

**Figure 42: Offset 1Ch: CSTS – Controller Status**

Bits	Type	Reset <sup>1</sup>	Description										
05	RO	0b	<p><b>Processing Paused (PP):</b> This bit indicates whether the controller is processing commands. If this bit is cleared to '0', then the controller is processing commands normally. If this bit is set to '1', then the controller has temporarily stopped processing commands in order to handle an event (e.g., firmware activation). This bit is only valid when CC.EN is set to '1' and CSTS.RDY is set to '1'.</p>										
04	RWC	HwInit	<p><b>NVM Subsystem Reset Occurred (NSSRO):</b> The initial value of this bit is set to '1' if the last occurrence of an NVM Subsystem Reset (refer to section 3.7.1) occurred while power was applied to the domain. The initial value of this bit is cleared to '0' following an NVM Subsystem Reset due to application of power to the domain. This bit is only valid if the controller supports the NVM Subsystem Reset feature defined in section 3.7.1 as indicated by CAP.NSSRS set to '1'.</p> <p>The reset value of this bit is cleared to '0' if an NVM Subsystem Reset causes activation of a new firmware image in the domain.</p>										
03:02	RO	00b	<p><b>Shutdown Status (SHST):</b> This field indicates the status of shutdown processing that is initiated by the host setting the CC.SHN field, the host setting the NSSD.NSSC field, or a Management Endpoint processing an NVMe-MI Shutdown command (refer to the NVM Express Management Interface Specification). Shutdown processing is able to occur on this controller as a consequence of a host setting the NSSD.NSSC field on another controller to initiate an NVM Subsystem Shutdown that affects this controller.</p> <p>The shutdown status values are defined as:</p> <table border="1" style="margin-left: 40px;"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Normal operation (no shutdown has been requested)</td> </tr> <tr> <td>01b</td> <td>Shutdown processing in progress</td> </tr> <tr> <td>10b</td> <td>Shutdown processing complete</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table> <p>If this field is set to 01b (i.e., shutdown processing in progress), then:</p> <ul style="list-style-type: none"> <li>• an NVM Subsystem Reset aborts both a controller shutdown and an NVM Subsystem Shutdown; and</li> <li>• any other type of Controller Level Reset (CLR): <ul style="list-style-type: none"> <li>○ may or may not abort a controller shutdown; and</li> <li>○ shall not abort an NVM Subsystem Shutdown.</li> </ul> </li> </ul> <p>If this field is cleared to 00b (i.e., normal operation) when a CLR is initiated, then that CLR shall not change the value of this field.</p> <p>If this field is set to 01b when a CLR is initiated, and shutdown processing is not aborted by that CLR, then that CLR shall not change the value of this field.</p> <p>If this field is set to 01b when a CLR is initiated and shutdown processing is aborted by that CLR, then that CLR shall clear this field to 00b.</p> <p>If this field is set to 10b (i.e., shutdown processing complete) when a CLR is initiated by NVM Subsystem Reset, then that CLR shall clear this field to 00b.</p> <p>If this field is set to 10b when a CLR is initiated by a method other than NVM Subsystem Reset and:</p> <ul style="list-style-type: none"> <li>• the CSTS.ST bit is set to '1', then that CLR shall not change the value of this field; and</li> <li>• the CSTS.ST bit is cleared to '0', then that CLR shall clear this field to 00b</li> </ul> <p>If the CSTS.ST bit is cleared to '0' and this field is set to 10b (i.e., controller shutdown processing is reported as complete), then to start executing commands on the controller:</p>	Value	Definition	00b	Normal operation (no shutdown has been requested)	01b	Shutdown processing in progress	10b	Shutdown processing complete	11b	Reserved
Value	Definition												
00b	Normal operation (no shutdown has been requested)												
01b	Shutdown processing in progress												
10b	Shutdown processing complete												
11b	Reserved												

Figure 42: Offset 1Ch: CSTS – Controller Status

Bits	Type	Reset <sup>1</sup>	Description
			<ul style="list-style-type: none"> <li>if the CC.EN bit is set to '1', after a shutdown operation then a CLR (e.g., a Controller Reset) followed by enabling the controller (i.e., host sets the CC.EN bit from '0' to '1') is required (refer to section 3.6.1). If a host submits commands to the controller without a prior CLR, then the behavior is undefined; and</li> <li>if the CC.EN bit is cleared to '0', then: <ul style="list-style-type: none"> <li>a CLR followed by enabling the controller is required (refer to sections 3.6.1 and 3.6.2); or</li> <li>the CC.EN bit is required to be set to '1' and the CC.SHN bit is required to be cleared to 00b with the same write to the CC property (refer to sections 3.6.1 and 3.6.2).</li> </ul> </li> </ul> <p>If the CSTS.ST bit is set to '1' and this field is set to 10b (i.e., NVM Subsystem Shutdown processing is reported as complete), then an NVM Subsystem Reset followed by enabling the controller is required to start executing commands (refer to section 3.6.3). If a host submits commands to the controller without a prior NVM Subsystem Reset, then the behavior is undefined.</p>
01	RO	Hwlnit	<p><b>Controller Fatal Status (CFS):</b> This bit is set to '1' when a fatal controller error occurred that could not be communicated in the appropriate Completion Queue. This bit is cleared to '0' when a fatal controller error has not occurred. Refer to section 9.5.</p> <p>The reset value of this bit is set to '1' when a fatal controller error is detected during controller initialization.</p>
00	RO	0b	<p><b>Ready (RDY):</b> This bit is set to '1' when the controller is ready to process submission queue entries after the CC.EN bit is set to '1'. This bit shall be cleared to '0' when the CC.EN bit is cleared to '0' once the controller is ready to be re-enabled. Commands should not be submitted to the controller until this bit is set to '1' after the CC.EN bit is set to '1'. Failure to follow this recommendation produces undefined results. Refer to the definition of the CAP.TO field, section 3.5.3, and section 3.5.4 for timing information related to this field.</p> <p>If an NVM Subsystem Shutdown that affects this controller is reported as in progress or is reported as complete (i.e., the CSTS.ST bit is set to '1' and the CSTS.SHST field is set to 01b or is set to 10b), then an NVM Subsystem Reset is required before this bit is allowed to be set to '1' from '0'. Refer to section 3.6.3.</p> <p>If a controller shutdown is reported as in progress or is reported as complete (i.e., the CSTS.ST bit is cleared to '0' and the CSTS.SHST field is set to 01b or is set to 10b), then before this bit is allowed to be set to '1' from '0', controller shutdown processing shall stop (e.g., complete or be terminated) and the CSTS.SHST field shall be cleared to 00b.</p>
Notes:			
1. During a Controller Level Reset, the field and bit values may transition to values other than the reset value prior to indicating the reset value.			



### 3.1.4.7 Offset 20h: NSSR – NVM Subsystem Reset

This optional property provides host software with the capability to initiate an NVM Subsystem Reset. Support for this property is indicated by the state of the NVM Subsystem Reset Supported (CAP.NSSRS) field. If the property is not supported, then the address range occupied by the property is reserved. Refer to section 3.7.1.

**Figure 43: Offset 20h: NSSR – NVM Subsystem Reset**

Bits	Type	Reset	Description
31:00	RW	0h	<b>NVM Subsystem Reset Control (NSSRC):</b> A write of the value 4E564D65h ("NVMe") to this field initiates an NVM Subsystem Reset. A write of any other value has no functional effect on the operation of the NVM subsystem. This field shall return the value 0h when read.

### 3.1.4.8 Offset 24h: AQA – Admin Queue Attributes

This property defines the attributes for the Admin Submission Queue and Admin Completion Queue. The Queue Identifier for the Admin Submission Queue and Admin Completion Queue is 0h. The Admin Submission Queue's priority is determined by the arbitration mechanism selected, refer to section 3.4.4. The Admin Submission Queue and Admin Completion Queue are required to be in physically contiguous memory.

This property shall not be reset by Controller Reset.

Note: It is recommended that the host use UEFI during boot operations. In low memory environments (e.g., Option ROMs in legacy BIOS environments) there may not be sufficient available memory to allocate the necessary Submission and Completion Queues. In these types of conditions, low memory operation of the controller is vendor specific.

**Figure 44: Offset 24h: AQA – Admin Queue Attributes**

Bits	Type	Reset	Description
31:28	RO	0h	Reserved
27:16	RW	0h	<b>Admin Completion Queue Size (ACQS):</b> Defines the size of the Admin Completion Queue in entries. Refer to section 3.3.3.1. Enabling a controller while this field is cleared to 0h produces undefined results. The minimum size of the Admin Completion Queue is two entries. The maximum size of the Admin Completion Queue is 4,096 entries. This is a 0's based value.
15:12	RO	0h	Reserved
11:00	RW	0h	<b>Admin Submission Queue Size (ASQS):</b> Defines the size of the Admin Submission Queue in entries. Refer to section 3.3.3.1. Enabling a controller while this field is cleared to 0h produces undefined results. The minimum size of the Admin Submission Queue is two entries. The maximum size of the Admin Submission Queue is 4,096 entries. This is a 0's based value.

### 3.1.4.9 Offset 28h: ASQ – Admin Submission Queue Base Address

This property defines the base memory address of the Admin Submission Queue.

This property shall not be reset by Controller Reset.

**Figure 45: Offset 28h: ASQ – Admin Submission Queue Base Address**

Bits	Type	Reset	Description
63:12	RW	Impl Spec	<b>Admin Submission Queue Base (ASQB):</b> This field specifies the 52 most significant bits of the 64-bit physical address for the Admin Submission Queue. This address shall be memory page aligned (based on the value in CC.MPS). All Admin commands, including creation of I/O Submission Queues and I/O Completions Queues shall be submitted to this queue. For the definition of Submission Queues, refer to section 4.1.

**Figure 45: Offset 28h: ASQ – Admin Submission Queue Base Address**

Bits	Type	Reset	Description
11:00	RO	0h	Reserved

**3.1.4.10 Offset 30h: ACQ – Admin Completion Queue Base Address**

This property defines the base memory address of the Admin Completion Queue.

This property shall not be reset by Controller Reset.

**Figure 46: Offset 30h: ACQ – Admin Completion Queue Base Address**

Bits	Type	Reset	Description
63:12	RW	Impl Spec	<b>Admin Completion Queue Base (ACQB):</b> This field specifies the 52 most significant bits of the 64-bit physical address for the Admin Completion Queue. This address shall be memory page aligned (based on the value in CC.MPS). All completion queue entries for the commands submitted to the Admin Submission Queue shall be posted to this Completion Queue. This queue is always associated with interrupt vector 0. For the definition of Completion Queues, refer to section 4.1.
11:00	RO	0h	Reserved

**3.1.4.11 Offset 38h: CMBLOC – Controller Memory Buffer Location**

This optional property defines the location of the Controller Memory Buffer (refer to section 8.2.1). If the controller does not support the Controller Memory Buffer (CAP.CMBS), this property is reserved. If the controller supports the Controller Memory Buffer and CMBMSC.CRE is cleared to '0', this property shall be cleared to 0h.

**Figure 47: Offset 38h: CMBLOC – Controller Memory Buffer Location**

Bits	Type	Reset	Description
31:12	RO	Impl Spec	<b>Offset (OFST):</b> Indicates the offset of the Controller Memory Buffer in multiples of the Size Unit specified in CMBSZ.
11:09	RO	000b	Reserved
08	RO	Impl Spec	<b>CMB Queue Dword Alignment (CQDA):</b> If this bit is set to '1', CDW11.PC is set to '1'; and the address pointer specifies Controller Memory Buffer, then the address pointer in a Create I/O Submission Queue command (refer to Figure 477) or a Create I/O Completion Queue command (refer to Figure 473) shall be Dword aligned.  If this bit is cleared to '0', then the I/O Submission Queues and I/O Completion Queues contained in the Controller Memory Buffer are aligned as defined by the PRP1 field of a Create I/O Submission Queue command (refer to Figure 477) or a Create I/O Completion Queue command (refer to Figure 473).
07	RO	Impl Spec	<b>CMB Data Metadata Mixed Memory Support (CDMMMS):</b> If this bit is set to '1', then the restriction on data and metadata use of Controller Memory Buffer by a command as defined in section 8.2.1 is not enforced. If this bit is cleared to '0', then the restriction on data and metadata use of Controller Memory Buffer by a command as defined in section 8.2.1 is enforced.
06	RO	Impl Spec	<b>CMB Data Pointer and Command Independent Locations Support (CDPCILS):</b> If this bit is set to '1', then the restriction that the PRP Lists and SGLs shall not be located in the Controller Memory Buffer if the command that they are associated with is not located in the Controller Memory Buffer is not enforced (refer to section 8.2.1). If this bit is cleared to '0', then that restriction is enforced.
05	RO	Impl Spec	<b>CMB Data Pointer Mixed Locations Support (CDPMLS):</b> If this bit is set to '1', then the restriction that for a particular PRP List or SGL associated with a single command, all memory that contains that particular PRP List or SGL shall reside in either the Controller Memory Buffer or outside the Controller Memory Buffer, is not enforced (refer to section 8.2.1). If this bit is cleared to '0', then that restriction is enforced.

**Figure 47: Offset 38h: CMBLOC – Controller Memory Buffer Location**

Bits	Type	Reset	Description
04	RO	Impl Spec	<b>CMB Queue Physically Discontiguous Support (CQPDS):</b> If this bit is set to '1', then the restriction that for all queues in the Controller Memory Buffer, the queue shall be physically contiguous, is not enforced (refer to section 8.2.1). If this bit is cleared to '0', then that restriction is enforced.
03	RO	Impl Spec	<b>CMB Queue Mixed Memory Support (CQMMS):</b> If this bit is set to '1', then for a particular queue placed in the Controller Memory Buffer, the restriction that all memory associated with that queue shall reside in the Controller Memory Buffer is not enforced (refer to section 8.2.1). If this bit is cleared to '0', then that requirement is enforced.
02:00	RO	Impl Spec	<b>Base Indicator Register (BIR):</b> Indicates the Base Address Register (BAR) that contains the Controller Memory Buffer. For a 64-bit BAR, the BAR for the least significant 32-bits of the address is specified. Values 000b, 010b, 011b, 100b, and 101b are valid. The address specified by the BAR shall be 4 KiB aligned.

### 3.1.4.12 Offset 3Ch: CMBSZ – Controller Memory Buffer Size

This optional property defines the size of the Controller Memory Buffer (refer to section 8.2.1). If the controller does not support the Controller Memory Buffer feature or if the controller supports the Controller Memory Buffer (CAP.CMBS) and CMBMSC.CRE is cleared to '0', then this property shall be cleared to 0h.

**Figure 48: Offset 3Ch: CMBSZ – Controller Memory Buffer Size**

Bits	Type	Reset	Description		
31:12	RO	Impl Spec	<b>Size (SZ):</b> Indicates the size of the Controller Memory Buffer available for use by the host. The size is in multiples of the Size Unit. If the Offset + Size exceeds the length of the indicated BAR, the size available to the host is limited by the length of the BAR.		
11:08	RO	Impl Spec	<b>Size Units (SZU):</b> Indicates the granularity of the Size field.		
				<b>Value</b>	<b>Granularity</b>
				0h	4 KiB
				1h	64 KiB
				2h	1 MiB
				3h	16 MiB
				4h	256 MiB
				5h	4 GiB
6h	64 GiB				
7h to Fh	Reserved				
07:05	RO	000b	Reserved		
04	RO	Impl Spec	<b>Write Data Support (WDS):</b> If this bit is set to '1', then the controller supports data and metadata in the Controller Memory Buffer for commands that transfer data from the host to the controller (e.g., Write). If this bit is cleared to '0', then data and metadata for commands that transfer data from the host to the controller shall not be transferred to the Controller Memory Buffer.		
03	RO	Impl Spec	<b>Read Data Support (RDS):</b> If this bit is set to '1', then the controller supports data and metadata in the Controller Memory Buffer for commands that transfer data from the controller to the host (e.g., Read). If this bit is cleared to '0', then data and metadata for commands that transfer data from the controller to the host shall not be transferred from the Controller Memory Buffer.		
02	RO	Impl Spec	<b>PRP SGL List Support (LISTS):</b> If this bit is set to '1', then the controller supports PRP Lists in the Controller Memory Buffer. If this bit is set to '1' and SGLs are supported by the controller, then the controller supports Scatter Gather Lists in the Controller Memory Buffer. If this bit is set to '1', then the Submission Queue Support bit shall be set to '1'. If this bit is cleared to '0', then PRP Lists and SGLs shall not be placed in the Controller Memory Buffer.		
01	RO	Impl Spec	<b>Completion Queue Support (CQS):</b> If this bit is set to '1', then the controller supports Admin and I/O Completion Queues in the Controller Memory Buffer. If this bit is cleared to '0', then Completion Queues shall not be placed in the Controller Memory Buffer.		

**Figure 48: Offset 3Ch: CMBSZ – Controller Memory Buffer Size**

Bits	Type	Reset	Description
00	RO	Impl Spec	<b>Submission Queue Support (SQS):</b> If this bit is set to '1', then the controller supports Admin and I/O Submission Queues in the Controller Memory Buffer. If this bit is cleared to '0', then Submission Queues shall not be placed in the Controller Memory Buffer.

**3.1.4.13 Offset 40h: BPINFO – Boot Partition Information**

This optional property defines the characteristics of Boot Partitions (refer to section 8.1.3). If the controller does not support the Boot Partitions feature, then this property shall be cleared to 0h.

**Figure 49: Offset 40h: BPINFO – Boot Partition Information**

Bits	Type	Reset	Description										
31	RO	Impl Spec	<b>Active Boot Partition ID (ABPID):</b> This bit indicates the identifier of the active Boot Partition.										
30:26	RO	0h	Reserved										
25:24	RO	00b	<p><b>Boot Read Status (BRS):</b> This field indicates the status of Boot Partition read operations initiated by the host writing to the BPRSEL.BPID field. Refer to section 8.1.3.</p> <p>The boot read status values are defined as:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>No Boot Partition read operation requested</td> </tr> <tr> <td>01b</td> <td>Boot Partition read in progress</td> </tr> <tr> <td>10b</td> <td>Boot Partition read completed successfully</td> </tr> <tr> <td>11b</td> <td>Error completing Boot Partition read</td> </tr> </tbody> </table> <p>If host software writes the BPRSEL.BPID field, this field transitions to 01b. After successfully completing a Boot Partition read operation (i.e., transferring the contents to the boot memory buffer), the controller sets this field to 10b. If there is an error completing a Boot Partition read operation, this field is set to 11b, and the contents of the boot memory buffer are undefined.</p>	Value	Definition	00b	No Boot Partition read operation requested	01b	Boot Partition read in progress	10b	Boot Partition read completed successfully	11b	Error completing Boot Partition read
Value	Definition												
00b	No Boot Partition read operation requested												
01b	Boot Partition read in progress												
10b	Boot Partition read completed successfully												
11b	Error completing Boot Partition read												
23:15	RO	0h	Reserved										
14:00	RO	Impl Spec	<b>Boot Partition Size (BPSZ):</b> This field defines the size of each Boot Partition in multiples of 128 KiB. Both Boot Partitions are the same size.										

**3.1.4.14 Offset 44h: BPRSEL – Boot Partition Read Select**

This optional property is used to initiate the transfer of a data in the Boot Partition (refer to section 8.1.3) from the controller to the host. If the controller does not support the Boot Partitions feature, then this property shall be cleared to 0h.

If the host attempts to read beyond the end of a Boot Partition (i.e., the Boot Partition Read Offset plus Boot Partition Read Size, is greater than the Boot Partition Size in bytes), the controller shall not transfer data and report an error in the BPINFO.BRS field.

**Figure 50: Offset 44h: BPRSEL – Boot Partition Read Select**

Bits	Type	Reset	Description
31	RW	0b	<b>Boot Partition Identifier (BPID):</b> This bit specifies the Boot Partition identifier for the Boot Partition read operation.
30	RO	0b	Reserved
29:10	RW	0h	<b>Boot Partition Read Offset (BPROF):</b> This field selects the offset into the Boot Partition, in 4 KiB units, that the controller copies into the Boot Partition Memory Buffer.
09:00	RW	0h	<b>Boot Partition Read Size (BPRSZ):</b> This field selects the read size in multiples of 4 KiB to copy into the Boot Partition Memory Buffer.

### 3.1.4.15 Offset 48h: BPMBL – Boot Partition Memory Buffer Location

This optional property specifies the memory buffer that is used as the destination for data when a Boot Partition is read (refer to section 8.1.3). If the controller does not support the Boot Partitions feature, then this property shall be cleared to 0h.

**Figure 51: Offset 48h: BPMBL – Boot Partition Memory Buffer Location**

Bits	Type	Reset	Description
63:12	RW	0h	<b>Boot Partition Memory Buffer Base Address (BMBBA):</b> This field specifies the 52 most significant bits of the 64-bit physical address for the Boot Partition Memory Buffer.
11:00	RO	0h	Reserved

### 3.1.4.16 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).

**Figure 52: Offset 50h: CMBMSC – Controller Memory Buffer Memory Space Control**

Bits	Type	Reset	Description
63:12	RW	0h	<b>Controller Base Address (CBA):</b> This field specifies the 52 most significant bits of the 64-bit base address for the Controller Memory Buffer's controller address range. The Controller Memory Buffer's controller base address and its size determine its controller address range.  The specified address shall be valid only under the following conditions: a) no part of the Controller Memory Buffer's controller address range is greater than $2^{64} - 1$ ; and b) if the Persistent Memory Region's controller memory space is enabled, then the Controller Memory Buffer's controller address range does not overlap the Persistent Memory Region's controller address range.
11:02	RO	0h	Reserved
01	RW	0b	<b>Controller Memory Space Enable (CMSE):</b> This bit specifies whether addresses supplied by the host are permitted to reference the Controller Memory Buffer.  If CMBMSC.CRE is cleared to '0' this bit has no effect, and the Controller Memory Buffer's controller memory space is not enabled.  If this bit is set to '1' and the controller base address is valid, then the Controller Memory Buffer's controller memory space is enabled. Otherwise, the controller memory space is not enabled.  If the Controller Memory Buffer's controller memory space is enabled, then addresses supplied by the host that fall within the Controller Memory Buffer's controller address range shall reference the Controller Memory Buffer.  If the Controller Memory Buffer's controller memory space is not enabled, then no address supplied by the host shall reference the Controller Memory Buffer. Instead, such addresses shall reference memory spaces other than the Controller Memory Buffer.
00	RW	0b	<b>Capabilities Registers Enabled (CRE):</b> This bit specifies whether the CMBLOC and CMBSZ properties are enabled. If this bit is set to '1', then CMBLOC is defined as shown in Figure 47 and CMBSZ is defined as shown in Figure 48. If this bit is cleared to '0', then CMBSZ and CMBLOC are cleared to 0h.

### 3.1.4.17 Offset 58h: CMBSTS – Controller Memory Buffer Status

This optional property indicates the status of the Controller Memory Buffer. If the controller supports the Controller Memory Buffer (CAP.CMBS), this property is mandatory. Otherwise, this property is reserved.

**Figure 53: Offset 58h: CMBSTS – Controller Memory Buffer Status**

Bits	Type	Reset	Description
31:01	RO	0h	Reserved
00	RO	0b	<b>Controller Base Address Invalid (CBAI):</b> This bit indicates whether the controller has failed to enable the Controller Memory Buffer's controller memory space because CMBMSC.CBA is invalid. If CMBMSC.CRE and CMBMSC.CMSE are set to '1', and CMBMSC.CBA is invalid, this bit shall be set to '1'. Otherwise, this bit shall be cleared to '0'.

### 3.1.4.18 Offset 5Ch: CMBEBS – Controller Memory Buffer Elasticity Buffer Size

This optional property identifies to the host the size of the CMB elasticity buffer. A value of 0h in this property indicates to the host that no information regarding the presence or size of a CMB elasticity buffer is available.

**Figure 54: Offset 5Ch: CMBEBS – Controller Memory Buffer Elasticity Buffer Size**

Bits	Type	Reset	Description												
31:8	RO	Impl Spec	<b>CMB Elasticity Buffer Size Base (CMBWBZ):</b> Indicates the size of the CMB elasticity buffer. The size of the CMB elasticity buffer is equal to the value in this field multiplied by the value specified by the CMB Elasticity Buffer Size Units field.												
7:5	RO	0h	Reserved												
4	RO	Impl Spec	<b>CMB Read Bypass Behavior (CMBRBB):</b> If a memory read does not conflict with any memory write in the CMB Elasticity Buffer (i.e., if the set of memory addresses specified by a read is disjoint from the set of memory addresses specified by all writes in the CMB Elasticity Buffer), and this bit is: <ul style="list-style-type: none"> <li>a) set to '1', then memory reads not conflicting with memory writes in the CMB Elasticity Buffer shall bypass those memory writes; and</li> <li>b) cleared to '0', then memory reads not conflicting with memory writes in the CMB Elasticity Buffer may bypass those memory writes.</li> </ul>												
3:0	RO	Impl Spec	<b>CMB Elasticity Buffer Size Units (CMBSZU):</b> Indicates the granularity of the CMB Elasticity Buffer Size Base field. <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Value</th> <th>Granularity</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Bytes</td> </tr> <tr> <td>1h</td> <td>1 KiB</td> </tr> <tr> <td>2h</td> <td>1 MiB</td> </tr> <tr> <td>3h</td> <td>1 GiB</td> </tr> <tr> <td>4h – Fh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Granularity	0h	Bytes	1h	1 KiB	2h	1 MiB	3h	1 GiB	4h – Fh	Reserved
Value	Granularity														
0h	Bytes														
1h	1 KiB														
2h	1 MiB														
3h	1 GiB														
4h – Fh	Reserved														

### 3.1.4.19 Offset 60h: CMBSWTP – Controller Memory Buffer Sustained Write Throughput

This optional property identifies to the host the maximum CMB sustained write throughput. A value of 0h in this property indicates to the host that no information regarding the CMB sustained write throughput is available.

**Figure 55: Offset 60h: CMBSWTP – Controller Memory Buffer Sustained Write Throughput**

Bits	Type	Reset	Description
31:8	RO	Impl Spec	<b>CMB Sustained Write Throughput (CMBSWTV):</b> Indicates the sustained write throughput of the CMB at the maximum payload size specified by the applicable NVMe Transport binding specification (e.g., the PCIe TLP payload size, as specified in the Max_Payload_Size (MPS) field of the PCIe Express Device Control (PXDC) register). The sustained write throughput of the CMB is equal to the value in this field multiplied by the units specified by the CMB Sustained Write Throughput Units field.

7:4	RO	0h	Reserved		
3:0	RO	Impl Spec	<b>CMB Sustained Write Throughput Units (CMBSWTU):</b> Indicates the granularity of the CMB Sustained Write Throughput field.		
				<b>Value</b>	<b>Granularity</b>
				0h	Bytes/second
				1h	1 KiB/second
				2h	1 MiB/second
				3h	1 GiB/second
4h – Fh	Reserved				

### 3.1.4.20 Offset 64h: NSSD – NVM Subsystem Shutdown

This optional property provides host software with the capability to initiate a normal or an abrupt NVM Subsystem Shutdown.

Support for this property is indicated by the state of the NVM Subsystem Shutdown Supported (CAP.NSSS) field. If the property is not supported, then the address range occupied by the register is reserved.

The NVM Subsystem Shutdown Enhancements Supported (CAP.NSSES) bit affects the functionality invoked by host modification of this property (refer to section 3.6.3).

**Figure 56: Offset 64h: NSSD – NVM Subsystem Shutdown**

Bits	Type	Reset	Description
31:00	RW	0h	<p><b>NVM Subsystem Shutdown Control (NSSC):</b> A write of the value 4E726D6Ch ("NrmI") to this field initiates a normal NVM Subsystem Shutdown on every controller:</p> <ul style="list-style-type: none"> <li>in the domain associated with the controller when CAP.CPS is set to 10b (i.e., domain scope) as specified in section 3.6.3.2 or</li> <li>in the NVM subsystem when CAP.CPS is set to 11b (i.e., NVM subsystem scope) in the NVM subsystem as specified in section 3.6.3.1.</li> </ul> <p>A write of the value 41627074h ("Abpt") to this field initiates an abrupt NVM subsystem shutdown on every controller:</p> <ul style="list-style-type: none"> <li>in the domain associated with the controller when CAP.CPS is set to 10b as specified in section 3.6.3.2; or</li> <li>in the NVM subsystem when CAP.CPS is set to 11b in the NVM subsystem as specified in section 3.6.3.1.</li> </ul> <p>A write of any other value to this field has no functional effect on the operation of the NVM subsystem. This field shall return the value 0h when read.</p>

### 3.1.4.21 Offset 68h: CRT0 – Controller Ready Timeouts

This property indicates the controller ready timeout values. This property is mandatory for controllers compliant with NVM Express Base Specification revision 2.0 and later.

**Figure 57: Offset 68h: CRTO – Controller Ready Timeouts**

Bits	Type	Reset	Description
31:16	RO	Impl Spec	<p><b>Controller Ready Independent of Media Timeout (CRIMT):</b> If the CAP.CRMS.CRIMS bit is cleared to '0', then the controller shall clear this field to 0h and the host should ignore this field.</p> <p>If the CAP.CRMS.CRIMS bit is set to '1', then this field contains the worst-case time that host software should wait after CC.EN transitions from '0' to '1' for the controller to become ready and be able to successfully process all commands that do not access attached namespaces and Admin commands that do not require access to media when the controller is in Controller Ready Independent of Media mode (i.e., the CC.CRIME bit is set to '1'). Attached namespaces and media required to process Admin commands may or may not be ready within this time period (refer to section 3.5.3, section 3.5.4, and Figure 84).</p> <p>This worst-case time may be experienced after events such as an abrupt shutdown or activation of a new firmware image; typical times are expected to be much shorter. This field is in 500 millisecond units.</p> <p>The value of this field should not exceed FFh (i.e., 127.5 seconds).</p>
15:0	RO	Impl Spec	<p><b>Controller Ready With Media Timeout (CRWMT):</b> This field contains the worst-case time that host software should wait after CC.EN transitions from '0' to '1' for:</p> <ul style="list-style-type: none"> <li>a) the controller to become ready and be able to successfully process all commands; and</li> <li>b) all attached namespaces and media required to process Admin commands to become ready,</li> </ul> <p>independent of which ready mode (refer to CC.CRIME) the controller is in (refer to section 3.5.3 and section 3.5.4).</p> <p>This worst-case time may be experienced after events such as an abrupt shutdown or activation of a new firmware image; typical times are expected to be much shorter. This field is in 500 millisecond units.</p> <p>The value of this field shall be greater than or equal to the value of the CRTO.CRIMT field and may be significantly larger than the value of the CRTO.CRIMT field.</p>

**3.1.4.22 Offset E00h: PMRCAP – Persistent Memory Region Capabilities**

This property indicates capabilities of the Persistent Memory Region. If the controller does not support the Persistent Memory Region feature, then this property shall be cleared to 0h.

This property shall not be reset by Controller Reset.

**Figure 58: Offset E00h: PMRCAP – Persistent Memory Region Capabilities**

Bits	Type	Reset	Description
31:25	RO	0h	Reserved
24	RO	Impl Spec	<p><b>Controller Memory Space Supported (CMSS):</b> If this bit is set to '1', then the addresses supplied by the host are permitted to reference the Persistent Memory Region only if the host has enabled the Persistent Memory Region's controller memory space.</p> <p>If the controller supports referencing the Persistent Memory Region with host-supplied addresses, then this bit shall be set to '1'. Otherwise, this bit shall be cleared to '0'.</p>
23:16	RO	Impl Spec	<p><b>Persistent Memory Region Timeout (PMRTO):</b> This field contains the minimum amount of time that host software should wait for the Persistent Memory Region to become ready or not ready after PMRCTL.EN is modified. The time in this field is expressed in Persistent Memory Region time units (refer to PMRCAP.PMRTU).</p>
15:14	RO	00b	Reserved



**Figure 58: Offset E00h: PMRCAP – Persistent Memory Region Capabilities**

Bits	Type	Reset	Description								
13:10	RO	Impl Spec	<p><b>Persistent Memory Region Write Barrier Mechanisms (PMRWBM):</b> This field lists mechanisms that may be used to ensure that previous writes to the Persistent Memory Region have completed and are persistent when the Persistent Memory Region is ready and operating normally. A bit in this field is set to '1' if the corresponding mechanism to ensure persistence is supported. A bit in this field is cleared to '0' if the corresponding mechanism to ensure persistence is not supported.</p> <p>At least one bit in this field shall be set to '1'.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>3:2</td> <td>Reserved</td> </tr> <tr> <td>1</td> <td><b>Completion of PMRSTS Read (CPMTSTSR):</b> The completion of a read to the PMRSTS property shall ensure that all prior writes to the Persistent Memory Region have completed and are persistent.</td> </tr> <tr> <td>0</td> <td><b>Completion of Memory Read (CMR):</b> The completion of a memory read from any Persistent Memory Region address ensures that all prior writes to the Persistent Memory Region have completed and are persistent.</td> </tr> </tbody> </table>	Bits	Description	3:2	Reserved	1	<b>Completion of PMRSTS Read (CPMTSTSR):</b> The completion of a read to the PMRSTS property shall ensure that all prior writes to the Persistent Memory Region have completed and are persistent.	0	<b>Completion of Memory Read (CMR):</b> The completion of a memory read from any Persistent Memory Region address ensures that all prior writes to the Persistent Memory Region have completed and are persistent.
			Bits	Description							
			3:2	Reserved							
			1	<b>Completion of PMRSTS Read (CPMTSTSR):</b> The completion of a read to the PMRSTS property shall ensure that all prior writes to the Persistent Memory Region have completed and are persistent.							
0	<b>Completion of Memory Read (CMR):</b> The completion of a memory read from any Persistent Memory Region address ensures that all prior writes to the Persistent Memory Region have completed and are persistent.										
9:8	RO	Impl Spec	<p><b>Persistent Memory Region Time Units (PMRTU):</b> Indicates Persistent Memory Region time units.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Persistent Memory Region Time Units</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>500 milliseconds</td> </tr> <tr> <td>01b</td> <td>minutes</td> </tr> <tr> <td>10b to 11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Persistent Memory Region Time Units	00b	500 milliseconds	01b	minutes	10b to 11b	Reserved
			Value	Persistent Memory Region Time Units							
			00b	500 milliseconds							
01b	minutes										
10b to 11b	Reserved										
7:5	RO	Impl Spec	<p><b>Base Indicator Register (BIR):</b> This field indicates the Base Address Register (BAR) that specifies the address and size of the Persistent Memory Region. Values 010b, 011b, 100b, and 101b are valid.</p>								
4	RO	Impl Spec	<p><b>Write Data Support (WDS):</b> If this bit is set to '1', then the controller supports data and metadata in the Persistent Memory Region for commands that transfer data from the host to the controller (e.g., Write). If this bit is cleared to '0', then data and metadata for commands that transfer data from the host to the controller shall not be transferred to the Persistent Memory Region.</p> <p>If PMRCAP.CMSS is cleared to '0', this bit shall be cleared to '0'.</p>								
3	RO	Impl Spec	<p><b>Read Data Support (RDS):</b> If this bit is set to '1', then the controller supports data and metadata in the Persistent Memory Region for commands that transfer data from the controller to the host (e.g., Read). If this bit is cleared to '0', then all data and metadata for commands that transfer data from the controller to the host shall not be transferred from the Persistent Memory Region.</p> <p>If PMRCAP.CMSS is cleared to '0', this bit shall be cleared to '0'.</p>								
2:0	RO	000b	Reserved								

**3.1.4.23 Offset E04h: PMRCTL – Persistent Memory Region Control**

This optional property controls the operation of the Persistent Memory Region. If the controller does not support the Persistent Memory Region feature, then this property shall be cleared to 0h.

This property shall not be reset by Controller Reset.

**Figure 59: Offset E04h: PMRCTL – Persistent Memory Region Control**

Bits	Type	Reset	Description
31:1	RO	0h	Reserved
0	RW	0b	<p><b>Enable (EN):</b> When set to '1', then the Persistent Memory Region is ready to process PCI Express memory read and write requests once PMRSTS.NRDY is cleared to '0'. When cleared to '0', then the Persistent Memory Region is disabled and PMRSTS.NRDY shall be set to '1' once the Persistent Memory Region is ready to be re-enabled.</p>

### 3.1.4.24 Offset E08h: PMRSTS – Persistent Memory Region Status

This optional property provides the status of the Persistent Memory Region. If the controller does not support the Persistent Memory Region feature, then this property shall be cleared to 0h.

This property shall not be reset by Controller Reset.

**Figure 60: Offset E08h: PMRSTS – Persistent Memory Region Status**

Bits	Type	Reset	Description												
31:13	RO	0h	Reserved												
12	RO	0b	<b>Controller Base Address Invalid (CBAI):</b> This field indicates whether the controller has failed to enable the Persistent Memory Region's controller memory space because the controller 64-bit base address specified by PMRMSCU.CBA and PMRMSCL.CBA are invalid. If PMRCAP.CMSS is set to '1', PMRMSCL.CMSE is set to '1', and the controller 64-bit base address specified by PMRMSCU.CBA and PMRMSCL.CBA is invalid, this bit shall be set to '1'. Otherwise, this bit shall be cleared to '0'.												
11:9	RO	000b	<p><b>Health Status (HSTS):</b> If the Persistent Memory Region is ready, then this field indicates the health status of the Persistent Memory Region. This field is always cleared to 000b when the Persistent Memory Region is not ready.</p> <p>The health status values are defined as:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td><b>Normal Operation:</b> The Persistent Memory Region is operating normally.</td> </tr> <tr> <td>001b</td> <td><b>Restore Error:</b> The Persistent Memory Region is operating normally and is persistent; however, the contents of the Persistent Memory Region may not have been restored correctly (i.e., may not contain the contents prior to the last power cycle, NVM Subsystem Reset, Controller Level Reset, or Persistent Memory Region disable).</td> </tr> <tr> <td>010b</td> <td><b>Read Only:</b> The Persistent Memory Region is read only. PCI Express memory write requests do not update the Persistent Memory Region. PCI Express memory read requests to the Persistent Memory Region return correct data.</td> </tr> <tr> <td>011b</td> <td><b>Unreliable:</b> The Persistent Memory Region has become unreliable. PCI Express memory reads may return invalid data or generate poisoned PCI Express TLP(s). Persistent Memory Region memory writes may not update memory or may update memory with undefined data. The Persistent Memory Region may also have become non-persistent.</td> </tr> <tr> <td>100b to 111b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	000b	<b>Normal Operation:</b> The Persistent Memory Region is operating normally.	001b	<b>Restore Error:</b> The Persistent Memory Region is operating normally and is persistent; however, the contents of the Persistent Memory Region may not have been restored correctly (i.e., may not contain the contents prior to the last power cycle, NVM Subsystem Reset, Controller Level Reset, or Persistent Memory Region disable).	010b	<b>Read Only:</b> The Persistent Memory Region is read only. PCI Express memory write requests do not update the Persistent Memory Region. PCI Express memory read requests to the Persistent Memory Region return correct data.	011b	<b>Unreliable:</b> The Persistent Memory Region has become unreliable. PCI Express memory reads may return invalid data or generate poisoned PCI Express TLP(s). Persistent Memory Region memory writes may not update memory or may update memory with undefined data. The Persistent Memory Region may also have become non-persistent.	100b to 111b	Reserved
Value	Definition														
000b	<b>Normal Operation:</b> The Persistent Memory Region is operating normally.														
001b	<b>Restore Error:</b> The Persistent Memory Region is operating normally and is persistent; however, the contents of the Persistent Memory Region may not have been restored correctly (i.e., may not contain the contents prior to the last power cycle, NVM Subsystem Reset, Controller Level Reset, or Persistent Memory Region disable).														
010b	<b>Read Only:</b> The Persistent Memory Region is read only. PCI Express memory write requests do not update the Persistent Memory Region. PCI Express memory read requests to the Persistent Memory Region return correct data.														
011b	<b>Unreliable:</b> The Persistent Memory Region has become unreliable. PCI Express memory reads may return invalid data or generate poisoned PCI Express TLP(s). Persistent Memory Region memory writes may not update memory or may update memory with undefined data. The Persistent Memory Region may also have become non-persistent.														
100b to 111b	Reserved														
8	RO	0b	<b>Not Ready (NRDY):</b> This bit indicates if the Persistent Memory Region is ready for use. If this bit is cleared to '0' and the PMRCTL.EN is set to '1', then the Persistent Memory Region is ready to accept and process PCI Express memory read and write requests. If this bit is set to '1' or the PMRCTL.EN bit is cleared to '0', then the Persistent Memory Region is not ready to process PCI Express memory read and write requests.												
7:0	RO	0h	<p><b>Error (ERR):</b> When the Persistent Memory Region is ready and operating normally, this field indicates whether previous memory writes to the Persistent Memory Region have completed without error. If this field is cleared to 0h, then previous writes to the Persistent Memory Region have completed without error and that the values written are persistent. A non-zero value in this field indicates the occurrence of an error that may have caused one or more of the previous writes to not have completed successfully. The meaning of any particular non-zero value is vendor specific.</p> <p>Once this field takes on a non-zero value, it maintains a non-zero value until the PCI Function is reset.</p>												

### 3.1.4.25 Offset E0Ch: PMREBS – Persistent Memory Region Elasticity Buffer Size

This optional property identifies to the host the size of the PMR elasticity buffer. A value of 0h in this property indicates to the host that no information regarding the presence or size of a PMR elasticity buffer is available.

This property shall not be reset by Controller Reset.

**Figure 61: Offset E0Ch: PMREBS – Persistent Memory Region Elasticity Buffer Size**

Bits	Type	Reset	Description												
31:8	RO	Impl Spec	<b>PMR Elasticity Buffer Size Base (PMRWBZ):</b> Indicates the size of the PMR elasticity buffer. The actual size of the PMR elasticity buffer is equal to the value in this field multiplied by the value specified by the PMR Elasticity Buffer Size Units field.												
7:5	RO	000b	Reserved												
4	RO	Impl Spec	<b>PMR Read Bypass Behavior (PMRRBB):</b> If a memory read does not conflict with any memory write in the PMR Elasticity Buffer (i.e., if the set of memory addresses specified by a read is disjoint from the set of memory addresses specified by all writes in the PMR Elasticity Buffer), and this bit is: <ol style="list-style-type: none"> <li>set to '1', then memory reads not conflicting with memory writes in the PMR Elasticity Buffer shall bypass those memory writes; and</li> <li>cleared to '0', then memory reads not conflicting with memory writes in the PMR Elasticity Buffer may bypass those memory writes.</li> </ol>												
3:0	RO	Impl Spec	<b>PMR Elasticity Buffer Size Units (PMRSZU):</b> Indicates the granularity of the PMR Elasticity Buffer Size Base field. <table border="1" data-bbox="771 913 1136 1085"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Bytes</td> </tr> <tr> <td>1h</td> <td>1 KiB</td> </tr> <tr> <td>2h</td> <td>1 MiB</td> </tr> <tr> <td>3h</td> <td>1 GiB</td> </tr> <tr> <td>4h to Fh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	Bytes	1h	1 KiB	2h	1 MiB	3h	1 GiB	4h to Fh	Reserved
Value	Definition														
0h	Bytes														
1h	1 KiB														
2h	1 MiB														
3h	1 GiB														
4h to Fh	Reserved														

### 3.1.4.26 Offset E10h: PMRSWTP – Persistent Memory Region Sustained Write Throughput

This optional property identifies to the host the maximum PMR sustained write throughput. A value of 0h in this property indicates to the host that no information regarding the PMR sustained write throughput is available.

This property shall not be reset by Controller Reset.

**Figure 62: Offset E10h: PMRSWTP – Persistent Memory Region Sustained Write Throughput**

Bits	Type	Reset	Description												
31:8	RO	Impl Spec	<b>PMR Sustained Write Throughput (PMRSWTV):</b> Indicates the sustained write throughput of the PMR at the maximum payload size specified by the applicable NVMe Transport binding specification (e.g., the PCIe TLP payload size, as specified in the Max_Payload_Size (MPS) field of the PCI Express Device Control (PXDC) register). The actual sustained write throughput of the PMR is equal to the value in this field multiplied by the units specified by the PMR Sustained Write Throughput Units field.												
7:4	RO	0h	Reserved												
3:0	RO	Impl Spec	<b>PMR Sustained Write Throughput Units (PMRSWTU):</b> Indicates the granularity of the PMR Sustained Write Throughput field. <table border="1" data-bbox="771 1680 1136 1854"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Bytes per second</td> </tr> <tr> <td>1h</td> <td>1 KiB / s</td> </tr> <tr> <td>2h</td> <td>1 MiB / s</td> </tr> <tr> <td>3h</td> <td>1 GiB / s</td> </tr> <tr> <td>7h to Fh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	Bytes per second	1h	1 KiB / s	2h	1 MiB / s	3h	1 GiB / s	7h to Fh	Reserved
Value	Definition														
0h	Bytes per second														
1h	1 KiB / s														
2h	1 MiB / s														
3h	1 GiB / s														
7h to Fh	Reserved														

### 3.1.4.27 Offset E14h: PMRMSCS – Persistent Memory Region Memory Space Control Lower

This optional property and the PMRMSCU property specify how the controller references the Persistent Memory Region with host-supplied addresses. If the controller supports the Persistent Memory Region's controller memory space (PMRCAP.CMSS), this property is mandatory. Otherwise, this property is reserved. The host shall access this register with aligned 32-bit accesses.

This property shall not be reset by Controller Reset.

**Figure 63: Offset E14h: PMRMSCS – Persistent Memory Region Memory Space Control Lower**

Bits	Type	Reset	Description
31:12	RW	0h	<p><b>Controller Base Address (CBA):</b> This field specifies the 20 least significant bits of the 52 most significant bits of the 64-bit base address for the Persistent Memory Region's controller address range. The Persistent Memory Region's controller base address and its size determine its controller address range.</p> <p>The 64-bit base address specified by this field and PMRMSCU.CBA when the CMSE bit is set to '1' shall be valid only under the following conditions:</p> <ul style="list-style-type: none"> <li>a) no part of the Persistent Memory Region's controller address range is greater than <math>2^{64} - 1</math>; and</li> <li>b) if the Controller Memory Buffer's controller memory space is enabled, then the Persistent Memory Region's controller address range does not overlap the Controller Memory Buffer's controller address range.</li> </ul>
11:02	RO	0h	Reserved
01	RW	0b	<p><b>Controller Memory Space Enable (CMSE):</b> This bit specifies whether addresses supplied by the host are permitted to reference the Persistent Memory Region.</p> <p>If this bit is set to '1' and the controller base address is valid, then the Persistent Memory Region's controller memory space is enabled. Otherwise, the controller memory space is not enabled.</p> <p>If the Persistent Memory Region's controller memory space is enabled, then addresses supplied by the host that fall within the Persistent Memory Region's controller address range shall reference the Persistent Memory Region.</p> <p>If the Persistent Memory Region's controller memory space is not enabled, then no address supplied by the host shall reference the Persistent Memory Region. Instead, such addresses shall reference memory spaces other than the Persistent Memory Region.</p>
00	RO	0b	Reserved

### 3.1.4.28 Offset E18h: PMRMSCU – Persistent Memory Region Memory Space Control Upper

This optional property and the PMRMSCS property specify how the controller references the Persistent Memory Region with host-supplied addresses. If the controller supports the Persistent Memory Region's controller memory space (PMRCAP.CMSS), this property is mandatory. Otherwise, this property is reserved. The host shall access this register with aligned 32-bit accesses.

This register shall not be reset by Controller Reset.

**Figure 64: Offset E18h: PMRMSCU – Persistent Memory Region Memory Space Control Upper**

Bits	Type	Reset	Description
31:00	RW	0h	<p><b>Controller Base Address (CBA):</b> This field specifies the 32 most significant bits of the 52 most significant bits of the 64-bit base address for the Persistent Memory Region's controller address range. The Persistent Memory Region's controller base address and its size determine its controller address range.</p>

## 3.2 NVM Subsystem Entities

### 3.2.1 Namespaces

#### 3.2.1.1 Namespace Overview

A namespace is a formatted quantity of non-volatile memory that may be directly accessed by a host. A namespace ID (NSID) is an identifier used by a controller to provide access to a namespace.

#### 3.2.1.2 Valid and Invalid NSIDs

Valid NSIDs are the range of possible NSIDs that may be used to refer to namespaces that exist in the NVM subsystem. Any NSID is valid, except if that NSID is 0h or greater than the Number of Namespaces field reported in the Identify Controller data structure (refer to Figure 312). NSID FFFFFFFFh is a broadcast value that is used to specify all namespaces. An invalid NSID is any value that is not a valid NSID and is also not the broadcast value.

Valid NSIDs are:

- a) allocated or unallocated in the NVM subsystem; and
- b) active or inactive for a specific controller.

#### 3.2.1.3 Allocated and Unallocated NSID Types

In the NVM subsystem, a valid NSID is:

- a) an allocated NSID; or
- b) an unallocated NSID.

Allocated NSIDs refer to namespaces that exist in the NVM subsystem. Unallocated NSIDs do not refer to any namespaces that exist in the NVM subsystem.

#### 3.2.1.4 Active and Inactive NSID Types

For a specific controller, an allocated NSID is:

- a) an active NSID; or
- b) an inactive NSID.

Active NSIDs for a controller refer to namespaces that are attached to that controller. Allocated NSIDs that are inactive for a controller refer to namespaces that are not attached to that controller.

Unallocated NSIDs are inactive NSIDs for all controllers in the NVM subsystem.

An allocated NSID may be an active NSID for some controllers and an inactive NSID for other controllers in the same NVM subsystem if the namespace that the NSID refers to is attached to some controllers, but not all controllers, in the NVM subsystem.

Refer to section 8.1.15 for actions associated with a namespace being detached or deleted.

#### 3.2.1.5 NSID and Namespace Relationships

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

Figure 65 summarizes the valid NSID types and Figure 66 visually shows the NSID types and how they relate.

**Figure 65: NSID Types and Relationship to Namespace**

Valid NSID Type	NSID relationship to namespace	Reference
Unallocated	Does not refer to any namespace that exists in the NVM subsystem	3.2.1.3

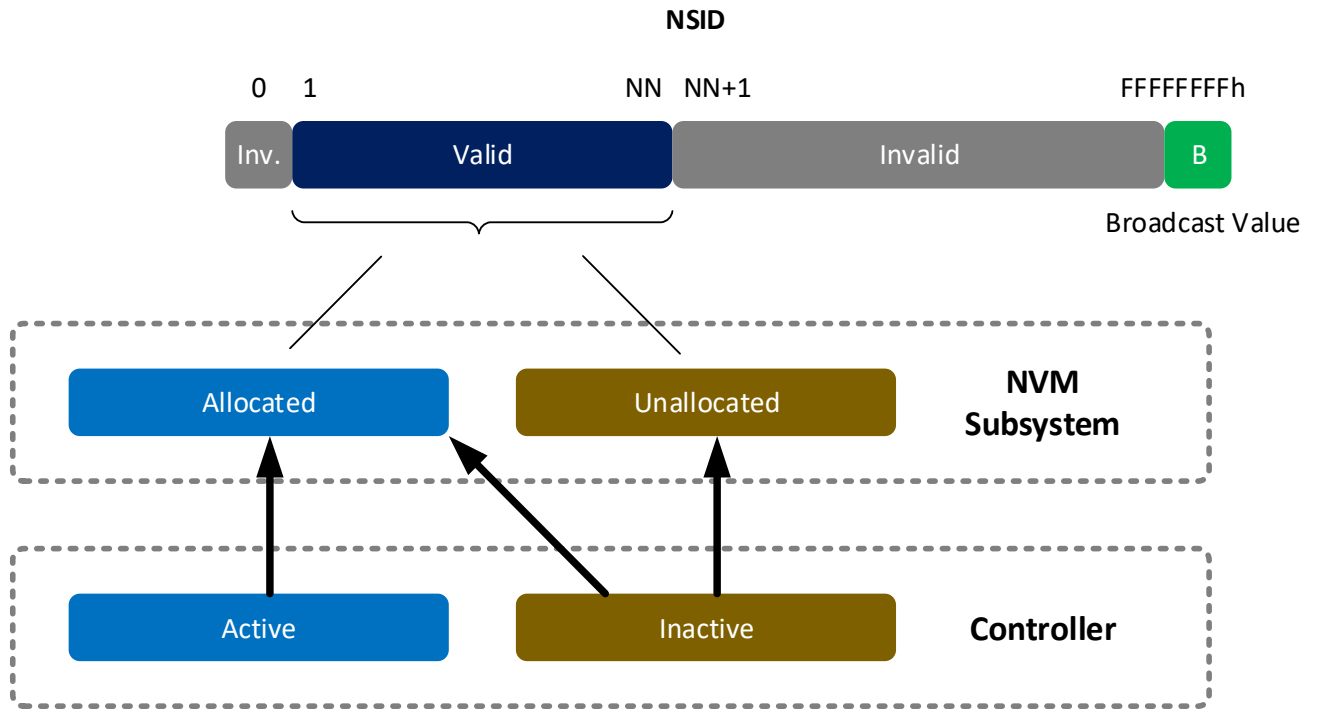
**Figure 65: NSID Types and Relationship to Namespace**

Valid NSID Type	NSID relationship to namespace	Reference
Allocated	Refers to a namespace that exists in the NVM subsystem	3.2.1.3
Inactive	Does not refer to a namespace that is attached to the controller <sup>1</sup>	3.2.1.4
Active	Refers to a namespace that is attached to the controller	3.2.1.4

Notes:

1. If allocated, refers to a namespace that is not attached to the controller. If unallocated, does not refer to any namespace.

**Figure 66: NSID Types**



**3.2.1.6 NSID and Namespace Usage**

If Namespace Management (refer to section 8.1.15), ANA Reporting (refer to section 8.1.1), or NVM Sets (refer to section 3.2.2) capabilities are supported, then NSIDs shall be unique within the NVM subsystem (e.g., NSID of 3 shall refer to the same physical namespace regardless of the accessing controller). If the Namespace Management, ANA Reporting, and NVM Sets capabilities are not supported, then NSIDs:

- a) for shared namespaces shall be unique within the NVM subsystem; and
- b) for private namespaces are not required to be unique within the NVM subsystem.

The Identify command (refer to section 5.1.13) may be used to determine the active NSIDs for a controller and the allocated NSIDs in the NVM subsystem.

If the MNAN field (refer to Figure 312) is cleared to 0h, then the maximum number of allocated NSIDs is the same as the value reported in the NN field (refer to Figure 312). If the MNAN field is non-zero, then the maximum number of allocated NSIDs may be less than the number of namespaces (e.g., an NVM subsystem may support a maximum valid NSID value (i.e., the NN field) set to 1,000,000 but support a maximum of 10 allocated NSID values).

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

1. Issue an Identify command with the CNS field cleared to 0h for each valid NSID (based on the Number of Namespaces value (i.e., MNAM field or NN field) in the Identify Controller data structure). If a non-zero data structure is returned for a particular NSID, then that is an active NSID; or
2. Issue an Identify command with a CNS field set to 2h to retrieve a list of up to 1,024 active NSIDs. If there are more than 1,024 active NSIDs, continue to issue Identify commands with a CNS field set to 2h until all active NSIDs are retrieved.

To determine the allocated NSIDs in the NVM subsystem, the host may issue an Identify command with the CNS field set to 10h to retrieve a list of up to 1,024 allocated NSIDs. If there are more than 1,024 allocated NSIDs, continue to issue Identify commands with a CNS field set to 10h until all allocated NSIDs are retrieved.

Namespace IDs may change across power off conditions. However, it is recommended that namespace IDs remain static across power off conditions to avoid issues with host software. To determine if the same namespace has been encountered, the host may use the:

- a) UUID field in the Namespace Identification Descriptor (refer to Figure 315), if present;
- b) NGUID field in the Identify Namespace data (refer to the applicable I/O Command Set specification) or in the Namespace Identification Descriptor, if present; or
- c) EUI64 field in the Identify Namespace data or in the Namespace Identification Descriptor, if present.

UIDREUSE bit in the NSFEAT field (refer to Figure 319 or the Identify Namespace data structure in the NVM Command Set Specification, if applicable) indicates NGUID and EUI64 reuse characteristics.

If Asymmetric Namespace Access Reporting is supported (i.e., the Asymmetric Namespace Access Reporting Support (ANARS) bit is set to '1' in the CMIC field in the Identify Controller data structure (refer to Figure 312)), refer to the applicable I/O Command Set specification for additional detail, if any.

A namespace may or may not have a relationship to a Submission Queue; this relationship is determined by the host software implementation. The controller shall support access to any attached namespace from any I/O Submission Queue.

### 3.2.1.7 I/O Command Set Associations

A namespace is associated with exactly one I/O Command Set. For I/O commands and I/O Command Set specific Admin commands, the I/O Command Set with which a submission queue entry is associated is determined by the Namespace Identifier (NSID) field in the command.

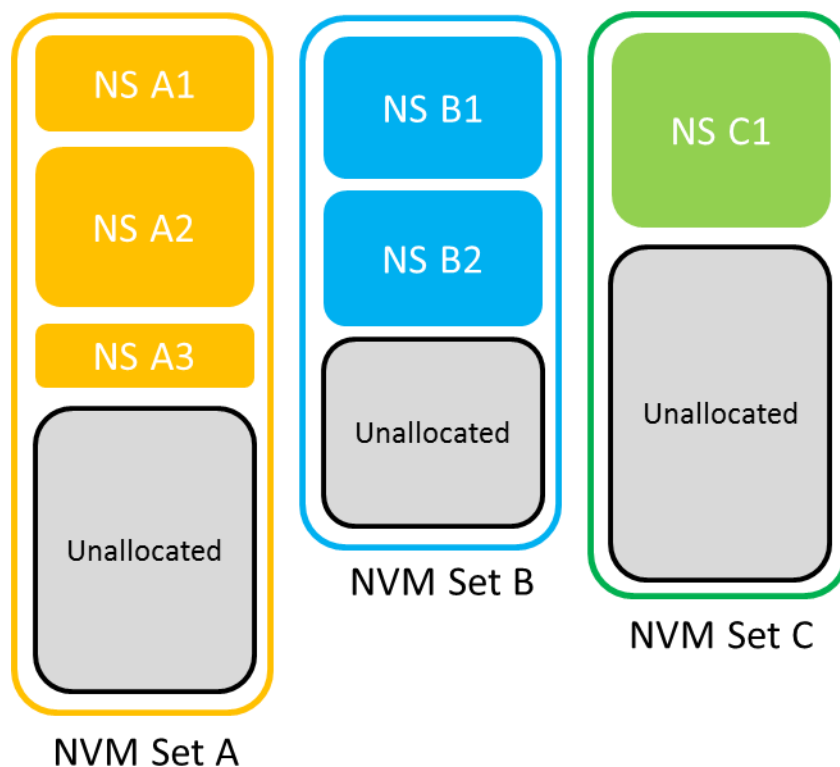
An NVM subsystem may contain namespaces each of which is associated with a different I/O Command Set. A controller may support attached namespaces that use any of the I/O Command Sets that the controller simultaneously supports as indicated in the I/O Command Set Profile (refer to section 5.1.25.1.17).

### 3.2.2 NVM Sets

An NVM Set is a collection of NVM that is separate (logically and potentially physically) from NVM in other NVM Sets. One or more namespaces that contain formatted storage may be created within an NVM Set and those namespaces inherit the attributes of the NVM Set. A namespace that contains formatted storage is wholly contained within a single NVM Set and shall not span more than one NVM Set.

Figure 67 shows an example of three NVM Sets. NVM Set A contains three namespaces (NS A1, NS A2, and NS A3). NVM Set B contains two namespaces (NS B1 and NS B2). NVM Set C contains one namespace (NS C1). Each NVM Set shown also contains 'Unallocated' regions that consist of NVM that is not yet allocated to a namespace.

**Figure 67: NVM Sets and Associated Namespaces**



There is a subset of Admin commands that are NVM Set aware as described in Figure 68.

**Figure 68: NVM Set Aware Admin Commands**

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

The host determines the NVM Sets present and their attributes using the Identify command with CNS value of 04h to retrieve the NVM Set List (refer to Figure 317). For each NVM Set, the attributes include:

- an identifier associated with the NVM Set;
- the optimal size for writes to the NVM Set;



- the total capacity of the NVM Set; and
- the unallocated capacity for the NVM Set.

An NVM Set Identifier is a 16-bit value that specifies the NVM Set with which an action is associated. An NVM Set Identifier is unique with the NVM subsystem. An NVM Set Identifier may be specified in NVM Set aware Admin commands (refer to Figure 68). An NVM Set Identifier value of 0h is reserved and is not a valid NVM Set Identifier. Unless otherwise specified, if the host specifies an NVM Set Identifier cleared to 0h for a command that requires an NVM Set Identifier, then that command shall abort with a status code of Invalid Field in Command.

Each NVM Set is associated with exactly one Endurance Group (refer to section 3.2.3).

The NVM Set with which a namespace that contains formatted storage is associated is reported in the Identify Namespace data structure (refer to the applicable NVMe I/O Command Set specification). When a host creates a namespace that contains formatted storage using the Namespace Management command, the host specifies the NVM Set Identifier of the NVM Set that the namespace is to be created in. The namespace that is created inherits attributes from the NVM Set (e.g., the optimal write size to the NVM).

If NVM Sets are supported, then all controllers in the NVM subsystem shall:

- Indicate support for NVM Sets in the Controller Attributes field in the Identify Controller data structure;
- Support the NVM Set Identifier in all commands that use the NVM Set Identifier;
- Support the NVM Set List for the Identify command;
- Indicate the NVM Set Identifier with which any namespace that contains formatted storage is associated in the Identify Namespace data structure for that namespace;
- Support Endurance Groups; and
- For each NVM Set, indicate the associated Endurance Group as an attribute.

If support for NVM Sets is not reported (i.e., the NVM Sets bit is cleared to '0' in the CTRATT field; refer to Figure 312), then the NVM Set Identifier field shall be cleared to 0h in all commands and data structures that support an NVM Set Identifier field.

### 3.2.3 Endurance Groups

Endurance may be managed within a single NVM Set (refer to section 3.2.2) or across a collection of NVM Sets. Each NVM Set is associated with an Endurance Group (refer to Figure 317). If two or more NVM Sets have the same Endurance Group Identifier, then endurance is managed by the NVM subsystem across that collection of NVM Sets. If only one NVM Set is associated with a specific Endurance Group Identifier, then endurance is managed locally to that NVM Set.

If NVM Sets are not supported, then endurance is managed by the NVM subsystem:

- within each Endurance Group if Endurance Groups are supported; or
- within the domain if Endurance Groups are not supported.

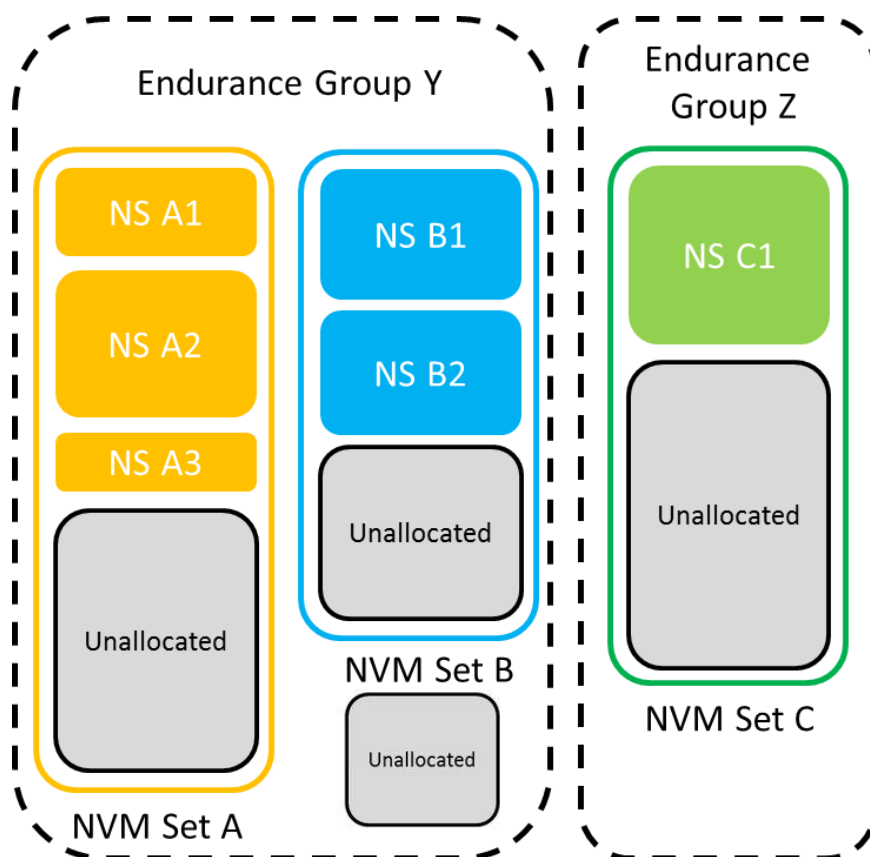
An Endurance Group shall be part of only one domain (refer to section 3.2.5).

An Endurance Group Identifier is a 16-bit value that specifies the Endurance Group with which an action is associated. An Endurance Group Identifier is unique within the NVM subsystem. An Endurance Group Identifier value of 0h is reserved and is not a valid Endurance Group Identifier. Unless otherwise specified, if the host specifies an Endurance Group Identifier cleared to 0h for a command that requires an Endurance Group Identifier, then that command shall abort with a status code of Invalid Field in Command.

The information that describes an Endurance Group is indicated in the Endurance Group Information log page (refer to section 5.1.12.1.10).

Figure 69 shows Endurance Groups added to the example in Figure 67. In this example, the endurance of NVM Set A and NVM Set B are managed together as part of Endurance Group Y, while the endurance of NVM Set C is managed only within NVM Set C which is the only NVM Set that is part of Endurance Group Z.

Figure 69: NVM Sets and Associated Namespaces



If Endurance Groups are supported, then the NVM subsystem and all controllers shall:

- indicate support for Endurance Groups in the Controller Attributes field in the Identify Controller data structure;
- indicate the Endurance Group Identifier with which the namespace is associated in the Identify Namespace data structure;
- support the Endurance Group Information log page; and
- support the Endurance Group Event Aggregate log page if more than one Endurance Group is supported in the NVM subsystem.

If Endurance Groups are not supported and the host sends a command in which an Endurance Group Identifier field is defined (e.g., Get Log Page), then that field shall be ignored by the controller.

If Endurance Groups are not supported and the controller returns information to the host that contains an Endurance Group Identifier field, then that field shall be cleared to 0h.

### 3.2.3.1 Configuring and Managing Endurance Group Events

The host may configure asynchronous events to be triggered when certain events occur for an Endurance Group. The host submits a Set Features command specifying the Endurance Group Event Configuration feature (refer to section 5.1.25.1.16), the Endurance Group, and the specific event(s) that shall trigger adding an entry to the Endurance Group Event Aggregate log page (refer to section 5.1.12.1.15).

The host configures events using a Set Features command for each Endurance Group.

The host submits a Set Features command specifying the Asynchronous Event Configuration feature (refer to section 5.1.25.1.5) with the Endurance Group Event Aggregate Log Change Notices bit set to '1' to specify that adding an entry to the Endurance Group Event Aggregate log page shall trigger an Endurance Group Event Aggregate Log Page Change Notice event to the host (refer to Figure 409).

The host determines the Endurance Groups that have outstanding events by reading the Endurance Group Event Aggregate log page. An entry is returned for each Endurance Group that has an event outstanding. The host may use the Endurance Group Identifier Maximum value reported in the Identify Controller data structure to determine the maximum size of this log page.

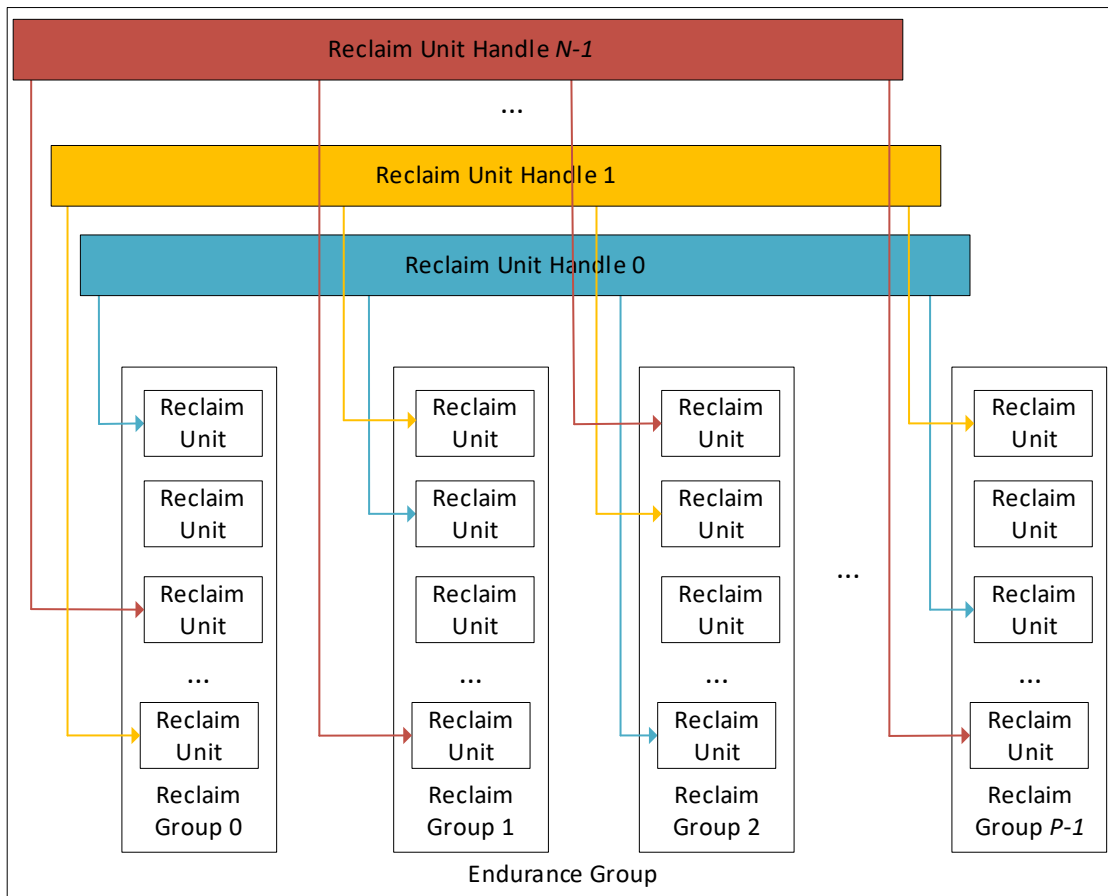
To determine the specific event(s) that have occurred for a reported Endurance Group, the host reads the Endurance Group Information log page (refer to Figure 218) for that Endurance Group. The Critical Warning field indicates the event(s) that have occurred (e.g., that all namespaces in the Endurance Group have been placed in read-only mode). All events for an Endurance Group are cleared if the controller successfully processes a read for the Endurance Group Information log page for that Endurance Group, where the Get Log Page command has the Retain Asynchronous Event bit cleared to '0'. If the Critical Warning field in the Endurance Group Information log page is cleared to 0h, then events for that Endurance Group are not reported in the Endurance Group Event Aggregate log page.

### 3.2.4 Reclaim Groups, Reclaim Unit Handles, and Reclaim Units

If Flexible Data Placement is enabled in an Endurance Group (refer to section 5.1.25.1.20), then the logical view of the non-volatile storage capacity in that Endurance Group is shown in Figure 70 and consists of:

- a set of one or more Reclaim Groups numbered from 0 to  $P-1$  where  $P$  is the value of the Number of Reclaim Groups field in the FDP Configuration Descriptor (refer to Figure 280). A Reclaim Group consists of one or more Reclaim Units; and
- one or more Reclaim Unit Handles numbered from 0 to  $N-1$  where  $N$  is the value of the Number of Reclaim Unit Handles field in the FDP Configuration Descriptor (refer to Figure 280).

A Reclaim Unit Handle consists of a reference to a Reclaim Unit in each Reclaim Group where user data from a write command is placed. A Reclaim Unit referenced by the Reclaim Unit Handle is only allowed to be referenced by at most one Reclaim Unit Handle. However, a specific Reclaim Unit is referenced by the same or different Reclaim Unit Handles as the Reclaim Unit is cycled from erased and back into use. When a Reclaim Unit is written to capacity, the controller updates that Reclaim Unit Handle to reference a different Reclaim Unit that is available for writing user data (e.g., non-volatile storage media that has been erased which is required prior to writing for program in place memories) and has not been written with any user data (i.e., an empty Reclaim Unit). Refer to section 8.1.10 for the details of how a host is able to issue a write command and place the user data into a Reclaim Unit.

**Figure 70: Flexible Data Placement Logical View of Non-Volatile Storage**

### 3.2.5 Domains and Divisions

#### 3.2.5.1 Overview

An NVM subsystem may be made up of a single domain or multiple domains (i.e., two or more). A domain is the smallest indivisible unit that shares state (e.g., power state, capacity information). An NVM subsystem that supports multiple domains shall support Asymmetric Namespace Access Reporting (refer to section 8.1.1).

A common example of a simple implementation of an NVM subsystem is one that consists of a single domain (i.e., multiple domains are not supported).

Each domain is independent, and the boundaries between domains are communication boundaries (e.g., fault boundaries, management boundaries). If multiple domains are present in an NVM subsystem, then those domains cooperate in the operation of that NVM subsystem. If a domain is unable to cooperate in the operation of the NVM subsystem, then the NVM subsystem has become divided.

A division is an event (e.g., failure of a domain) or action (e.g., management action or reconfiguration) within the NVM subsystem that affects communication between the domains contained in the NVM subsystem (refer to Figure 71 and Figure 72). If a division exists, global state within the NVM subsystem may be impacted (e.g., a controller may only have information about the state of the domains with which the controller is able to communicate). A division event or action may:

- affect access to namespaces (refer to section 8.1.1); or
- impact operations that have NVM subsystem scope (e.g., TNVMCAP, sanitize, format, SMART information).

A domain is comprised of:

- zero or more controllers; and
- zero or more NVM Endurance Groups.

If an NVM subsystem supports multiple domains, then all controllers in that NVM subsystem shall:

- set the MDS bit to '1' in the CTRATT field in the Identify Controller data structure (refer to Figure 312);
- set the Domain Identifier in each Endurance Group descriptor, if supported, to a non-zero value; and
- set the Domain Identifier in each Identify Controller data structure to a non-zero value.

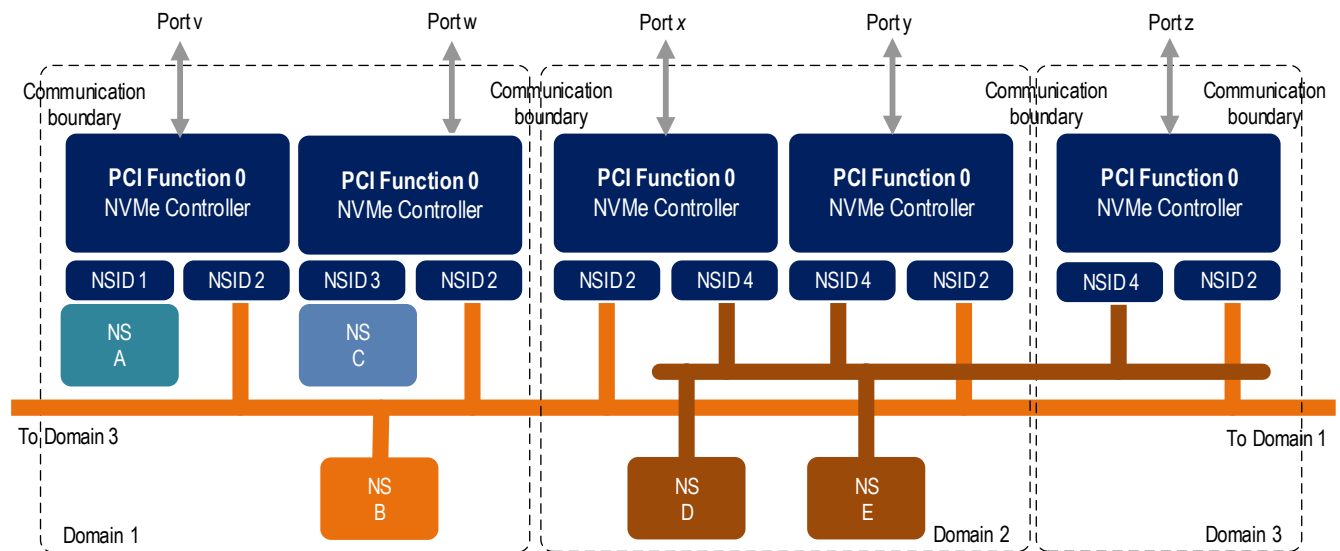
If an NVM subsystem supports multiple domains, then controllers in that NVM subsystem may:

- support Endurance Groups (refer to Endurance Groups bit in the CTRATT field of Identify Controller data structure).

For an NVM subsystem that supports multiple domains, each domain shall be assigned a domain identifier that is unique within the NVM subsystem (refer to the Domain Identifier field in Figure 312 and section 3.2.5.3). For an NVM subsystem that does not support multiple domains, Domain Identifier fields are cleared to 0h.

Figure 71 shows an example of an NVM subsystem that consists of three domains. Domain 1 contains two controllers and some amount of NVM storage capacity which has been allocated to two private namespaces (i.e., NS A and NS C) and a shared namespace (i.e., NS B). Domain 2 contains two controllers and some amount of NVM storage capacity which has been allocated to two shared namespaces (i.e., NS D and NS E). Domain 3 contains one controller, and no NVM storage capacity.

**Figure 71: Example 1 Domain Structure**

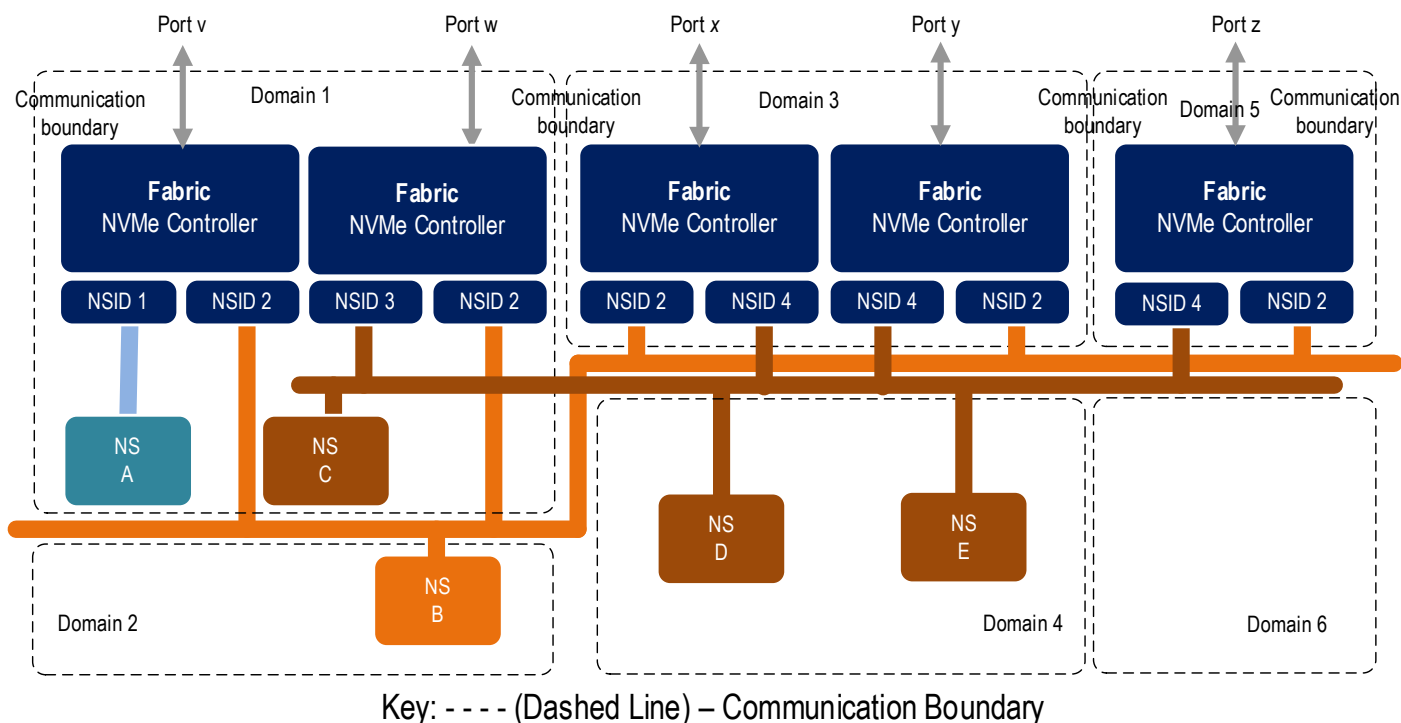


If, in the example shown in Figure 71, a division event occurs that results in Domain 1 no longer being able to communicate with Domain 2 and Domain 3, then the NVM subsystem would consist of two parts. The first part consists of Domain 1 and the second part consists of Domain 2 and Domain 3.

Figure 72 shows an example of an NVM subsystem that consists of six domains, of which, three are domains that contain controllers. Domain 1 is a domain that contains two controllers and some amount of NVM storage capacity from which NVM Endurance Groups have been created that contain a private namespace (i.e., NS A) and a shared namespace (i.e., NS C). Domain 2 is a domain that contains no controllers and contains some amount of NVM storage capacity from which NVM Endurance Groups have been created that contain a shared namespace (i.e., NS B). Domain 3 is a domain that contains two

controllers and no NVM storage capacity. Domain 4 is a domain that contains no controllers and contains some amount of NVM storage capacity from which NVM Endurance Groups have been created that contain two shared namespaces (i.e., NS D and NS E). Domain 5 is a domain that contains one controller and no NVM storage capacity. Domain 6 is a domain that contains no controllers and no NVM storage capacity allocated to an NVM Endurance Group (i.e., an empty domain).

**Figure 72: Example 2 Domain Structure**



### 3.2.5.2 Domains and Reservations

If an NVM subsystem supports multiple domains and Persistent Reservations (refer to section 8.1.22), then resumption after a division event (e.g., resumption of operation, resumption of communication) requires that all persistent reservation state within the domains in the NVM subsystem that are no longer divided be synchronized (i.e., updated).

If the reservation state for a namespace is not synchronized, then the ANA Group that contains that namespace shall transition to the ANA Inaccessible state (refer to section 8.1.1.6) and remain in that state until the Persistent Reservation state is synchronized. If the Persistent Reservation state is not able to be synchronized, then:

- a transition to the ANA Persistent Loss state occurs and commands are processed as described in section 8.1.1.7; or
- the controller may stop processing commands and set the Controller Fatal Status (CSTS.CFS) bit to '1' (refer to section 9.5).

### 3.2.5.3 Domain Identifier Use (Informative)

Domain Identifier values indicate the parts of the NVM subsystem that comprise a domain.

The host may use these values to determine which Endurance Groups (refer to section 3.2.3) are contained in the same domain and which are contained in a different domain. Examples of host use of the domain identifier include:

- host data redundancy software (e.g., RAID) that may use the Endurance Group's Domain Identifier to determine which Endurance Groups may fail together (e.g., Endurance Groups in the same domain) and which Endurance Groups may fail independently (e.g., Endurance Groups in different domains); and
- host application software may use the controller's Domain Identifier to determine which controllers share domains (e.g., controllers that may fail together) and which controllers are a part of different domains (e.g., controllers that may fail independently).

### 3.3 NVM Queue Models

The NVM Express interface is based on a paired Submission and Completion Queue mechanism. Commands are placed by host software into a Submission Queue. Completions are placed into the associated Completion Queue by the controller. When using a memory-based transport queue model (refer to section 3.3.1), multiple Submission Queues may utilize the same Completion Queue. When using a message-based transport queue model (refer to section 3.3.2) each Submission Queue maps to a single Completion Queue.

#### 3.3.1 Memory-based Transport Queue Model (PCIe)

##### 3.3.1.1 Queue Setup and Initialization

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

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

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

##### 3.3.1.2 Queue Usage

The submitter of entries to a memory-based transport queue uses the current Tail entry pointer to identify the next open queue slot. The submitter increments the Tail entry pointer after placing the new entry to the open queue slot. If the Tail entry pointer increment exceeds the queue size, the Tail entry shall roll to zero. The submitter may continue to place entries in free queue slots as long as the Full queue condition is not met (refer to section 3.3.1.5).

Note: The submitter shall take queue wrap conditions into account.

The consumer of entries on a memory-based transport queue uses the current Head entry pointer to identify the slot containing the next entry to be consumed. The consumer increments the Head entry pointer after consuming the next entry from the queue. If the Head entry pointer increment exceeds the queue size, the Head entry pointer shall roll to zero. The consumer may continue to consume entries from the queue as long as the Empty queue condition is not met (refer to section 3.3.1.4).

Note: The consumer shall take queue wrap conditions into account.

Creation and deletion of memory-based transport Submission Queue and associated Completion Queues are required to be ordered correctly by host software. Host software creates the Completion Queue before creating any associated Submission Queue. Submission Queues may be created at any time after the associated Completion Queue is created. Host software deletes all associated Submission Queues prior to deleting a Completion Queue. To abort all commands submitted to the Submission Queue host software issues a Delete I/O Submission Queue command for that queue (refer to section 3.3.1.3).

Host software writes the Submission Queue Tail Doorbell and the Completion Queue Head Doorbell (refer to the Transport Specific Controller Properties section in the NVMe over PCIe Transport Specification) to communicate new values of the corresponding entry pointers to the controller. If host software writes an invalid value to the Submission Queue Tail Doorbell or Completion Queue Head Doorbell property and an Asynchronous Event Request command is outstanding, then an asynchronous event is posted to the Admin Completion Queue with a status code of Invalid Doorbell Write Value. The associated queue is then deleted and recreated by host software. For a Submission Queue that experiences this error, the controller may complete previously consumed commands; no additional commands are consumed. This condition may be caused by host software attempting to add an entry to a full Submission Queue or remove an entry from an empty Completion Queue.

Host software checks completion queue entry Phase Tag (P) bits in memory to determine whether new completion queue entries have been posted (refer to section 4.2.4). The Completion Queue Tail pointer is only used internally by the controller and is not visible to the host. The controller uses the SQ Head Pointer (SQHD) field in completion queue entries to communicate new values of the Submission Queue Head Pointer to the host. A new SQHD value indicates that submission queue entries have been consumed, but does not indicate either execution or completion of any command. Refer to section 4.2.

A submission queue entry is submitted to the controller when the host writes the associated Submission Queue Tail Doorbell with a new value that indicates that the Submission Queue Tail Pointer has moved to or past the slot in which that submission queue entry was placed. A Submission Queue Tail Doorbell write may indicate that one or more submission queue entries have been submitted.

A submission queue entry has been consumed by the controller when a completion queue entry is posted that indicates that the Submission Queue Head Pointer has moved past the slot in which that submission queue entry was placed. A completion queue entry may indicate that one or more submission queue entries have been consumed.

A completion queue entry is posted to the Completion Queue when the controller write of that completion queue entry to the next free Completion Queue slot inverts the Phase Tag (P) bit from its previous value in memory (refer to section 4.2.4). The controller may generate an interrupt to the host to indicate that one or more completion queue entries have been posted.

A completion queue entry has been consumed by the host when the host writes the associated Completion Queue Head Doorbell with a new value that indicates that the Completion Queue Head Pointer has moved past the slot in which that completion queue entry was placed. A Completion Queue Head Doorbell write may indicate that one or more completion queue entries have been consumed.

Once a submission queue entry or a completion queue entry has been consumed, the slot in which it was placed is free and available for reuse. Altering a submission queue entry after that entry has been submitted but before that entry has been consumed results in undefined behavior. Altering a completion queue entry after that entry has been posted but before that entry has been consumed results in undefined behavior.

#### **3.3.1.2.1 Completion Queue Flow Control**

If there are no free slots in a Completion Queue, then the controller shall not post status to that Completion Queue until slots become available. In this case, the controller may stop processing additional submission queue entries associated with the affected Completion Queue until slots become available. The controller shall continue processing for other Submission Queues not associated with the affected Completion Queue.



### 3.3.1.3 Queue Abort

To abort a large number of commands, the host may use:

- the Cancel command (refer to section 7.1); or
- delete and recreate the I/O Submission Queue (refer to section 3.7.3).

Specifically, to abort all commands that are submitted to an I/O Submission Queue, host software should:

- issue a Cancel command to that queue with the Cancel Action set to Multiple Command Cancel and the NSID field set to FFFFFFFFh; or
- issue a Delete I/O Submission Queue command for that queue. After that submission queue has been successfully deleted, indicating that all commands have been completed or aborted, then host software should recreate the queue by submitting a Create I/O Submission Queue command. Host software may then re-submit commands to the associated I/O Submission Queue.

If the host is no longer able to communicate with the controller before that host receives either:

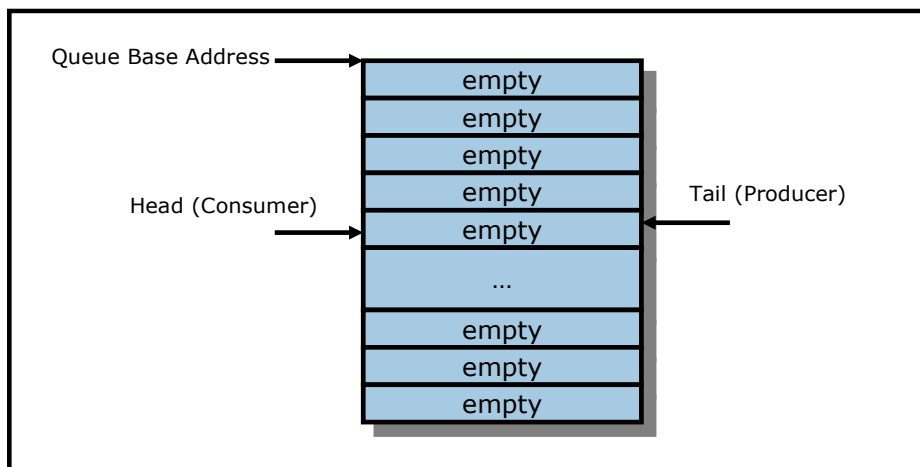
- completions for all outstanding commands submitted on that I/O Submission Queue (refer to section 3.4.5); or
- a successful completion for the Delete I/O Submission Queue command for that I/O Submission Queue,

then it is strongly recommended that the host take the steps described in section 9.6 to avoid possible data corruption caused by interaction between outstanding commands and subsequent commands submitted by that host to another controller.

### 3.3.1.4 Empty Queue

The queue is Empty when the Head entry pointer equals the Tail entry pointer. Figure 73 defines the Empty Queue condition.

**Figure 73: Empty Queue Definition**

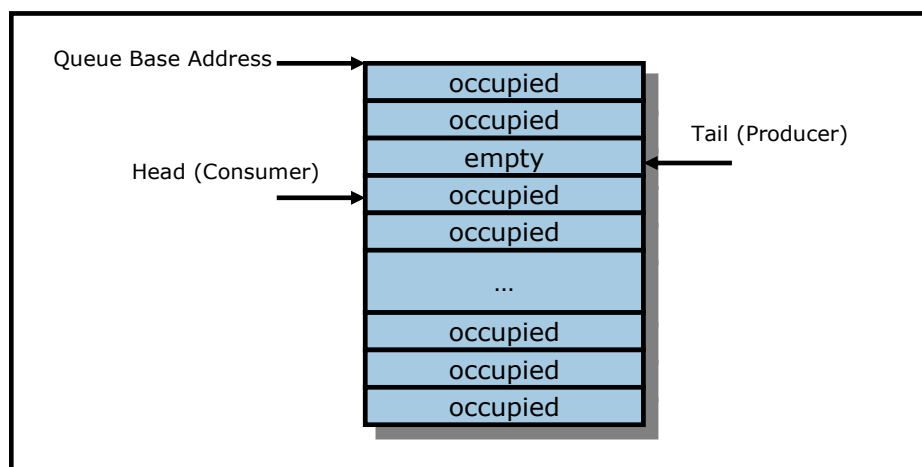


### 3.3.1.5 Full Queue

The queue is Full when the Head equals one more than the Tail. The number of entries in a queue when full is one less than the queue size. Figure 74 defines the Full Queue condition.

Note: Queue wrap conditions shall be taken into account when determining whether a queue is Full.

Figure 74: Full Queue Definition



### 3.3.2 Message-based Transport Queue Model (Fabrics)

For NVMe over Fabrics, a queue is a unidirectional communication channel that is used to send capsules between a host and a controller. A host uses Submission Queues to send command capsules (refer to section 3.3.2.1.1) to a controller. A controller uses Completion Queues to send response capsules (refer to section 3.3.2.1.2) to a host. Submission and Completion Queues are created in pairs using the Connect command (refer to section 3.3.2.2).

The NVMe Transport is responsible for delivering command capsules to the controller and notifying the controller of capsule arrival in a transport-specific fashion.

Altering a command capsule between host submission to the Submission Queue and transport delivery of that capsule to the controller results in undefined behavior.

NVMe Transports are not required to provide any additional end-to-end flow control. Specific NVMe Transports may require low level flow control for congestion avoidance and reliability; any such additional NVMe Transport flow control is outside the scope of this specification.

Flow control differs for Submission Queues (refer to section 3.3.2.1.1, section 3.3.2.6, and section 3.3.2.7) and Completion Queues (refer to section 3.3.2.1.2, section 3.3.2.8, and section 3.3.1.2.1).

#### 3.3.2.1 Capsules and Data Transfers

This section describes capsules and data transfer mechanisms necessary to support message-based transport queues. These mechanisms are used for Fabrics commands, Admin commands, and I/O commands when using the message-based transport queue model.

A capsule is an NVMe unit of information exchanged between a host and a controller. A capsule may contain commands, responses, SGLs, and/or data. The data may include user data (e.g., logical block data and metadata that is transferred as a contiguous part of the logical block) and data structures associated with the command.

The capsule size for the Admin Queue commands and responses is fixed and defined in the NVMe Transport binding specification. The controller indicates in the Identify Controller data structure the capsule command and response sizes that the host shall use with I/O commands.

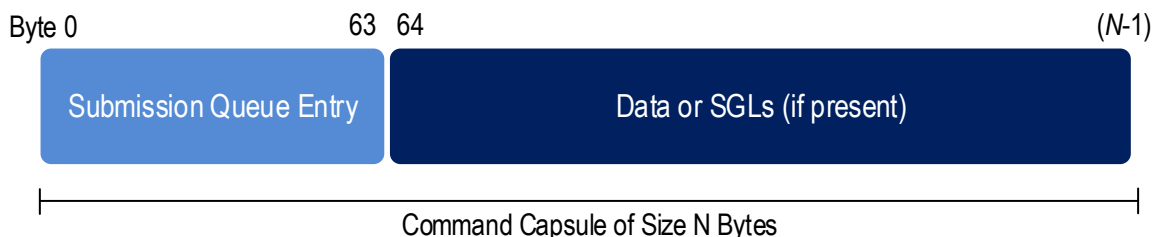
The controller shall support SGL based data transfers for commands on both the Admin Queue and I/O Queues. Data may be transferred within the capsule or through memory transactions based on the underlying NVMe Transport as indicated in the SGL descriptors associated with the command capsule. The SGL types supported by an NVMe Transport are specified in the NVMe Transport binding specification.

The value of unused and not reserved capsule fields (e.g., the capsule is larger than the command / response and associated data) is undefined and shall not be interpreted by the recipient.

### 3.3.2.1.1 Command Capsules

A command capsule is sent from a host to a controller. It contains a submission queue entry (SQE) and may optionally contain data or SGLs. The SQE is 64 bytes in size and contains the Admin command, I/O command, or Fabrics command to be executed.

**Figure 75: Command Capsule**



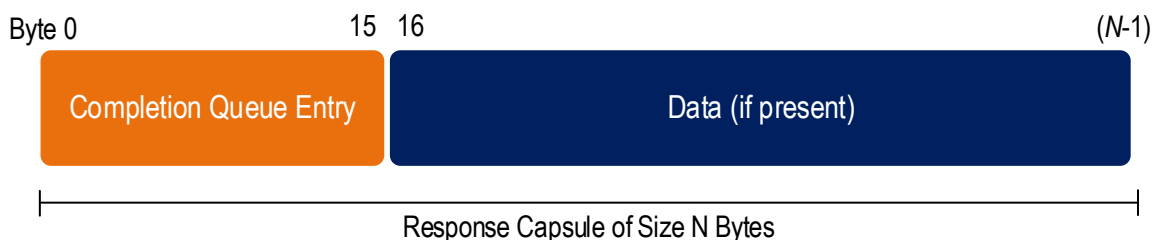
The Command Identifier field in the SQE shall be unique among all outstanding commands associated with that queue. If there is data or additional SGLs to be transferred within the capsule, then the SGL descriptor in the SQE contains a Data Block, Segment Descriptor, or Last Segment Descriptor specifying an appropriate Offset address. The definition for the submission queue entry when the command is a Fabrics command is shown in section 4.1.2. The definition for the submission queue entry when the command is an Admin command or I/O command is shown in section 4.1.1. Bytes 03:00 share a common format across commands.

### 3.3.2.1.2 Response Capsules

A response capsule is sent from the NVM subsystem to the host. It contains a completion queue entry (CQE) and may optionally contain data. The CQE is the completion queue entry associated with a previously issued command capsule.

If a command requests data and the SGL in the associated command capsule specifies a Data Block descriptor with an Offset, the data is included in the response capsule. If the SGL(s) in the command capsule specify a region in host memory, then data is transferred via memory transactions.

**Figure 76: Response Capsule**



The completion queue entry is 16 bytes in size and contains a two byte status field.

The definition for the completion queue entry for a Fabrics command is shown in section 4.2.2. The definition for the completion queue entry when the command is an Admin command or I/O command is defined in section 4.2.1, where the SQ Identifier and Phase Tag fields are reserved because they are not used in NVMe over Fabrics. Use of the SQHD field depends on whether SQ flow control is disabled for the queue pair, refer to section 6.3.

### 3.3.2.1.3 Data Transfers

Data may be transferred within capsules or by memory transfers. SGLs are used to specify the location of data. Metadata, if transferred, is a contiguous part of the user data with which that metadata is associated. The SGL descriptor(s) (refer to section 4.3.2) specify whether the command's data is transferred through memory or within the capsule. The capsule may contain either SGLs or data (not a mixture of both) following

the SQE. If additional SGLs are required, then the SGLs are included in the capsule immediately after the SQE. If an invalid offset is specified in an SGL descriptor, then a status code of SGL Offset Invalid shall be returned.

SGLs shall be supported within a capsule. The NVMe Transport binding specification defines the SGL Descriptor Types and Sub Types that are supported for the corresponding NVMe Transport. The NVMe Transport binding specification also specifies if SGLs may be supported in host memory.

#### **3.3.2.1.3.1 Data and SGL Locations within a Command Capsule**

The submission queue entry within the command capsule includes one SGL entry. If there are additional SGL entries to be transferred in the command capsule, then those entries shall be contiguous and located immediately after the submission queue entry.

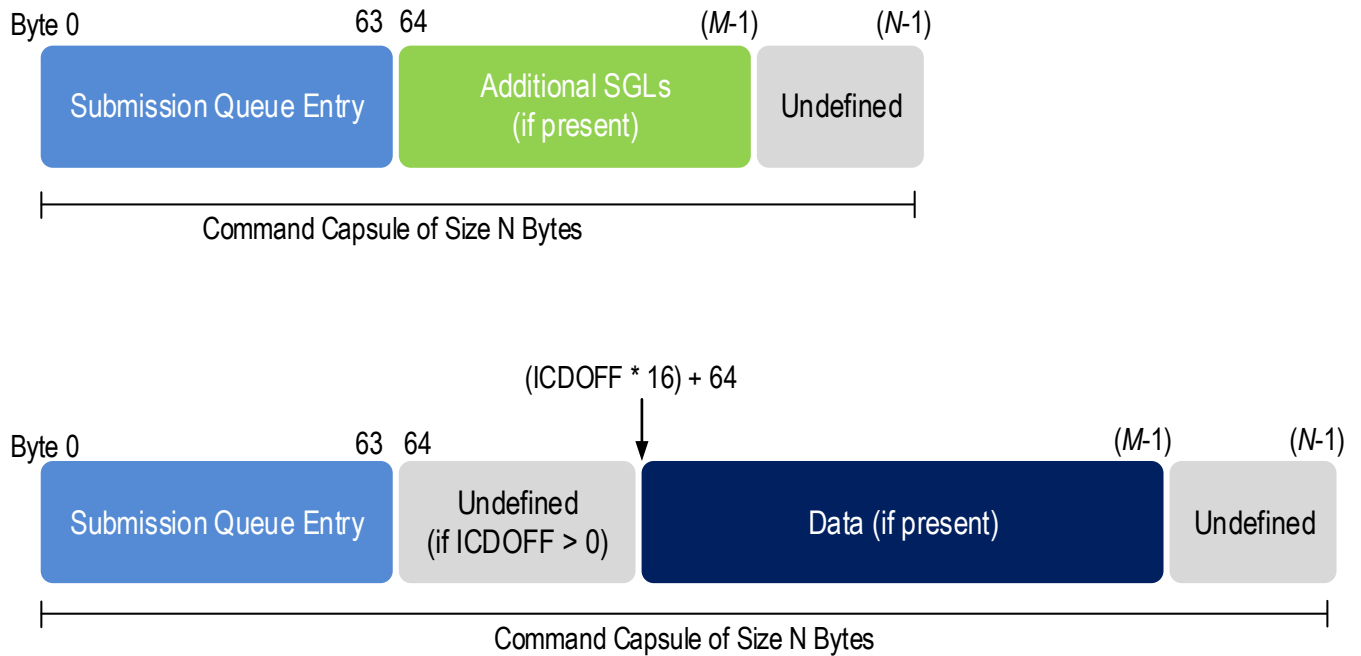
An NVMe Transport binding specification defines the support for data as part of the command capsule. The controller indicates the starting location of data within a command capsule via the In Capsule Data Offset (ICDOFF) field in the Identify Controller data structure.

There are restrictions for SGLs that the host should follow:

- if the ICDOFF field is a non-zero value, then all SGL descriptors following the submission queue entry shall not have a total size greater than  $(ICDOFF * 16)$ ;
- if the SGL descriptors following the submission queue entry have a total size greater than  $(ICDOFF * 16)$ , then the controller shall abort the command with a status code of Invalid Number of SGL Descriptors;
- the host shall not place more SGL Data Block or Keyed SGL Data Block descriptors within a capsule than the maximum indicated in the Identify Controller data structure; and
- if the host places more SGL Data Block or Keyed SGL Data Block descriptors in a capsule than the maximum indicated in the Maximum SGL Data Block Descriptors field in the Identify Controller data structure, then the controller shall abort the command with a status code of Invalid Number of SGL Descriptors.

The host shall start data (if present) in command capsules at byte offset  $(ICDOFF * 16)$  from the end of the submission queue entry.

**Figure 77: Data and SGL Locations within a Command Capsule**

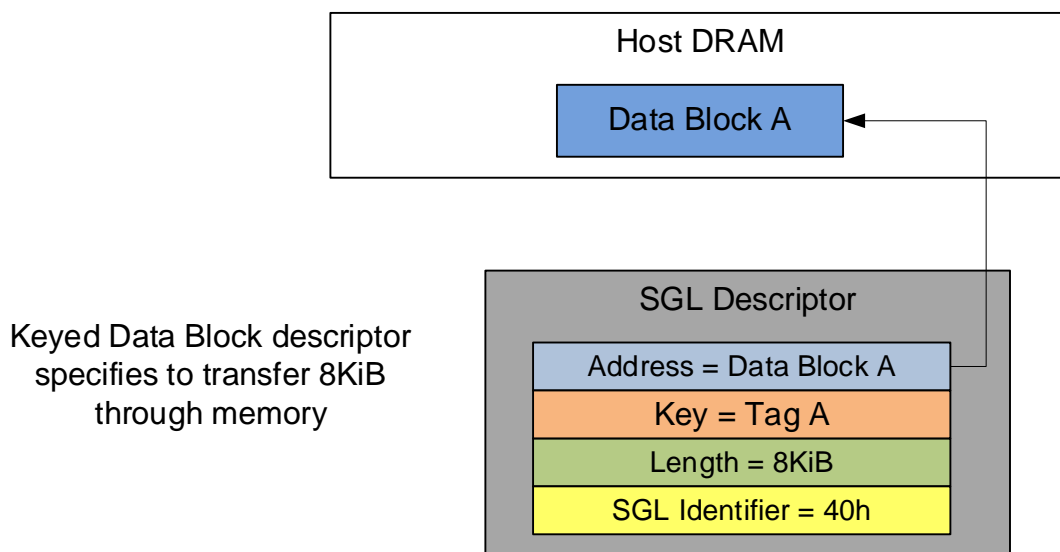


### 3.3.2.1.3.2 Data Transfer Examples

The data transfer examples in Figure 78 and Figure 79 show SGL examples for a Write command where data is transferred via a memory transaction or within the capsule. The SGL may use a key as part of the data transfer depending on the requirements of the NVMe Transport used.

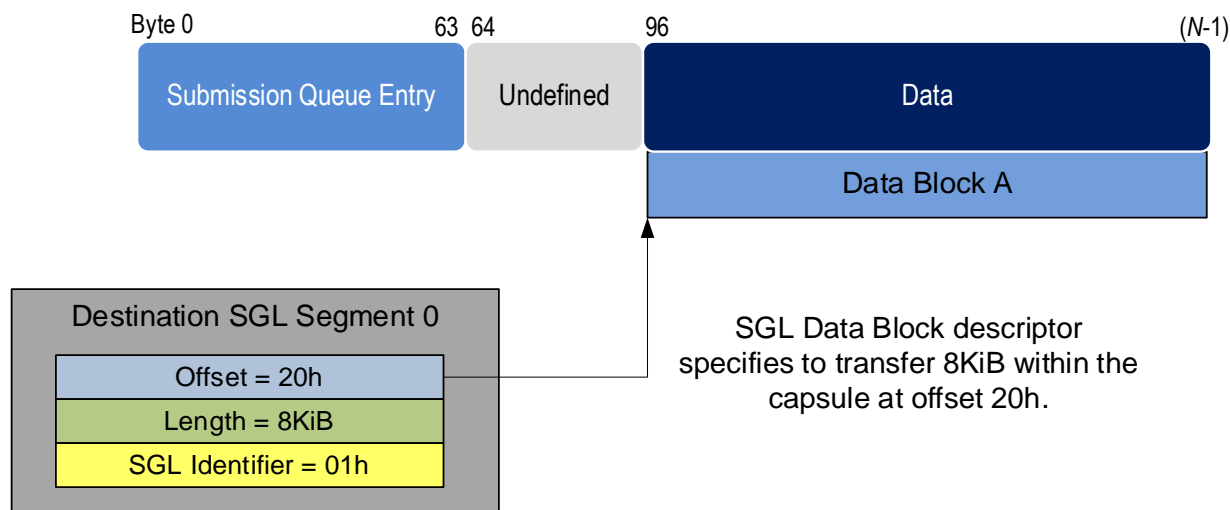
The first example shows an 8KiB write where all of the data is transferred via memory transactions. In this case, there is one SGL descriptor that is contained within the submission queue entry at CMD.SGL1. The SGL descriptor is a Keyed SGL Data Block descriptor. If more SGLs are required to complete the command, the additional SGLs are contained in the command capsule.

**Figure 78: SGL Example Using Memory Transactions**



The second example shows an 8KiB write where all of the data is transferred within the capsule. In this case, the SGL descriptor is an SGL Data Block descriptor specifying an Offset of 20h based on an ICDOFF value of 2h.

**Figure 79: SGL Example Using In Capsule Data Transfer**



### 3.3.2.2 Queue Creation

Message-based controllers use the Connect command (refer to section 6.3) to create 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 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 (i.e., Create I/O Completion Queue, Create I/O Submission Queue, Delete I/O Completion Queue, Delete I/O Submission Queue).

An NVMe Transport connection is established between a host and an NVM subsystem prior to the transfer of any capsules or data. The mechanism used to establish an NVMe Transport connection is NVMe Transport specific and defined by the corresponding NVMe Transport binding specification. The NVMe Transport may require a separate NVMe Transport connection for each Admin Queue or I/O Queue or may utilize the same NVMe Transport connection for all Admin and I/O Queues associated with a particular controller. An NVMe Transport may also require that NVMe layer information be passed between the host and controller in the process of establishing an NVMe Transport connection (e.g., exchange queue size to appropriately size send and receive buffers).

The Connect command specifies the Queue ID and type (Admin or I/O), the size of the Submission and Completion Queues, queue attributes, Host NQN, NVM Subsystem NQN, and Host Identifier. The Connect command may specify a particular controller if the NVM subsystem supports a static controller model. The Connect response indicates whether the connection was successfully established as well as whether NVMe in-band authentication is required.

The Connect command is submitted to the same Admin Queue or I/O Queue that the Connect command creates. The underlying NVMe Transport connection that is used for that queue is created first and the Connect command and response capsules are sent over that NVMe Transport connection. The Connect command shall be sent once to a queue.

When a Connect command successfully completes, the corresponding Submission and Completion Queues are created. If NVMe in-band authentication is required as indicated in the Connect response, then

NVMe in-band authentication shall be performed before the queues may be used to perform other Fabrics commands, Admin commands, or I/O commands.

Once a Connect command for an Admin Queue has completed successfully (and NVMe in-band authentication, if required, has succeeded), only Fabrics commands may be submitted until the controller is ready (CSTS.RDY = 1). Both Fabrics commands and Admin commands may be submitted to the Admin Queue while the controller is ready. A Connect command for an I/O Queue may be submitted after the controller is ready. Once a Connect command for an I/O Queue has completed successfully (and NVMe in-band authentication, if required, has succeeded), I/O commands may be submitted to the queue.

The Connect response contains the controller ID allocated to the host.

After an Admin Queue is created on a controller, all subsequent Connect commands sent from the same host to that controller, to create an I/O Queue, are required to:

- utilize the same NVMe Transport;
- have the same Host NQN;
- have the same NVM Subsystem NQN; and
- either have the:
  - same Host Identifier value; or
  - a Host Identifier value of 0h, if supported (refer to section 5.1.12.3.1).

### 3.3.2.3 Queue Initialization and Queue State

When a Connect command successfully completes, the corresponding Admin Submission and Completion Queue or I/O Submission and Completion Queues are created. If the host sends a Connect command specifying the Queue ID of a queue which already exists, then the controller shall abort the command with a status code of Command Sequence Error.

The Authentication Requirements (AUTHREQ) field in the Connect response indicates if NVMe in-band authentication is required. If AUTHREQ is cleared to 0h, the created queue is ready for use after the Connect command completes successfully. If AUTHREQ is set to a non-zero value, the created queue is ready for use after NVMe in-band authentication has been performed successfully using the Authentication Send and Authentication Receive Fabrics commands.

If a controller requires or is undergoing NVMe in-band authentication for a queue pair, then a controller shall abort all commands received on that queue other than authentication commands with a status code of Authentication Required. After the NVMe in-band authentication has been performed successfully on a queue, then a controller shall abort all authentication commands on that queue with a status code of Command Sequence Error.

When an Admin Queue is first created, the associated controller is disabled (i.e., CC.EN is initialized to '0'). A disabled controller shall abort all commands other than Fabrics commands on the Admin Queue with a status code of Command Sequence Error. After the controller is enabled, it shall accept all supported Admin commands in addition to Fabrics commands.

A created I/O queue shall abort all commands with a status code of Command Sequence Error if the associated controller is disabled.

### 3.3.2.4 I/O Queue Deletion

NVMe over Fabrics deletes an individual I/O Queue and may delete the associated NVMe Transport connection as a result of:

- the exchange of a Disconnect command and response (refer to section 6.4) between a host and controller; or
- the detection and processing of a transport error on an NVMe Transport connection.

The host indicates support for the deletion of an individual I/O Queue by setting the Individual I/O Queue Deletion Support (INDIVIOQDELS) bit to '1' in the CATTR field in the Connect command (refer to Figure 545) used to create the Admin Queue. The controller indicates support for the deletion of an individual I/O

Queue by setting the Disconnect Command Support (DCS) bit to '1' in the OFCS field of the Identify Controller data structure (refer to Figure 312).

If both the host and the controller support deletion of an individual I/O Queue, then the termination of an individual I/O Queue impacts only that I/O Queue (i.e., the association and all other I/O Queues and their associated NVMe Transport connections are not impacted). If either the host or the controller does not support deletion of an individual I/O Queue, then the deletion of an individual I/O Queue or the termination of an NVMe Transport connection causes the association to be terminated.

NVMe over Fabrics uses the Disconnect command to delete an Individual I/O Queue. This command is sent on the I/O Submission Queue to be deleted and affects only that I/O Submission Queue and its associated I/O Completion Queue (i.e., other I/O Queues are not affected). To delete an I/O Queue, the NVMe Transport connection for that I/O Queue is used. If all Queues associated with an NVMe Transport connection are deleted, then the NVMe Transport connection may be deleted after completion of the Disconnect command. Actions necessary to delete the NVMe Transport connection are transport specific. The association between the host and the controller is not affected.

If a Disconnect command returns a status code other than success, the host may delete an I/O Queue using other methods including:

- waiting a vendor specific amount of time and retry the Disconnect command;
- deleting the NVMe Transport connection (note: this may impact other I/O Queues);
- performing a Controller Level Reset (note: this impacts other I/O Queues); or
- ending the host to controller association.

If the transport requires a separate NVMe Transport connection for each Admin and I/O Queue (refer to section 3.3.2.2), then the host should not delete an NVMe Transport connection until after:

- a Disconnect command has been submitted to the I/O Submission Queue; and
- the response for that Disconnect command has been received by the host on the corresponding I/O Completion Queue or a vendor specific timeout (refer to section 3.9) has occurred while waiting for that response.

If the transport requires a separate NVMe Transport connection for each Admin and I/O Queue, then the controller should not delete an NVMe Transport connection until after:

- a Disconnect command has been received on the I/O Submission Queue and processed by the controller;
- the responses for commands received by the controller on that I/O Submission Queue prior to receiving the Disconnect command have been sent to the host on the corresponding I/O Completion Queue; and
- the resulting response for that Disconnect command has been sent to the host on the corresponding I/O Completion queue (i.e., this response is the last response sent). It is recommended that the controller delay destroying the NVMe Transport connection to allow time for the Disconnect command response to be received by the host (e.g., a transport specific event occurs or a transport specific time period elapses).

If the transport utilizes the same NVMe Transport connection for all Admin and I/O Queues associated with a particular controller (refer to section 3.3.2.2), then the deletion of an individual I/O Queue has no impact on the NVMe Transport connection.

A Disconnect command is the last I/O Submission Queue entry processed by the controller for an I/O Queue. Controller processing of the Disconnect command completes or aborts all commands on the I/O Queue on which the Disconnect command was received. The controller determines whether to complete or abort each of those commands. Until the controller sends a successful completion for a Disconnect command, outstanding commands may continue being processed by the controller. The controller ensures that there is no further processing of any command sent on that I/O Queue after posting the completion queue entry for the Disconnect command as described in section 6.4.



The response to the Disconnect command is the last I/O Completion Queue entry processed by the host for an I/O Queue. To avoid command aborts, the host should wait for all outstanding commands on an I/O Queue to complete before sending the Disconnect command.

If the controller terminates an NVMe Transport connection or detects an NVMe Transport connection loss, then the controller shall stop processing all commands received on I/O Queues associated with that NVMe Transport connection within the time reported in the CQT field (refer to Figure 312), if non-zero.

If the host terminates an NVMe Transport connection or detects an NVMe Transport connection loss before the responses are received for all outstanding commands submitted to the associated I/O Queue (refer to section 3.4.5), then it is strongly recommended that the host take the steps described in section 9.6 to avoid possible data corruption caused by interaction between outstanding commands and subsequent commands submitted by that host to another controller.

### 3.3.2.5 Submission Queue Flow Control Negotiation

Use of Submission Queue (SQ) flow control is negotiated for each queue pair by the Connect command and the controller response to the Connect command. SQ flow control shall be used unless it is disabled as a result of that negotiation. If SQ flow control is disabled, then the Submission Queue Head Pointer (SQHD) field is reserved in all Fabrics response capsules for that queue pair after the response to the Connect command (i.e., in all subsequent response capsules for that queue pair, the controller shall clear the SQHD field to 0h and the host should ignore the SQHD field).

If the host requests that SQ flow control be disabled for a queue pair, then the host should size each Submission Queue to support the maximum number of commands that the host could have outstanding at one time for that Submission Queue.

The maximum size of the Admin Submission Queue is specified in the Admin Max SQ Size (ASQSZ) field of the Discovery Log Page Entry for the NVM subsystem (refer to section 5.1.12.3.1).

The maximum size of an I/O Submission Queue is specified in the Maximum Queue Entries Supported (MQES) field of the Controller Capabilities (CAP) property for the controller (refer to section 3.1.4.1).

The value of the Maximum Outstanding Commands (MAXCMD) field in the Identify Controller data structure indicates the maximum number of commands that the controller processes at one time for a particular I/O Queue. The host may use this value to size I/O Submission Queues and optimize the number of commands submitted at one time per queue to achieve the best performance.

If SQ flow control is disabled, then the host should limit the number of outstanding commands for a queue pair to be less than the size of the Submission Queue. If the controller detects that the number of outstanding commands for a queue pair is greater than or equal to the size of the Submission Queue, then the controller shall:

- a) stop processing commands and set the Controller Fatal Status (CSTS.CFS) bit to '1' (refer to section 9.5); and
- b) terminate the NVMe Transport connection and end the association between the host and the controller.

### 3.3.2.6 Submission Queue Flow Control

This section applies only to Submission Queues that use SQ flow control.

The Submission Queue has a Head entry pointer and a Tail entry pointer that are used to manage the queue and determine the number of Submission Queue capsules available to the host for new submissions. The Head and Tail entry pointers are initialized to 0h when a queue is created. All arithmetic operations and comparisons on entry pointers are performed modulo the queue size with queue wrap conditions taken into account. The host increments the Tail entry pointer when the host adds a capsule to a queue. The controller increments the Head entry pointer when that controller removes a capsule from the queue.

The Submission Queue Head entry pointer is maintained by the controller and is communicated to the host in the SQHD field of completion queue entries. The host uses the received SQHD values for Submission Queue management (e.g., to determine whether the Submission Queue is full).

The Submission Queue Tail entry pointer is local to the host and is not communicated to the controller.

The Submission Queue is full when the Head entry pointer equals one more than the Tail entry pointer (i.e., incrementing the Tail entry pointer has caused it to wrap around to just behind the Head entry pointer). A full Submission Queue contains one less capsule than the queue size. A host may continue to submit commands to a Submission Queue as long as the queue is not full.

If the controller detects that the host has submitted a command capsule to a full Submission Queue, then the controller shall:

- a) stop processing commands and set the Controller Fatal Status (CSTS.CFS) bit to '1' (refer to section 9.5); and
- b) terminate the NVMe Transport connection and end the association between the host and the controller.

The Submission Queue is empty when the Head entry pointer equals the Tail entry pointer.

### 3.3.2.7 Submission Queue Head Pointer Update Optimization

Submission Queue Head Pointer update optimization does not apply to queue pairs for which Submission Queue (SQ) flow control is disabled, as the SQHD field is reserved if SQ flow control is disabled, refer to section 3.3.2.5 and to section 6.3.

The NVMe Transport may omit transmission of the SQHD value for a response capsule that:

- a) contains a Generic Command status (i.e., Status Code Type 0h) indicating successful completion of a command (i.e., Status Code 00h);
- b) is not a Connect response capsule; and
- c) is not a Disconnect response capsule.

If a new SQHD value is not received in a response capsule, the host continues to use its previous SQHD value. Thus, at the NVMe layer there is a logical progression of SQHD values despite the fact that the NVMe Transport may not actually transfer the SQHD value in each response capsule.

The NVMe Transport may deliver response capsules that do not contain an SQHD value to the host in any order. The applicable NVMe Transport binding specification defines how presence versus absence of an SQHD value in a response capsule is indicated by the NVMe Transport.

Periodic SQHD updates at the host are required to avoid Submission Queue (SQ) starvation as SQHD value transmission in responses is the only means of releasing SQ slots for host reuse.

An NVMe Transport may transmit an SQHD value in every response capsule. If an NVMe Transport does not transmit an SQHD value in every response capsule, then an SQHD value should be transmitted periodically (e.g., in at least one of every  $n$  response capsules on a CQ, where  $n$  is 10% of the size of the associated SQ) or more often. An SQHD value should always be transmitted if 90% or more of the slots in the associated SQ are occupied at the subsystem.

### 3.3.2.8 Completion Queue Considerations

Completion Queue flow control (refer to section 3.3.1.2.1) is not used in the message-based transport queue model. Message-based transport Completion Queues do not use either Head entry pointers or Tail entry pointers.

The host should size each Completion Queue to support the maximum number of commands that the host could have outstanding at one time for a particular Submission Queue. The Completion Queue size may be larger than the size of the corresponding Submission Queue to accommodate responses for commands that are being processed by the controller in addition to responses for commands that are still in the Submission Queue.

If the size of a Completion Queue is too small for the number of outstanding commands and the controller submits a response capsule to a full Completion Queue, then the results are undefined.

The value of the Maximum Outstanding Commands (MAXCMD) field in the Identify Controller data structure indicates the maximum number of commands that the controller processes at one time for a particular I/O Queue. The host may use this value to size I/O Completion Queues and optimize the number of commands submitted at one time per queue to achieve the best performance.

Altering a response capsule between controller submission to the Completion Queue and transport delivery of that capsule to the host results in undefined behavior.

### 3.3.2.9 Transport Requirements

This section defines requirements that all NVMe Transports that support an NVMe over Fabrics implementation shall meet.

The NVMe Transport may support NVMe Transport error detection and report errors to the NVMe layer in command status values. The controller may record NVMe Transport specific errors in the Error Information log page (refer to section 5.1.12.1.2). Transport errors that cause loss of a message or loss of data in a way that the low-level NVMe Transport cannot replay or recover should cause:

- the deletion of the individual I/O Queues (refer to section 3.3.2.4) and the associated NVMe Transport connection on which that NVMe Transport level error occurred; or
- termination of the NVMe Transport connection and the association between the host and controller.

The NVMe Transport shall provide reliable delivery of capsules between a host and NVM subsystem (and allocated controller) over each connection. The NVMe Transport may deliver command capsules in any order on each queue except for I/O commands that are part of fused operations (refer to section 3.4.2).

For command capsules that are part of fused operations for I/O commands, the NVMe Transport:

1. shall deliver the first and second command capsules for each fused operation to the queue in-order; and
2. shall not deliver any other command capsule for the same Submission Queue between delivery of the two command capsules for a fused operation.

The NVMe Transport shall provide reliable delivery of response capsules from an NVMe subsystem to a host over each connection. The NVMe Transport shall deliver response capsules that include an SQ Head Pointer (SQHD) value to the host in-order; this includes all Connect response capsules and all Disconnect response capsules.

## 3.3.3 Queueing Attributes

### 3.3.3.1 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 294).

### 3.3.3.2 Queue Identifier

Each queue is identified through a 16-bit ID value that is assigned to the queue when it is created. Both I/O Submission Queue identifiers and I/O Completion Queue identifiers are a value from 1 to 65,535.

### 3.3.3.3 Queue Priority

If the weighted round robin with urgent priority class arbitration mechanism is supported, then host software may assign a queue priority service class of Urgent, High, Medium, or Low. If the weighted round robin with

urgent priority class arbitration mechanism is not supported, then the priority setting is not used and is ignored by the controller.

### 3.3.3.4 Queue Coordination

There is one Admin Queue pair associated with multiple I/O queue pairs. The Admin Submission Queue and Completion Queue are used to carry out functions that impact the entire controller. An I/O Submission Queue and Completion Queue may be used to carry out I/O (read/write) operations and may be distributed across CPU cores and threads.

An Admin command may impact one or more I/O queue pairs. The host should ensure that Admin actions are coordinated with threads that are responsible for the I/O queue pairs to avoid unnecessary error conditions. The details of this coordination are outside the scope of this specification.

## 3.4 Command Processing

This section describes the command issue and completion mechanism. It also describes how commands are built by host software and command completion processing.

Commands shall only be submitted by the host when the controller is ready as indicated in the Controller Status property (CSTS.RDY) and after appropriate I/O Submission Queue(s) and I/O Completion Queue(s) have been created.

### 3.4.1 Command Ordering Requirements

Commands which are not part of a fused operation (refer to section 3.4.2) and which comply with atomic operations requirements (refer to section 3.4.3), are processed as independent entities without reference to other commands submitted to the same I/O Submission Queue or to commands submitted to other I/O Submission Queues. For example, the controller is not responsible for checking the LBA of an NVM Command Set Read command or Write command to ensure any type of ordering between commands. If a Read command is submitted for LBA *x* and there is a Write command also submitted for LBA *x*, there is no guarantee of the order of completion for those commands (the Read command may finish first or the Write command may finish first). If there are ordering requirements between these commands, host software or the associated application is required to enforce that ordering above the level of the controller.

### 3.4.2 Fused Operations

Fused operations enable a more complex command by “fusing” together two simpler commands. This feature is optional; support for this feature is indicated in the FUSES field in the Identify Controller data structure in Figure 312.

Whether a command is part of a fused operation is specified by the Fused Operation (FUSE) field of Command Dword 0 shown in Figure 91. The FUSE field also specifies whether the command is the first command in the fused operation or the second command in the fused operation. If the FUSE field is set to a non-zero value and the controller does not support the requested fused operation, then the controller should abort the command with a status code of Invalid Field in Command.

In a fused operation, the requirements are:

- The commands shall be executed in sequence as an atomic unit. The controller shall behave as if no other operations have been executed between these two commands;
- The operation ends at the point an error is encountered in either command. If the first command in the sequence failed, then the second command in the sequence shall be aborted. If the second command in the sequence failed, then the completion status of the first command is sequence specific and is defined within the Fused Operation section of the applicable I/O Command Set specification;
- The commands shall be inserted next to each other in the same Submission Queue. If the controller processes a command violating this condition (e.g., a command with the FUSE field cleared to 00b (i.e., Normal operation) is inserted immediately after a command specifying the FUSE field set to 01b (i.e., Fused operation, first command), or is inserted immediately before a command specifying

the FUSE field set to 10b (i.e., Fused operation, second command)), then the controller shall abort the command specifying non-zero values of the FUSE field with a status code of Command Aborted due to Missing Fused Command. If the first command is in the last slot in the Submission Queue, then the second command shall be in the first slot in the Submission Queue as part of wrapping around. In the memory-based transport queue model, the Submission Queue Tail doorbell pointer update shall indicate both commands as part of one doorbell update. In the message-based transport queue model, the command capsules shall be submitted in-order;

- To abort the fused operation, the host submits an Abort command separately for each of the commands; and
- A completion queue entry is posted by the controller for each of the commands.

Refer to each I/O Command Set specification for applicability and additional details, if any.

### 3.4.3 Atomic Operations

The definition for atomic operations is command set specific. Refer to each I/O Command Set specification for applicability and additional details, if any.

### 3.4.4 Command Arbitration

After a command has been submitted to the controller (refer to section 1.5.19), the controller transfers submitted commands into the controller for subsequent processing using a vendor specific algorithm.

A command is being processed when the controller and/or namespace state is being accessed or modified by the command such as:

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

A command is completed when a completion queue entry for the command has been posted to the corresponding Completion Queue. Upon completion, all controller state and/or namespace state modifications made by that command are globally visible to all subsequently submitted commands.

A candidate command is a submitted command which has been transferred into the controller that the controller deems ready for processing. The controller selects command(s) for processing from the pool of submitted commands for each Submission Queue. The commands that comprise a fused operation shall be processed together and in order by the controller. The controller may select candidate commands for processing in any order. The order in which commands are selected for processing does not imply the order in which commands are completed.

Arbitration is the method used to determine the Submission Queue from which the controller starts processing the next candidate command(s). Once a Submission Queue is selected using arbitration, the Arbitration Burst setting determines the maximum number of commands that the controller may start processing from that Submission Queue before arbitration shall again take place. A fused operation may be considered as one or two commands by the controller.

All controllers shall support the round robin command arbitration mechanism. A controller may optionally implement weighted round robin with urgent priority class and/or a vendor specific arbitration mechanism. The Arbitration Mechanism Supported field in the Controller Capabilities property (CC.AMS) indicates optional arbitration mechanisms supported by the controller.

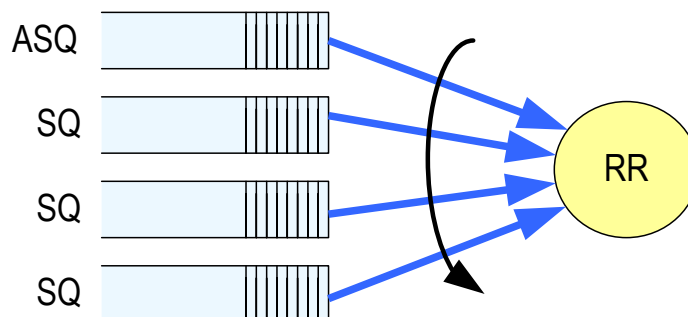
In order to make efficient use of the non-volatile memory, it is often advantageous to execute multiple commands from a Submission Queue in parallel. For Submission Queues that are using weighted round robin with urgent priority class or round robin arbitration, host software may configure an Arbitration Burst setting. The Arbitration Burst setting indicates the maximum number of commands that the controller may launch at one time from a particular Submission Queue. It is recommended that host software configure the Arbitration Burst setting as close to the recommended value by the controller as possible (specified in

the Recommended Arbitration Burst field of the Identify Controller data structure in Figure 312), taking into consideration any latency requirements. Refer to section 5.1.25.1.1.

### 3.4.4.1 Round Robin Arbitration

If the round robin arbitration mechanism is selected, the controller shall implement round robin command arbitration amongst all Submission Queues, including the Admin Submission Queue. In this case, all Submission Queues are treated with equal priority. The controller may select multiple candidate commands for processing from each Submission Queue per round based on the Arbitration Burst setting.

**Figure 80: Round Robin Arbitration**



### 3.4.4.2 Weighted Round Robin with Urgent Priority Class Arbitration

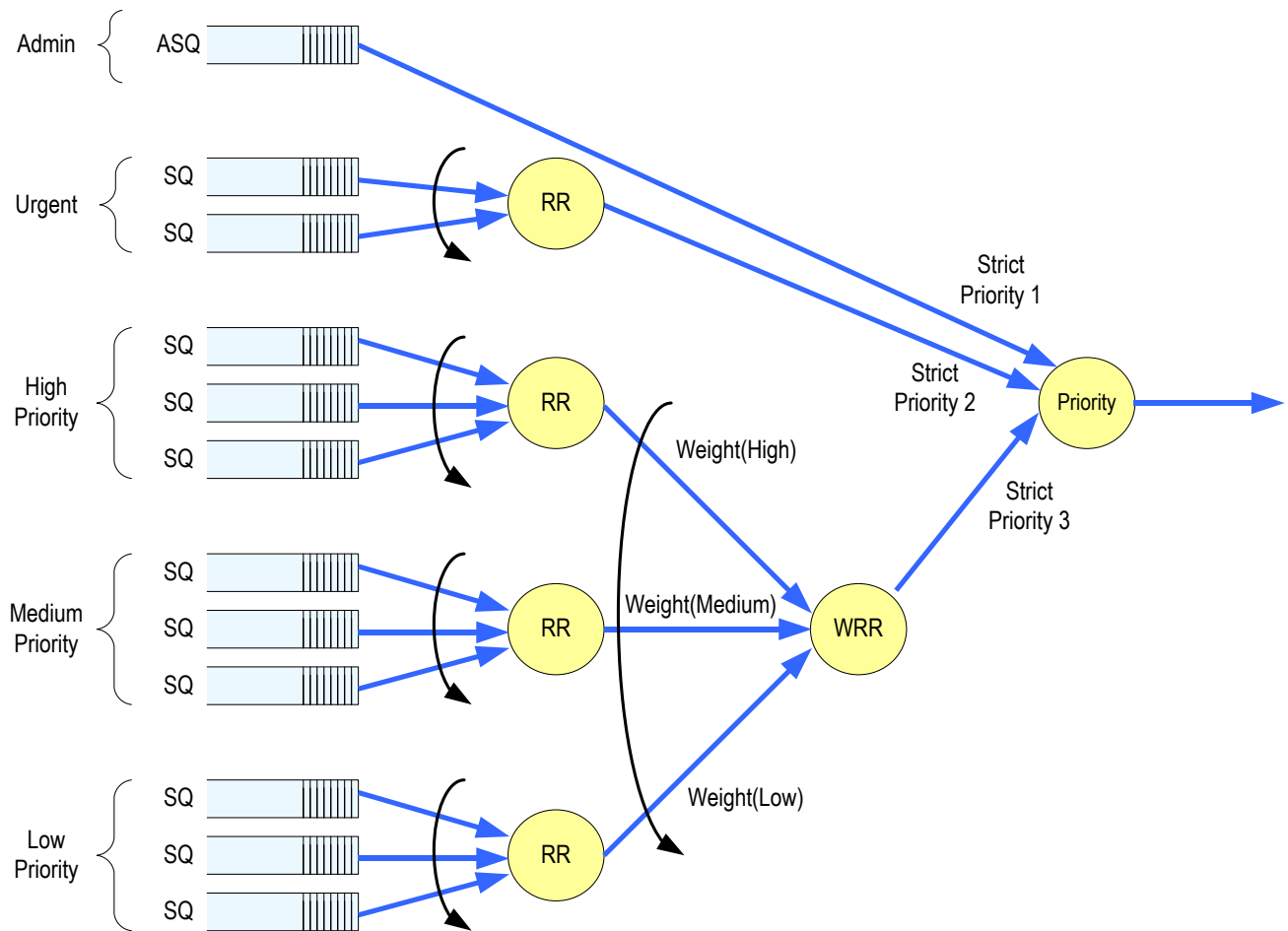
In this arbitration mechanism, there are three strict priority classes and three weighted round robin priority levels. If Submission Queue A is of higher strict priority than Submission Queue B, then all candidate commands in Submission Queue A shall start processing before candidate commands from Submission Queue B start processing.

The highest strict priority class is the Admin class that includes any command submitted to the Admin Submission Queue. This class has the highest strict priority above commands submitted to any other Submission Queue.

The next highest strict priority class is the Urgent class. Any I/O Submission Queue assigned to the Urgent priority class is serviced next after commands submitted to the Admin Submission Queue, and before any commands submitted to a weighted round robin priority level. Host software should use care in assigning any Submission Queue to the Urgent priority class since there is the potential to starve I/O Submission Queues in the weighted round robin priority levels as there is no fairness protocol between Urgent and non Urgent I/O Submission Queues.

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

**Figure 81: Weighted Round Robin with Urgent Priority Class Arbitration**



In Figure 81, the Priority decision point selects the highest priority candidate command selected next to start processing.

### 3.4.4.3 Vendor Specific Arbitration

A vendor may choose to implement a vendor specific arbitration mechanism. The mechanism(s) are outside the scope of this specification.

### 3.4.5 Outstanding Commands

A command is outstanding if:

- the host has submitted that command to the controller;
- the host has not received a completion for that command; and
- as described in this section:
  - the host has not performed an action that causes that command to no longer be outstanding; and
  - the host has not otherwise determined that that command is no longer outstanding.

A submitted command is no longer outstanding after the host:

- receives a completion for that command;

- receives a successful completion with Immediate Abort Not Performed bit cleared to '0' in Dword 0 of the completion queue entry for an Abort command specifying that outstanding command (refer to section 5.1.1);
- receives a successful completion with the Commands Aborted field set to 1h for a Cancel command with an Action Code of Single Command Cancel and specifying that outstanding command (refer to section 7.1);
- reads a CSTS.RDY bit value that indicates a controller is not able to process commands except for Fabrics commands (i.e., a value of '0'), if that outstanding command is not a Fabrics command (refer to section 3.7.2);
- reads a CSTS.SHST field value that indicates that the controller shutdown is complete (i.e., a value of 10b), if that outstanding command is not a Fabrics command (refer to section 3.6.1 for memory-based transports and section 3.6.2 for message-based transports);
- if using a memory-based transport, receives a successful completion for a Delete I/O Submission Queue command if that outstanding command was sent on the deleted I/O Submission queue (refer to section 3.3.1.3);
- if using a message-based transport:
  - receives a successful completion for a Disconnect command if that outstanding command was sent on the same I/O queue as the Disconnect command (refer to section 3.3.2.4); or
  - restores communication to the same controller after losing communication to that controller (refer to section 3.9.5).

If an outstanding command ceases to be outstanding for one of these reasons, then further controller processing of that command is no longer possible.

### 3.5 Controller Initialization

This section describes the recommended procedure for initializing a controller.

#### 3.5.1 Memory-based Controller Initialization (PCIe)

Upon completion of the transport-specific controller initialization steps defined within the relevant NVMe Transport binding specification, the host should perform the following sequence of actions to initialize the controller to begin executing commands:

1. The host waits for the controller to indicate that any previous reset is complete by waiting for CSTS.RDY to become '0';
2. The host configures the Admin Queue by setting the Admin Queue Attributes (AQA), Admin Submission Queue Base Address (ASQ), and Admin Completion Queue Base Address (ACQ) to appropriate values;
3. The host determines the supported I/O Command Sets by checking the state of CAP.CSS and appropriately initializing CC.CSS as follows:
  - a. If the CAP.CSS.NOIOCSS bit is set to '1', then the CC.CSS field should be set to 111b;
  - b. If the CAP.CSS.IOCSS bit is set to '1', then the CC.CSS field should be set to 110b; and
  - c. If the CAP.CSS.IOCSS bit is cleared to '0' and the CAP.CSS.NCSS bit is set to '1', then the CC.CSS field should be set to 000b;
4. The controller settings should be configured. Specifically:
  - a. The arbitration mechanism should be selected in CC.AMS; and
  - b. The memory page size should be initialized in CC.MPS;
5. The host enables the controller by setting CC.EN to '1';
6. The host waits for the controller to indicate that the controller is ready to process commands. The controller is ready to process commands when CSTS.RDY is set to '1';
7. The host determines the configuration of the controller by issuing the Identify command specifying the Identify Controller data structure (i.e., CNS 01h);



8. The host determines any I/O Command Set specific configuration information as follows:
  - a. If the CAP.CSS.IOCSS bit is set to '1', then the host does the following:
    - i. Issue the Identify command specifying the Identify I/O Command Set data structure (CNS 1Ch); and
    - ii. Issue the Set Features command with the I/O Command Set Profile Feature Identifier (FID 19h) specifying the index of the I/O Command Set Combination (refer to Figure 327) to be enabled;and
  - b. For each I/O Command Set that is enabled (Note: the NVM Command Set is enabled if the CC.CSS field is set to 000b):
    - i. Issue the Identify command specifying the I/O Command Set specific Active Namespace ID list (CNS 07h) with the appropriate Command Set Identifier (CSI) value of that I/O Command Set; and
    - ii. For each NSID that is returned:
      1. If the enabled I/O Command Set is the NVM Command Set or an I/O Command Set based on the NVM Command Set (e.g., the Zoned Namespace Command Set) issue the Identify command specifying the Identify Namespace data structure (CNS 00h); and
      2. Issue the Identify command specifying each of the following data structures (refer to Figure 310): the I/O Command Set specific Identify Namespace data structure, the I/O Command Set specific Identify Controller data structure, and the I/O Command Set independent Identify Namespace data structure;
9. If the controller implements I/O queues, then the host should determine the number of I/O Submission Queues and I/O Completion Queues supported using the Set Features command with the Number of Queues feature identifier. After determining the number of I/O Queues, the NVMe Transport specific interrupt registers (e.g., MSI and/or MSI-X registers) should be configured;
10. If the controller implements I/O queues, then the host should allocate the appropriate number of I/O Completion Queues based on the number required for the system configuration and the number supported by the controller. The I/O Completion Queues are allocated using the Create I/O Completion Queue command;
11. If the controller implements I/O queues, then the host should allocate the appropriate number of I/O Submission Queues based on the number required for the system configuration and the number supported by the controller. The I/O Submission Queues are allocated using the Create I/O Submission Queue command; and
12. To enable asynchronous notification of optional events, the host should issue a Set Features command specifying the events to enable. To enable asynchronous notification of events, the host should submit an appropriate number of Asynchronous Event Request commands. This step may be done at any point after the controller signals that the controller is ready (i.e., CSTS.RDY is set to '1').

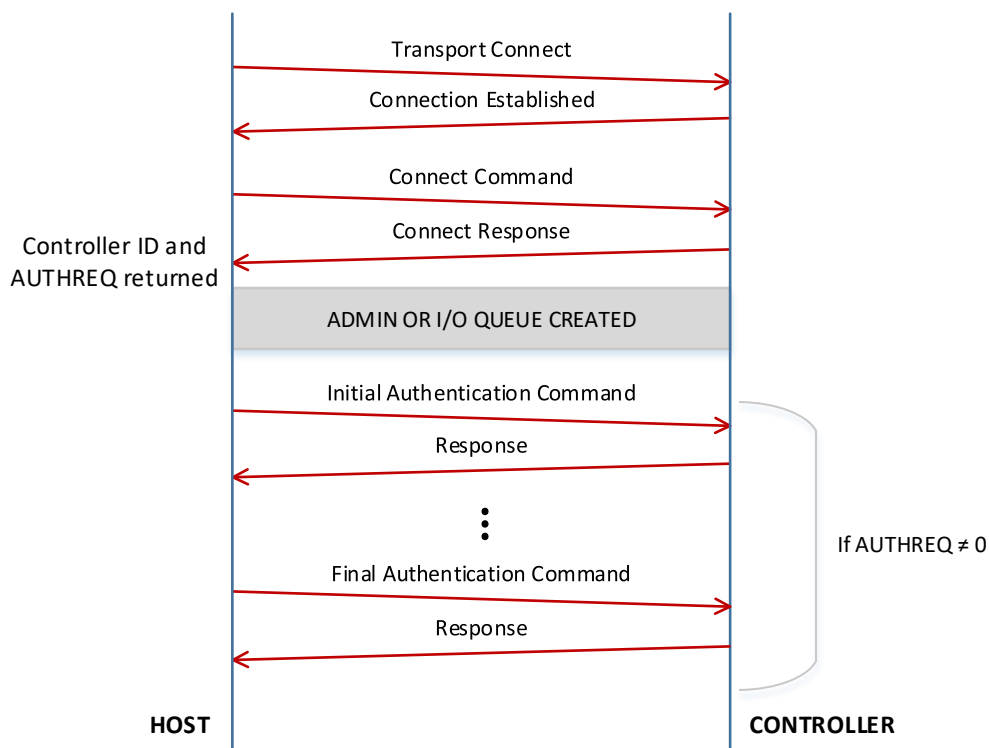
After performing these steps, the controller shall be ready to process Admin or I/O commands issued by the host.

For exit of the D3 power state (refer to the PCI Express Base Specification), the initialization steps outlined should be followed.

### **3.5.2 Message-based Controller Initialization (Fabrics)**

The host selects the NVM subsystem with which to create a host to controller association. The host first establishes an NVMe Transport connection with the NVM subsystem. Next the host forms an association with a controller and creates the Admin Queue using the Fabrics Connect command. Finally, the host configures the controller and creates I/O Queues. Figure 82 is a ladder diagram that describes the queue creation process for an Admin Queue or an I/O Queue.

Figure 82: Queue Creation Flow



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

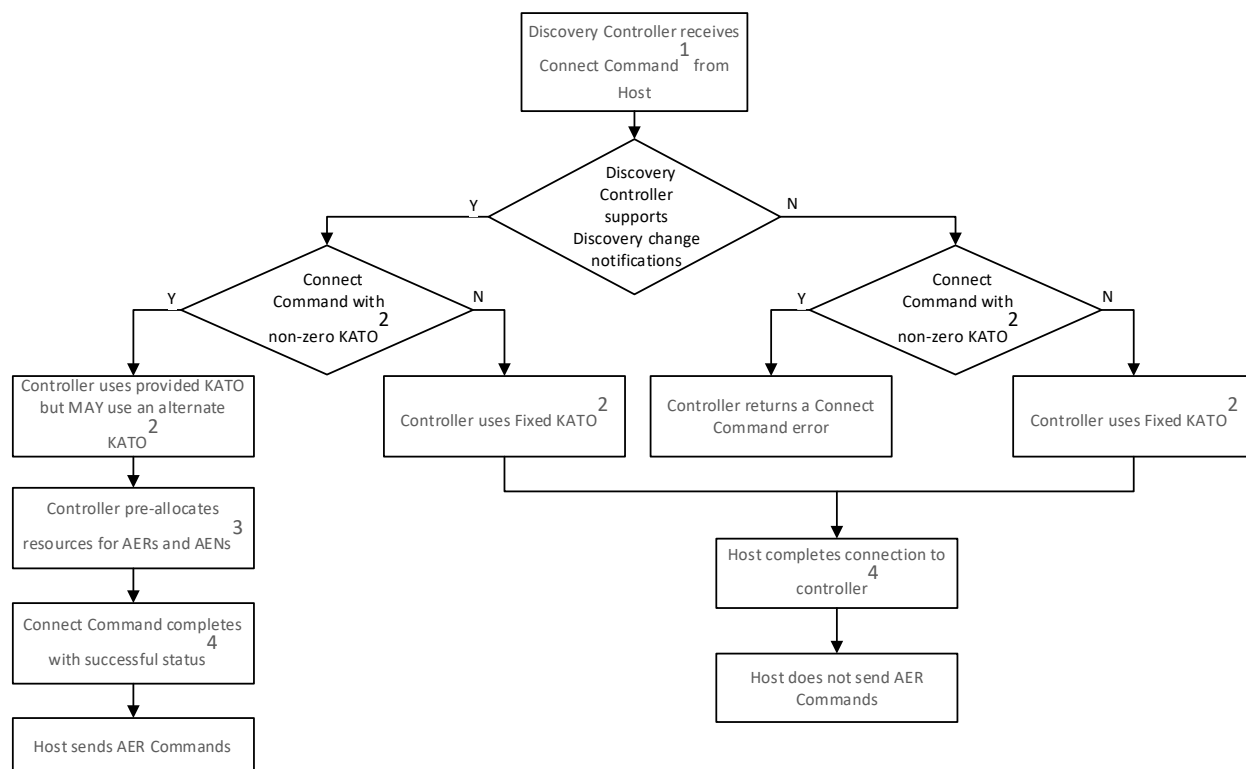
1. NVMe in-band authentication is performed if required (refer to section 8.3.4.2);
2. The host determines the controller capabilities;
3. The host determines the supported I/O Command Sets by checking the state of CAP.CSS and appropriately initializing CC.CSS as follows:
  - a. If the CAP.CSS.NOIOCSS bit is set to '1', then the CC.CSS field should be set to 111b;
  - b. If the CAP.CSS.IOCSS bit is set to '1', then the CC.CSS field should be set to 110b; and
  - c. If the CAP.CSS.IOCSS bit is cleared to '0' and the CAP.CSS.NCSS bit is set to '1', then the CC.CSS field should be set to 000b;
4. The host configures controller settings. Specific settings include:
  - a. The arbitration mechanism should be selected in CC.AMS; and
  - b. The memory page size should be initialized in CC.MPS;
5. The controller should be enabled by setting CC.EN to '1';
6. The host should wait for the controller to indicate the controller is ready to process commands. The controller is ready to process commands when CSTS.RDY is set to '1';
7. The host determines the configuration of the controller by issuing the Identify command specifying the Identify Controller data structure (i.e., CNS 01h);
8. The host determines any I/O Command Set specific configuration information as follows:
  - a. If the CAP.CSS.IOCSS bit is set to '1', then the host does the following:
    - i. Issue the Identify command specifying the Identify I/O Command Set data structure (CNS 1Ch); and

- ii. Issue the Set Features command with the I/O Command Set Profile Feature Identifier (FID 19h) specifying the index of the I/O Command Set Combination (refer to Figure 327) to be enabled;
  - and
  - b. For each I/O Command Set that is enabled (Note: the NVM Command Set is enabled if the CC.CSS field is set to 000b):
    - i. Issue the Identify command specifying the I/O Command Set specific Active Namespace ID list (CNS 07h) with the appropriate Command Set Identifier (CSI) value of that I/O Command Set; and
    - ii. For each NSID that is returned:
      - 1. If the enabled I/O Command Set is the NVM Command Set or an I/O Command Set based on the NVM Command Set (e.g., the Zoned Namespace Command Set) issue the Identify command specifying the Identify Namespace data structure (CNS 00h); and
      - 2. Issue the Identify command specifying each of the following data structures (refer to Figure 311): the I/O Command Set specific Identify Namespace data structure, the I/O Command Set specific Identify Controller data structure, and the I/O Command Set independent Identify Namespace data structure;
9. The host should determine:
- a. the maximum I/O Queue size using CAP.MQES; and
  - b. the number of I/O Submission Queues and I/O Completion Queues supported using the response from the Set Features command with the Number of Queues feature identifier;
10. The host should use the Connect command (refer to section 6.3) to create I/O Submission and Completion Queue pairs; and
11. To enable asynchronous notification of optional events, the host should issue a Set Features command specifying the events to enable. The host may submit one or more Asynchronous Event Request commands to be notified of asynchronous events as described by section 5.1.2. This step may be done at any point after the controller signals that the controller is ready (i.e., CSTS.RDY is set to '1').

The association may be removed if step 5 (i.e., setting CC.EN to '1') is not completed within 2 minutes after establishing the association.

### 3.5.2.1 Discovery Controller Initialization

The initialization process for Discovery controllers is described in Figure 83.

**Figure 83: Discovery Controller Initialization process flow****Notes:**

1. Refer to section 6.3.
2. Refer to the Keep Alive command in section 5.1.14.
3. Refer to the Asynchronous Event Request command in section 5.1.2.
4. Refer to the following steps in this section.

After the Connect command completes with a status of Successful Completion, the host performs the following steps:

1. NVMe authentication is performed if required (refer to section 8.3.4.2);
2. The host determines the controller's capabilities by reading the Controller Capabilities property;
3. The host configures the controller's settings by writing the Controller Configuration property, including setting CC.EN to '1' to enable command processing;
4. The host waits for the controller to indicate that the controller is ready to process commands. The controller is ready to process commands when CSTS.RDY is set to '1' in the Controller Status property; and
5. The host determines the features and capabilities of the controller by issuing an Identify command, specifying each applicable Controller data structure.

After initializing the Discovery controller, the host reads the Discovery log page. Refer to section 5.1.12.3.1.

### 3.5.3 Controller Ready Modes During Initialization

There are two controller ready modes:

- **Controller Ready With Media:** By the time the controller becomes ready (i.e., by the time that CSTS.RDY transitions from '0' to '1') after the controller is enabled (i.e., CC.EN transitions from '0' to '1'), then:
  - a) the controller shall be able to process all commands without error as described in section 3.5.4.1; and

- b) all namespaces attached to the controller and all media required to process Admin commands shall be ready (i.e., commands are not permitted to be aborted with a status code of Namespace Not Ready with the Do Not Retry bit cleared to '0' or Admin Command Media Not Ready with the Do Not Retry bit cleared to '0').
- **Controller Ready Independent of Media:** After the controller is enabled, all namespaces attached to the controller and media required to process Admin commands may or may not become ready by the time the controller becomes ready. Any Admin command or I/O command that specifies one or more namespaces attached to the controller is permitted to be aborted with a status code of Namespace Not Ready with the Do Not Retry bit cleared to '0' until CRTO.CRWMT amount of time after the controller is enabled.

Admin commands that require access to the media are permitted to be aborted with a status code of Admin Command Media Not Ready with the Do Not Retry bit cleared to '0' until CRTO.CRWMT amount of time after the controller is enabled. Refer to Figure 84 for a list of Admin commands that are permitted to be aborted with a status code of Admin Command Media Not Ready.

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

- a) all Admin commands not listed in Figure 84 by the time the controller is ready;
- b) all Admin commands listed in Figure 84 no later than CRTO.CRWMT amount of time after the controller is enabled; and
- c) all I/O commands no later than CRTO.CRWMT amount of time after the controller is enabled.

**Figure 84: Admin Commands Permitted to Return a Status Code of Admin Command Media Not Ready**

Admin Command	Additional Restrictions
Capacity Management	
Device Self-test	If the Device Self-Test would result in testing one or more namespaces, then returning a status code of Admin Command Media Not Ready is permitted. If the Device Self-Test would not result in testing any namespaces, then returning a status code of Admin Command Media Not Ready is not permitted.
Firmware Commit	
Firmware Image Download	
Get LBA Status	
Get Log Page	Get Log Page is only permitted to return a status code of Admin Command Media Not Ready for the following log pages: <ul style="list-style-type: none"> <li>• Device Self-test</li> <li>• Firmware Slot Information</li> <li>• Telemetry Controller-Initiated</li> <li>• Telemetry Host-Initiated</li> <li>• Predictable Latency Per NVM Set</li> <li>• Predictable Latency Event Aggregate</li> <li>• Persistent Event Log</li> <li>• LBA Status Information</li> <li>• Endurance Group Event Aggregate</li> <li>• Media Unit Status</li> <li>• Supported Capacity Configuration List</li> <li>• Boot Partition</li> <li>• Reservation Notification</li> <li>• Rotational Media Information</li> <li>• Vendor Specific</li> </ul>
Namespace Attachment	
Namespace Management	
Format NVM	
Sanitize	

**Figure 84: Admin Commands Permitted to Return a Status Code of Admin Command Media Not Ready**

Admin Command	Additional Restrictions
Security Receive <sup>1</sup>	
Security Send <sup>1</sup>	
Vendor Specific	
Notes:	
1. A host may require discovery operations performed via Security Send/Receive (e.g., TCG Level 0 Discovery) to be processed prior to media being ready. Therefore, it is recommended that controllers not return Admin Command Media Not Ready for such discovery operations.	

The Controller Ready Modes Supported (CAP.CRMS) field (refer to Figure 36) indicates which controller ready modes are supported. The CAP.CRMS field consists of two bits:

- the Controller Ready With Media Support (CAP.CRMS.CRWMS) bit; and
- the Controller Ready Independent of Media Support (CAP.CRMS.CRIMS) bit.

Controllers shall set the CAP.CRMS.CRWMS bit to '1' (i.e., set the CAP.CRMS field to 01b or 11b). The CAP.CRMS.CRWMS bit was not defined prior to NVM Express Base Specification, Revision 2.0. Controllers compliant with revisions earlier than NVM Express Base Specification, Revision 2.0 may clear the CAP.CRMS.CRWMS field to 00b.

The Controller Ready Independent of Media Enable (CC.CRIME) bit (refer to Figure 41) controls the controller ready mode based on the value of the CAP.CRMS field as follows:

- If the CAP.CRMS field is cleared to 00b, the controller ready mode is not able to be selected. In this case, the read-only CC.CRIME bit shall be cleared to '0' and should be ignored by host software;
- If the CAP.CRMS field is set to 01b (i.e., the CAP.CRMS.CRIMS bit is cleared to '0' and the CAP.CRMS.CRWMS bit is set to '1'), then the controller is in Controller Ready With Media mode and the read-only CC.CRIME bit shall be cleared to '0'; and
- If the CAP.CRMS field is set to 11b, then both controller ready modes are supported, and the host may select the controller ready mode by modifying the value of the CC.CRIME bit. In this situation, the host should set the controller ready mode by writing to the CC.CRIME bit before the controller is enabled (e.g., as part of the initialization sequence of actions described in section 3.5).

### 3.5.4 Controller Ready Timeouts During Initialization

The CAP.CRMS field was not defined prior to NVM Express Base Specification, Revision 2.0. Controllers compliant with revisions earlier than NVM Express Base Specification, Revision 2.0 may clear the CAP.CRMS field to 00b. This section is applicable to controllers that clear the CAP.CRMS field to 00b and controllers that set CAP.CRMS to a non-zero value.

There are three controller ready timeout fields:

1. CAP.TO (refer to Figure 36);
2. CRTO.CRWMT (refer to Figure 57); and
3. CRTO.CRIMT (refer to Figure 57).

The details regarding these timeouts during controller initialization are as follows:

- The CAP.TO field shall be set as described in Figure 36;
- If the CAP.CRMS field is cleared to 00b, then the worst-case time the host should wait after the controller is enabled (i.e., CC.EN transitions from '0' to '1') for the controller to become ready (CSTS.RDY transitions from '0' to '1') is indicated by CAP.TO;
- If the controller is in Controller Ready With Media mode (i.e., the CC.CRIME bit is cleared to '0'), then:

- i. the Controller Ready Independent of Media Timeout (CRTO.CRIMT) field is not applicable; and
  - ii. the Controller Ready With Media Timeout (CRTO.CRWMT) indicates the worst-case time the host should wait after the controller is enabled for:
    - 1. the controller to become ready and be able to process all commands without error as described in section 3.5.4.1; and
    - 2. all attached namespaces and media required to process Admin commands to become ready,
- and
- d) If the controller is in Controller Ready Independent of Media mode (i.e., the CC.CRIME bit is set to '1'), then
    - i. the Controller Ready With Media Timeout (CRTO.CRWMT) field indicates the worst-case time that host software should wait for all attached namespaces and media required to process Admin commands to become ready after the controller is enabled; and
    - ii. the Controller Ready Independent of Media Timeout (CRTO.CRIMT) field indicates the worst-case time the host should wait after the controller is enabled for the controller to become ready and be able to process:
      - 1. all commands that do not access attached namespaces; and
      - 2. Admin commands that do not require access to media,without error as described in section 3.5.4.1.

Changes to the value of the CC.CRIME bit shall have no effect on the values of the CRTO.CRWMT and CRTO.CRIMT fields. Changes to the value of the CC.CRIME bit may have an effect on the value of the CAP.TO field (refer to Figure 36).

#### 3.5.4.1 Handling Errors During Initialization

If the CAP.CRMS field is non-zero and the controller has been enabled by transitioning CC.EN from '0' to '1' and the controller encounters a failure that prevents:

- a) at least one:
  - command that does not access attached namespaces; or
  - Admin command that does not require access to media (refer to Figure 84),from being able to be processed without error by the amount of time indicated by the:
  - Controller Ready Independent of Media Timeout (CRTO.CRIMT) field since the controller was enabled if the controller is in Controller Ready Independent of Media mode (i.e., the CC.CRIME bit is set to '1'); or
  - Controller Ready With Media Timeout (CRTO.CRWMT) field since the controller was enabled if the controller is in Controller Ready With Media mode (i.e., the CC.CRIME bit is cleared to '0');
- b) at least one namespace attached to the controller from becoming ready by the amount of time indicated by the Controller Ready With Media Timeout (CRTO.CRWMT) field since the controller was enabled; or
- c) media required by at least one Admin command from becoming ready by the amount of time indicated by the Controller Ready With Media Timeout (CRTO.CRWMT) field since the controller was enabled,

then:

- a) if the controller has not become ready, then the controller shall become ready (i.e., set CSTS.RDY to '1') no later than CRTO.CRWMT amount of time after the controller was enabled; and

- b) if the Persistent Event log page is supported, then the controller shall record an NVM Subsystem Hardware Error Event with the NVM Subsystem Hardware Error Event code set to a value of Controller Ready Timeout Exceeded in the Persistent Event log page (refer to Figure 236).

### 3.6 Shutdown Processing

This section describes the recommended procedure for shutdown processing prior to a power-off condition.

There are two shutdown processing mechanisms, controller shutdown (refer to sections 3.6.1 and 3.6.2) and NVM Subsystem Shutdown (refer to section 3.6.3). The CSTS.ST bit indicates the shutdown mechanism that is in progress, if any (refer to Figure 42). A host requests a controller shutdown by modifying the CC.SHN field (refer to Figure 41). A host requests an NVM Subsystem Shutdown by modifying the NSSD property (refer to section 3.1.4.20) or by issuing an NVMe-MI Shutdown command to a Management Endpoint (refer to the NVM Express Management Interface Specification).

At most one shutdown processing mechanism is able to be in progress for a controller at any time. If an NVM Subsystem Shutdown is requested while a controller shutdown is in progress, then the NVM Subsystem Shutdown overrides the controller shutdown. The progress and completion of shutdown processing is indicated by the CSTS.SHST field (refer to Figure 42).

NVM Subsystem Shutdown should not be supported by any NVM subsystem that does not support more than one controller, without counting virtual controllers (e.g., NVM Subsystem Shutdown should not be supported by an NVM subsystem that supports one primary controller and multiple secondary controllers) (refer to section 8.2.6).

Figure 85 describes the interactions of the shutdown processing state indicated by the CSTS.SHST field with the state of the controller indicated by the CC.EN bit (refer to Figure 41) and by the CSTS.RDY bit (refer to Figure 42). The four possible media states in Figure 85 are: shutdown, shutdown in progress, usable, and initialization in progress.

**Figure 85: Shutdown Processing Interactions**

CC.EN	CSTS.RDY	CSTS.SHST	Controller able to process Admin and I/O commands <sup>4</sup>	Media state	Controller ready to be powered off
0	0	00b	no	any	implementation specific <sup>1</sup>
0	0	01b	no <sup>3</sup>	shutdown in progress	no
0	0	10b	no	shutdown	yes
1	1	00b	yes	initialization in progress or usable <sup>2</sup>	no
1	1	01b	no <sup>3</sup>	shutdown in progress	no
1	1	10b	no	shutdown	yes

Notes:

1. In some cases (e.g., following initial application of power, or following a Controller Level Reset that occurred while shutdown processing was reported as complete), the controller is permitted to initialize the media and cease being ready to be powered off as a consequence.
2. If the CC.CRIME bit is cleared to '0', then the media is usable. If the CC.CRIME bit is set to '1', then either media initialization is in progress or the media is usable (refer to Figure 41).
3. While shutdown processing is in progress, the controller may abort any command with a status code of Commands Aborted due to Power Loss Notification.
4. I/O commands are only able to be processed by a controller that supports I/O commands. Fabrics commands are always able to be processed by a controller that supports Fabrics commands.



Figure 85 does not include transition conditions for a controller that is becoming ready or is undergoing reset. During these transitions, the CC.EN bit and the CSTS.RDY bit have different values (refer to Figure 41 and Figure 42). The media may or may not be usable during these transitions.

Figure 85 does not include the NVMe-MI effects of processing an Admin command that requires access to the media (refer to Figure 84) and specifies the Ignore Shutdown bit set to '1' is processed by the controller via the out-of-band mechanism (refer to the NVMe Express Management Interface Specification). Processing of such a command causes the media to become usable, after which the media may or may not be returned to its previous condition.

### 3.6.1 Memory-based Controller Shutdown (PCIe)

It is recommended that the host perform an orderly shutdown of the controller by following the procedure in this section when a power-off or shutdown condition is imminent.

The host should perform the following actions in sequence for a normal controller shutdown:

1. If the controller is enabled (i.e., CC.EN (refer to Figure 41) is set to '1'):
  - a. Stop submitting any new I/O commands to the controller and allow any outstanding commands to complete;
  - b. If the controller implements I/O queues, then the host should delete all I/O Submission Queues, using the Delete I/O Submission Queue command (refer to section 5.2.4). A result of the successful completion of the Delete I/O Submission Queue command is that any remaining commands outstanding are aborted;
  - c. If the controller implements I/O queues, then the host should delete all I/O Completion Queues, using the Delete I/O Completion Queue command (refer to section 5.2.3);

and

2. The host should set the Shutdown Notification (CC.SHN) field (refer to Figure 41) to 01b to indicate a normal controller shutdown operation. The controller indicates when shutdown processing is completed by updating the Shutdown Status (CSTS.SHST) field to 10b and the Shutdown Type (CSTS.ST) field (refer to Figure 42) is cleared to '0'.

The host should perform the following actions in sequence for an abrupt shutdown:

1. If the controller is enabled (i.e., CC.EN is set to '1'), then stop submitting any new I/O commands to the controller; and
2. The host should set the Shutdown Notification (CC.SHN) field (refer to Figure 41) to 10b to indicate an abrupt shutdown operation. The controller indicates when shutdown processing is completed by updating the Shutdown Status (CSTS.SHST) field (refer to Figure 42) to 10b and CSTS.ST (refer to Figure 42) is cleared to '0'.

For entry to the D3 power state (refer to the PCI Express Base Specification), the shutdown steps outlined for a normal controller shutdown should be followed.

It is recommended that the host wait a minimum of the RTD3 Entry Latency reported in the Identify Controller data structure (refer to Figure 312) for the shutdown operations to complete; if the value reported in RTD3 Entry Latency is 0h, then the host should wait for a minimum of one second. While shutdown processing is in progress on a controller, it is not recommended to disable that controller via the CC.EN bit. This causes a Controller Reset which may impact the time required to complete shutdown processing. While shutdown processing is in progress on a controller, that controller may abort any command with a status code of Commands Aborted due to Power Loss Notification.

The controller is ready to be powered off (e.g., the media is in the shutdown state (refer to Figure 85)) when the CSTS.ST bit is cleared to '0', and the CSTS.SHST field indicates that controller shutdown processing is complete (i.e., the CSTS.SHST field is set to 10b), regardless of the value of the CC.EN bit. The controller remains ready to be powered off (e.g., the media remains in the shutdown state) until:

- A. the controller is enabled (i.e., the CC.EN bit transitions from '0' to '1');
- B. the controller is reset by a Controller Level Reset; or

- C. an Admin command that requires access to the media (refer to Figure 84) and specifies the Ignore Shutdown bit set to '1' is processed by the controller via the out-of-band mechanism (refer to the NVM Express Management Interface Specification).

If a Controller Level Reset occurs while controller shutdown processing is reported as complete (i.e., the CSTS.ST bit is cleared to '0' and the CSTS.SHST field is set to 10b), then the controller may remain ready to be powered off (e.g., the media remains in the shutdown state) or the controller may cease being ready to be powered off (e.g., because the controller is preparing the media for use (refer to section 3.7.2)).

If the power scope for the controller includes multiple controllers (e.g., the CAP.CPS field is set to 10b or is set to 11b), and any controller included in that power scope is not ready to be powered off, then the portion of the NVM subsystem included in that power scope is not ready to be powered off.

To start executing commands on the controller after that controller reports controller shutdown processing complete (i.e., the CSTS.ST bit is cleared to '0' and the CSTS.SHST field is set to 10b) utilizing the CC.EN bit:

- if the CC.EN bit is set to '1', then a Controller Level Reset is required to clear the CC.EN bit to '0' on that controller and the CC.EN bit is subsequently required to be set to '1' as part of the initialization sequence (refer to section 3.5); and
- if the CC.EN bit is cleared to '0', then:
  - a Controller Level Reset is required and the CC.EN bit is subsequently required to be set to '1' as part of the initialization sequence (refer to section 3.5); or
  - the CC.EN bit is required to be set to '1' and the CC.SHN field is required to be cleared to 00b with the same write to the CC property (refer to Figure 41). The controller clears the CSTS.SHST field to 00b in response to that write.

The initialization sequence (refer to section 3.5) should then be executed on that controller.

It is an implementation choice whether the host aborts all outstanding commands to the Admin Queue prior to the controller shutdown. The only commands that should be outstanding to the Admin Queue when the controller reports shutdown processing complete are Asynchronous Event Request commands.

If the host is no longer able to communicate with the controller before that host receives either:

- completions for all outstanding commands submitted to that controller (refer to section 3.4.5); or
- a CSTS.SHST field value that indicates that the controller shutdown is complete,

then it is strongly recommended that the host take the steps described in section 9.6 to avoid possible data corruption caused by interaction between outstanding commands and subsequent commands submitted by that host to another controller.

### 3.6.2 Message-based Controller Shutdown (Fabrics)

To initiate a shutdown of a controller, the host should use the Property Set command (refer to section 6.6) to set the Shutdown Notification (CC.SHN) field to:

- 01b to initiate a normal shutdown operation; or
- 10b to initiate an abrupt shutdown.

After the host initiates a controller shutdown, the host may either disconnect at the NVMe Transport level or the host may choose to poll CSTS.SHST to determine when the controller shutdown is complete (i.e., the controller should not initiate a disconnect at the NVMe Transport level). It is an implementation choice whether the host aborts all outstanding commands prior to initiating the shutdown.

If the host is no longer able to communicate with the controller before that host receives either:

- completions for all outstanding commands submitted to that controller (refer to section 3.4.5); or
- a CSTS.SHST field value that indicates that the controller shutdown is complete,

then it is strongly recommended that the host take the steps described in section 9.6 to avoid possible data corruption caused by interaction between outstanding commands and subsequent commands submitted by that host to another controller.

The CC.EN field is not used to shutdown the controller (i.e., it is used for Controller Reset).

From the time a controller shutdown is initiated until:

- a Controller Level Reset occurs; or
- the controller, if dynamic, is removed from the NVM subsystem,

the controller shall:

- process only Fabrics commands (refer to Figure 540); and
- disable the Keep Alive timer, if supported.

After the CC.EN bit transitions to '0' (i.e., due to Controller Level Reset), the association between the host and controller shall be preserved for at least 2 minutes. After this time, the association may be removed if the controller has not been re-enabled.

To start executing commands on the controller after that controller reports controller shutdown processing complete (i.e., the CSTS.ST bit is cleared to '0' and the CSTS.SHST field is set to 10b) utilizing the CC.EN bit:

- if the CC.EN bit is set to '1', then a Controller Level Reset is required to clear the CC.EN bit to '0' on that controller and the CC.EN bit is subsequently required to be set to '1' as part of the initialization sequence (refer to section 3.5); and
- if the CC.EN bit is cleared to '0', then:
  - a Controller Level Reset is required and the CC.EN bit is subsequently required to be set to '1' as part of the initialization sequence (refer to section 3.5); or
  - the CC.EN bit is required to be set to '1' and the CC.SHN field is required to be cleared to 00b with a single Property Set command (refer to section 6.6) that changes the CC property (refer to Figure 41). The controller clears the CSTS.SHST field to 00b in response to that write.

The initialization sequence (refer to section 3.5) should then be executed on that controller.

### 3.6.3 NVM Subsystem Shutdown

An NVM Subsystem Shutdown initiates a shutdown of all controllers in a domain or NVM subsystem from a single controller.

Interactions between NVM Subsystem Shutdown and Power Loss Signaling processing are described in section 8.2.5.

A controller indicates support for the NVM Subsystem Shutdown Feature by setting the CAP.NSSS bit to '1' (refer to Figure 36).

The NVM Subsystem Shutdown Feature defined in this revision of the NVM Express Base Specification includes some functionality that differs from the functionality of the NVM Subsystem Shutdown Feature defined in revision 2.0 of the NVM Express Base Specification. A controller indicates support for these functionality differences by setting the NVM Subsystem Shutdown Enhancements Supported (CAP.NSSES) bit to '1' (refer to Figure 36).

If a controller sets the CAP.NSSES bit to '1', then while an NVM Subsystem Shutdown is reported as in progress or is reported as complete (i.e., while the CSTS.ST bit is set to '1' and the CSTS.SHST field is set to 01b or is set to 10b):

- a. a Controller Reset initiates a Controller Level Reset (CLR) (refer to section 3.7.2); and
- b. the values of both the CSTS.ST bit and the CSTS.SHST field (refer to Figure 40) are not changed by a CLR initiated by any method other than an NVM Subsystem Reset.

If a controller clears the CAP.NSSES bit to '0', then, as defined in revision 2.0 of the NVM Express Base Specification:

- a. while an NVM Subsystem Shutdown is reported as in progress or is reported as complete, a Controller Reset does not initiate a CLR (i.e., Controller Reset is disabled); and
- b. while an NVM Subsystem Shutdown is reported as complete, any CLR initiated by any transport-specific reset type may clear the value of the CSTS.ST bit to '0' and may clear the value of the CSTS.SHST field to 00b.

A host is able to support NVM Subsystem Shutdown functionality both on controllers that set the CAP.NSSES bit to '1' and on controllers that clear the CAP.NSSES bit to '0' by ensuring that any NVM Subsystem Shutdown is followed by an NVM Subsystem Reset regardless of the value of the CSTS.ST bit and the value of the CSTS.SHST field.

### 3.6.3.1 NVM Subsystem Shutdown in a Single Domain NVM Subsystem

A normal shutdown on all controllers within the NVM subsystem (i.e., normal NVM Subsystem Shutdown) is initiated by:

- a host writing the value 4E726D6Ch ("NrmI") to NSSD.NSSC when CAP.CPS is set to 11b; or
- issuing an NVMe-MI Shutdown command to a Management Endpoint (refer to the NVM Express Management Interface Specification) specifying a normal shutdown.

For each controller in the NVM subsystem for this normal NVM Subsystem Shutdown, if:

- CSTS.SHST is set to 00b; and
- An outstanding Asynchronous Event Request command exists,

then the controller shall issue a Normal NVM Subsystem Shutdown event prior to shutting down the controller.

An abrupt shutdown on all controllers within the NVM subsystem (i.e., abrupt NVM Subsystem Shutdown) is initiated by:

- a host writing the value 41627074h ("Abpt") to NSSD.NSSC when CAP.CPS is set to 11b; or
- issuing an NVMe-MI Shutdown command to a Management Endpoint (refer to the NVM Express Management Interface Specification) specifying an abrupt shutdown.

While NVM Subsystem Shutdown processing is in progress, any controller in the NVM subsystem may abort any command with a status code of Commands Aborted due to Power Loss Notification.

It is recommended that the host wait a minimum of the NVM Subsystem Shutdown Latency reported in the Identify Controller data structure (refer to Figure 312) for NVM Subsystem Shutdown processing to complete; if the reported NVM Subsystem Shutdown Latency value is 0h, then the host should wait for a minimum of 30 seconds. While an NVM Subsystem Shutdown is reported as in progress, it is not recommended to reset the NVM subsystem via an NVM Subsystem Reset or a power cycle (which causes an NVM Subsystem Reset). This aborts the NVM Subsystem Shutdown which may impact the subsequent time required for the NVM subsystem to become ready to perform I/O (e.g., after power is reapplied following a power cycle).

For either a normal or an abrupt NVM Subsystem Shutdown, the NVM subsystem is ready to be powered off (e.g., the media is in the shutdown state (refer to Figure 85)) when the CSTS.ST bit is set to '1' and the CSTS.SHST field indicates that shutdown processing is complete (i.e., the CSTS.SHST field is set to 10b) on any controller in the NVM subsystem. The NVM subsystem shall not set the CSTS.SHST field to 10b on any controller in the NVM subsystem until the entire NVM subsystem is ready to be powered off. The NVM Subsystem shall indicate that NVM Subsystem Shutdown processing is complete by setting the CSTS.SHST field to 10b on all controllers in the NVM subsystem. The NVM subsystem remains ready to be powered off (e.g., the media remains in the shutdown state) until:

- A. an NVM Subsystem Reset; or
- B. an Admin command that requires access to the media (refer to Figure 84) and specifies the Ignore Shutdown bit set to '1' is processed by any controller via the out-of-band mechanism (refer to the NVM Express Management Interface Specification).

If a normal or an abrupt NVM Subsystem Shutdown is reported as in progress or is reported as complete within the NVM subsystem (i.e., the CSTS.ST bit is set to '1' and the CSTS.SHST field is set to either 01b or 10b on all controllers in the NVM subsystem, indicating that an NVM Subsystem Reset has not occurred since initiation of that NVM Subsystem Shutdown), then:

- an NVM Subsystem Reset:
  - shall abort any in progress NVM Subsystem Shutdown;
  - clears the CSTS.SHST field to 00b in all controllers in the NVM subsystem; and
  - clears the CSTS.ST bit to '0' in all controllers in the NVM subsystem;

and

- a Controller Level Reset of any controller in the NVM subsystem that is initiated by any other method (refer to section 3.7.2):
  - shall not abort any in progress NVM Subsystem Shutdown;
  - does not change the values of the CSTS.ST bit and the CSTS.SHST field, as described in Figure 40; and
  - shall not cause that NVM subsystem to cease being ready to be powered off (e.g., shall not transition the media out of the shutdown state) if that NVM Subsystem was ready to be powered off when that Controller Level Reset was initiated.

To start executing commands on the controller after that controller reports NVM Subsystem Shutdown processing complete (i.e., the CSTS.ST bit is set to '1' and the CSTS.SHST field is set to 10b):

- regardless of the value of the CC.EN bit, an NVM Subsystem Reset is required; and
- the CC.EN bit is subsequently required to be set to '1' as part of the initialization sequence (refer to section 3.5).

The initialization sequence (refer to section 3.5) should then be executed on that controller.

### 3.6.3.2 Domain Shutdown in a Multiple Domain NVM Subsystem

A normal NVM Subsystem Shutdown on this controller and all controllers within the associated domain is initiated by:

- a host writing the value 4E726D6Ch ("NrmI") to NSSD.NSSC when CAP.CPS is set to 10b; or
- issuing an NVMe-MI Shutdown command to a Management Endpoint (refer to the NVM Express Management Interface Specification) specifying a normal shutdown.

For each controller in the domain for this normal NVM subsystem shutdown, if:

- CSTS.SHST is cleared to 00b; and
- An outstanding Asynchronous Event Request command exists,

then the controller shall issue a Normal NVM Subsystem Shutdown event prior to shutting down the controller.

An abrupt NVM Subsystem Shutdown to this controller and all controllers within the associated domain is initiated by:

- a host writing the value 41627074h ("Abpt") to NSSD.NSSC when CAP.CPS is set to 10b; or
- issuing an NVMe-MI Shutdown command to a Management Endpoint (refer to the NVM Express Management Interface Specification) specifying an abrupt shutdown.

While NVM Subsystem Shutdown processing is in progress, any controller in the domain may abort any command with a status code of Commands Aborted due to Power Loss Notification.

It is recommended that the host wait a minimum of the NVM Subsystem Shutdown Latency reported in the Identify Controller data structure (refer to Figure 312) for NVM Subsystem Shutdown processing on a domain to complete; if the reported NVM Subsystem Shutdown Latency value is 0h, then the host should wait for a minimum of 30 seconds. While an NVM Subsystem Shutdown is reported as in progress, it is not recommended to reset the domain via either an NVM Subsystem Reset on the domain or power cycling the domain (which causes an NVM Subsystem Reset on the domain). This aborts the NVM Subsystem Shutdown which may impact the subsequent time required for the domain to become ready to perform I/O (e.g., after power is reapplied following a power cycle).

For either a normal or an abrupt NVM Subsystem Shutdown on the domain, the domain is ready to be powered off (e.g., the media is in the shutdown state (refer to Figure 85)) when the CSTS.ST bit is set to '1' and the CSTS.SHST field indicates that shutdown processing is complete (i.e., the CSTS.SHST field is set to 10b) on any controller in the domain. The NVM subsystem shall not set the CSTS.SHST field to 10b on any controller in the domain until the entire domain is ready to be powered off. The NVM Subsystem shall indicate that NVM Subsystem Shutdown processing is complete by setting the CSTS.SHST field to 10b on all controllers in the domain. The domain remains ready to be powered off (e.g., the media remains in the shutdown state) until:

- A. an NVM Subsystem Reset occurs on that domain; or
- B. an Admin command that requires access to the media (refer to Figure 84) and specifies the Ignore Shutdown bit set to '1' is processed by any controller in the domain via the out-of-band mechanism (refer to the NVM Express Management Interface Specification).

If a normal or an abrupt NVM Subsystem Shutdown is reported as in progress or is reported as complete within a domain (i.e., the CSTS.ST bit is set to '1' and the CSTS.SHST field is set to either 01b or 10b on all controllers in the domain, indicating that an NVM Subsystem Reset on the domain has not occurred since initiation of that NVM Subsystem Shutdown), then:

- an NVM Subsystem Reset on the domain:
  - shall abort any in progress NVM Subsystem Shutdown on the domain; and
  - clears the CSTS.SHST field to 00b in all controllers in the domain; and
  - clears the CSTS.ST bit to '0' in all controllers in the domain;

and

- a Controller Level Reset of any controller in the domain that is initiated by any other method (refer to section 3.7.2)
  - shall not abort any in progress NVM Subsystem Shutdown on the domain;
  - does not change the values of the CSTS.ST bit and the CSTS.SHST field, as described in Figure 42; and
  - shall not cause the domain to cease being ready to be powered off (e.g., shall not transition the media out of the shutdown state) if the domain was ready to be powered off when that Controller Level Reset was initiated.

To start executing commands on the controller after that controller reports NVM Subsystem Shutdown processing complete (i.e., the CSTS.ST bit is set to '1' and the CSTS.SHST field is set to 10b):

- regardless of the value of CC.EN, an NVM Subsystem Reset on that domain is required; and
- the CC.EN bit is subsequently required to be set to '1' as part of the initialization sequence (refer to section 3.5).

The initialization sequence (refer to section 3.5) should then be executed on that controller.

### **3.7 Resets**

#### **3.7.1 NVM Subsystem Reset**

Interactions between NVM Subsystem Reset and Power Loss Signaling processing are described in section 8.2.5.

##### **3.7.1.1 Single Domain NVM Subsystems**

The scope of an NVM Subsystem Reset depends on whether the NVM subsystem supports multiple domains. In an NVM subsystem that does not support multiple domains, the scope of the NVM Subsystem Reset is the entire NVM subsystem.

An NVM Subsystem Reset is initiated when:

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

When an NVM Subsystem Reset occurs, the entire NVM subsystem is reset. This includes the initiation of a Controller Level Reset on all controllers that make up the NVM subsystem, disabling of the Persistent Memory Region associated with all controllers that make up the NVM subsystem, and any transport specific actions defined in the applicable NVMe transport specification.

The occurrence of an NVM Subsystem Reset while power is applied to the NVM subsystem is reported by the initial value of the CSTS.NSSRO field following the NVM Subsystem Reset. This field may be used by host software to determine if the sudden loss of communication with a controller was due to an NVM Subsystem Reset or some other condition.

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

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

##### **3.7.1.2 Multiple Domain NVM Subsystems**

The scope of an NVM Subsystem Reset depends on whether the NVM subsystem supports multiple domains. In an NVM subsystem that supports multiple domains, the scope of the NVM Subsystem Reset is either the controllers that are in a domain or the entire NVM subsystem.

An NVM Subsystem Reset on a domain is initiated when:

- Power is applied to that domain;
- A value of 4E564D65h (i.e., “NVMe”) is written to the NSSR.NSSRC field of one of the controllers in that domain; or
- A vendor specific event occurs within that domain.

When an NVM Subsystem Reset occurs the entire domain is reset. This includes the initiation of a Controller Level Reset on all controllers that are in the domain, disabling of the Persistent Memory Region associated with all controllers that are in the domain, and any transport specific actions defined in the applicable NVMe transport specification.

Alternatively, an NVM Subsystem Reset in an NVM subsystem that supports multiple domains may reset the entire NVM subsystem.

The occurrence of an NVM Subsystem Reset while power is applied to the domain is reported by the initial value of the CSTS.NSSRO field following the NVM Subsystem Reset. This field may be used by host software to determine if the sudden loss of communication with a controller was due to an NVM Subsystem Reset or some other condition.

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

### 3.7.2 Controller Level Reset

The following methods initiate a Controller Level Reset:

- NVM Subsystem Reset;
- Controller Reset (i.e., host clears the CC.EN bit 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, the CSTS.RDY bit is cleared to '0'; and
- All NVMe controller properties defined in either section 3.1.4 or the applicable NVMe Transport binding specification and all internal controller state are reset, with the following exceptions:
  - for memory-based controllers:
    - the following are not reset as part of a Controller Level Reset caused by a Controller Reset:
      - Admin Queue properties (i.e., AQA, ASQ, and ACQ);
      - Persistent Memory Region properties (i.e., PMRCAP, PMRCTL, PMRSTS, PMREBS, PMRSWTP, PMRMSCU, and PMRMSCU); and
      - The Controller Memory Buffer Memory Space Control property (CMBMSC) (refer to the NVM Express NVMe over PCIe Transport Specification);
    - and
    - the following are not reset as part of a Controller Level Reset caused by a Function Level Reset:
      - the Controller Memory Buffer Memory Space Control property (CMBMSC);
    - and
  - for message-based controllers:
    - there are no exceptions.

In all Controller Level Reset cases except a Controller Reset, the controller properties defined by the transport (e.g., the PCIe registers defined by the PCIe Base Specification) are reset as defined by the applicable NVMe Transport binding specification (e.g., the NVM Express NVMe over PCIe Transport Specification).

In all Controller Level Reset cases, if the media is not usable and an NVM Subsystem Shutdown that includes the controller is neither reported as in progress nor reported as complete (i.e., the CSTS.ST bit is cleared to '0' or the CSTS.SHST field is cleared to 00b), then the controller is permitted to initialize the media for use upon completion of the Controller Level Reset.

To continue after a Controller Level Reset, the host should:

- update transport specific state and controller property state as appropriate;



- set the CC.EN bit to '1';
- wait for the CSTS.RDY bit to be set to '1';
- configure the controller using Admin commands as needed;
- create I/O Completion Queues and I/O Submission Queues as needed; and
- proceed with normal I/O operations.

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

If the host is no longer able to communicate with the controller before that host receives either:

- completions for all outstanding commands submitted to that controller (refer to section 3.4.5); or
- a CSTS.RDY bit value cleared to '0',

then it is strongly recommended that the host take the steps described in section 9.6 to avoid possible data corruption caused by interaction between outstanding commands and subsequent commands submitted by that host to another controller.

### 3.7.3 Queue Level Reset

The host may reset and/or reconfigure the I/O Submission and I/O Completion Queues by resetting them. A queue level reset is performed by deleting and then recreating the queue. In this process, the host should wait for all pending commands to the appropriate I/O Submission Queue(s) to complete.

To perform the queue level reset on a controller using the memory-based transport model, the host submits the Delete I/O Submission Queue or Delete I/O Completion Queue command to the Admin Queue specifying the identifier of the queue to be deleted. After successful command completion of the queue delete operation, the host then recreates the queue by submitting the Create I/O Submission Queue or Create I/O Completion Queue command. As part of the creation operation, the host may modify the attributes of the queue. Note that if a queue level reset is performed on an I/O Completion Queue, the I/O Submission Queues that are utilizing the I/O Completion Queue should be deleted before the I/O Completion Queue is reset and recreated after the I/O Completion Queue is recreated. The behavior of an I/O Submission Queue without a corresponding I/O Completion Queue is undefined.

To perform the queue level reset on a controller using the message-based transport model, the host sends a Disconnect command to the I/O Queue which is to be deleted. After successful command completion of the Disconnect command, the host then recreates the I/O Submission Queue and I/O Completion Queue by submitting the Connect command with a QID specified that is not 00h. As part of the Connect command, the host may modify the attributes of the I/O queues.

The host should ensure that the appropriate I/O Submission Queue or I/O Completion Queue is idle before deleting that queue. Submitting a queue deletion command causes any pending commands to be aborted by the controller; this may or may not result in a completion queue entry being posted for the aborted command(s).

## 3.8 NVM Capacity Model

### 3.8.1 Overview

NVM subsystems may report capacity-related information for multiple entities within the NVM subsystem. This capacity reporting model includes capacity reporting for the NVM subsystem, the domain (refer to section 3.2.5), the Endurance Group (refer to section 3.2.3), the NVM Set (refer to section 3.2.2), namespaces that contain formatted storage (refer to section 3.2.1), and the Media Unit (refer to section 1.5.54). Some, all, or none of this reporting may be supported by an NVM subsystem.

Figure 14 shows the hierarchical relationships of the entities within an NVM subsystem which are used to manage NVM capacity.

The capacity in an NVM Set is able to be allocated to one or more namespaces, and each namespace that contains formatted storage exists entirely in that NVM Set (refer to section 3.2.2). Not all of the capacity in the NVM Set is required to be allocated to a namespace.

If the controller supports NVM Sets, then the capacity in an Endurance Group is able to be allocated to one or more NVM Sets and each NVM Set exists entirely in that Endurance Group (refer to section 3.2.3). Not all of the capacity in an Endurance Group is required to be allocated to an NVM Set.

If the controller supports Endurance Groups and does not indicate support for NVM Sets, then in all data structures that contain an NVMSETID field, the NVMSETID field shall be cleared to 0h.

If the controller does not support Endurance Groups, then in all data structures that contain an ENDGID field, the ENDGID field shall be cleared to 0h.

If the controller supports Endurance Groups, then the capacity in a domain is able to be allocated to one or more Endurance Groups, and each Endurance Group exists entirely in that domain (refer to section 3.2.5). Not all of the capacity in a domain is required to be allocated to an Endurance Group.

NVM subsystems may report the composition of Endurance Groups and NVM Sets in terms of Media Units. Each Media Unit is allocated to exactly one Endurance Group. If NVM Sets are supported, each Media Unit is allocated to exactly one NVM Set. Data is transferred to and from Media Units via Channels. Each Media Unit is connected to one or more Channels. Each Channel is connected to one or more Media Units.

A host uses Capacity Management (refer to section 8.1.4) to allocate:

- a) Domain capacity to Endurance Groups;
- b) Endurance Group capacity to NVM Sets;
- c) Media Units to Endurance Groups; and
- d) Media Units to NVM Sets,

as part of creating those entities.

A host uses the Namespace Management create operation (refer to section 8.1.15) to allocate capacity to namespaces that contain formatted storage.

### 3.8.2 Media Unit Organization Examples

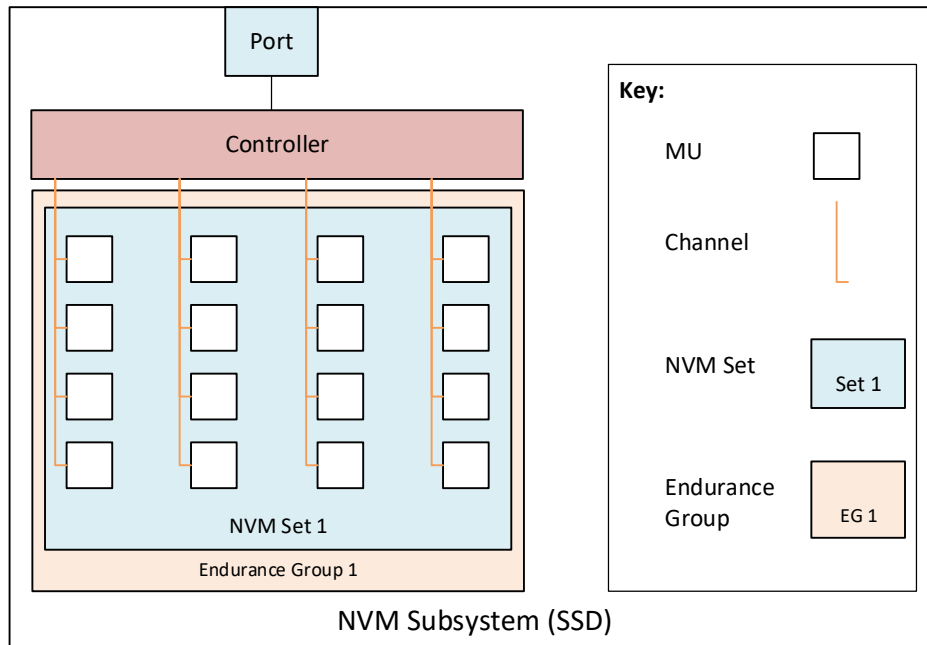
Allocation of Media Units is used to organize the physical NVM resources in an NVM subsystem to meet particular performance goals.

The following examples show an NVM subsystem with all resources in a single domain. The domain has four Channels, with four Media Units attached to each Channel.

#### 3.8.2.1 Simple NVM Subsystem

Figure 86 shows an example of a single domain NVM subsystem where endurance is managed across all media units. The performance goal is maximum bandwidth, which is achieved by allowing each read or write operation to simultaneously access all Media Units. All Media Units are in the same Endurance Group and in the same NVM Set.

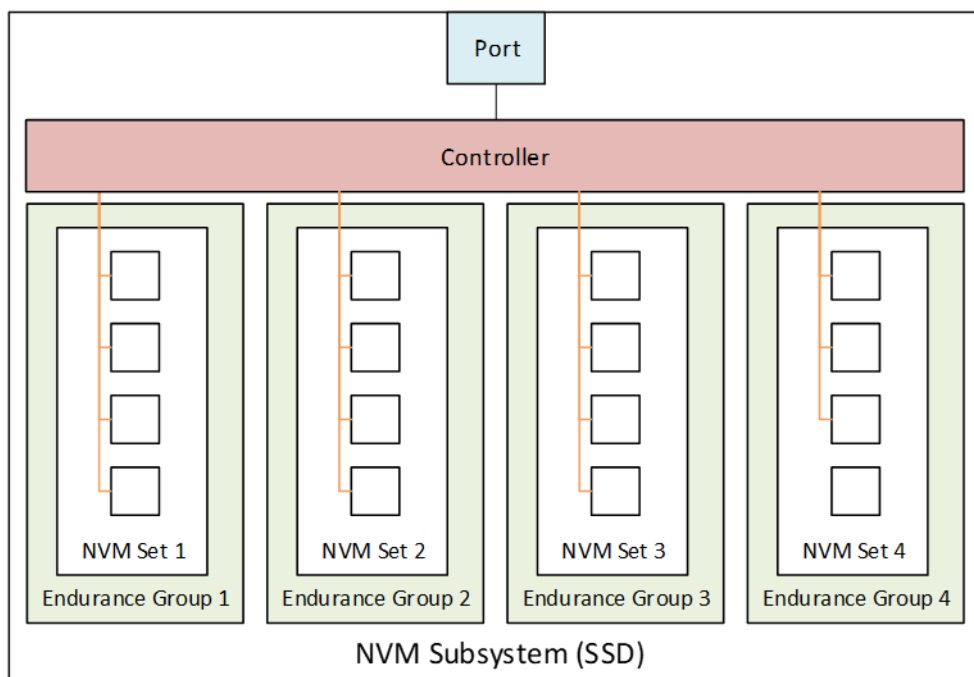
**Figure 86: Simple NVM Subsystem**



The Capacity Configuration Descriptor for this example contains one Endurance Group Configuration Descriptor. The Endurance Group Configuration Descriptor contains one NVM Set Identifier and four Channel Configuration Descriptors. Each Channel Configuration Descriptor contains four Media Unit Configuration Descriptors.

### 3.8.2.2 Vertically-Organized NVM Subsystem

Figure 87 shows an example of a single domain NVM subsystem where the performance goal is isolation among four NVM Sets at the cost of bandwidth. Endurance is managed separately for each set. Media Units sharing a Channel are allocated to the same Endurance Group. All Media Units in an Endurance Group are allocated to the same NVM Set. The bandwidth for any NVM Set is likely to be less than or equal to the bandwidth of the Channel of that NVM Set.

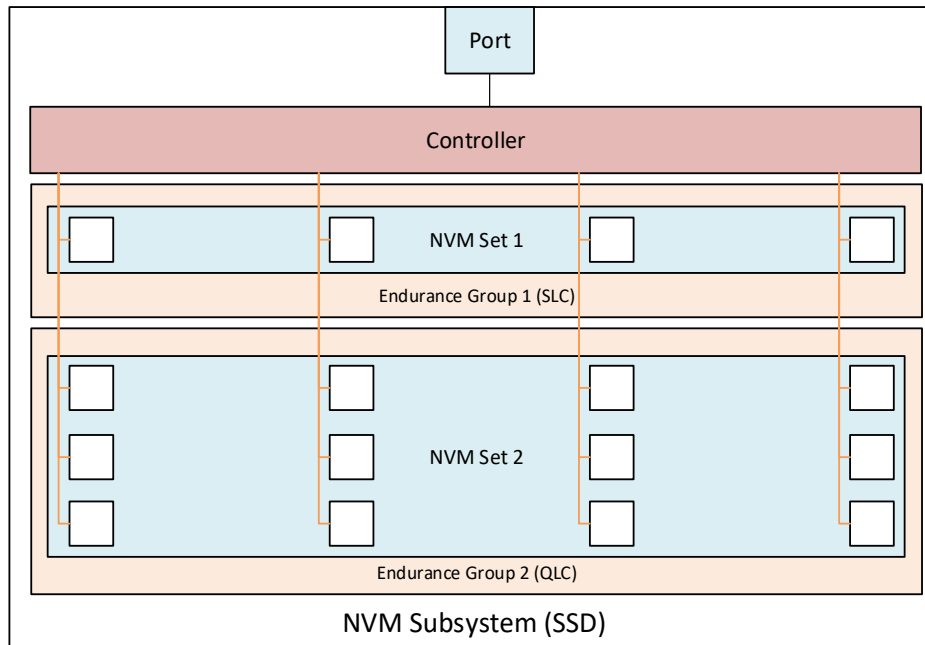
**Figure 87: Vertically-Organized NVM Subsystem**

The Capacity Configuration Descriptor for this example contains four Endurance Group Configuration Descriptors. Each Endurance Group Configuration Descriptor contains one NVM Set Identifier and one Channel Configuration Descriptor. Each Channel Configuration Descriptor contains four Media Unit Configuration Descriptors.

### 3.8.2.3 Horizontally-Organized Dual NAND NVM Subsystem

Figure 88 shows an example of a single domain NVM subsystem where the Media Units are NAND which is capable of being operated as QLC or at a lower density. The performance goal is for maximum bandwidth to a small NVM Set operating as SLC and to a larger NVM Set operating as QLC.

**Figure 88: Horizontally-Organized Dual NAND NVM Subsystem**



The Capacity Configuration Descriptor for this example contains two Endurance Group Configuration Descriptors. The first Endurance Group Configuration Descriptor for this example:

- indicates a Capacity Adjustment Factor of approximately 400;
- contains one NVM Set Identifier; and
- contains four Channel Configuration Descriptors. Each Channel Configuration Descriptor contains one Media Unit Configuration Descriptor.

The second Endurance Group Configuration Descriptor for this example:

- indicates a Capacity Adjustment Factor of 100;
- contains one NVM Set Identifier; and
- contains four Channel Configuration Descriptors. Each Channel Configuration Descriptor contains three Media Unit Configuration Descriptors.

### 3.8.3 Capacity Reporting

For an NVM subsystem that does not support multiple domains, the capacity information reported in the Identify Controller data structure (i.e., the TNVMCAP field and the UNVMCAP field in Figure 312) describes the capacity for the NVM subsystem. If the MEGCAP field is non-zero, that field indicates the largest entity (e.g., Endurance Group, NVM Set, namespace that contains formatted storage) that may be created in the NVM subsystem.

For an NVM subsystem that supports multiple domains, the capacity information reported in the Identify Controller data structure describes the capacity accessible by the controller processing the Identify command. The host may use the Identify command to access the Domain List data structure (refer to section 5.1.13.2.15) to determine the domains that are accessible by the controller and the capacity information for each of those domains. If the Max Endurance Group Domain Capacity field is non-zero, then the field describes the largest entity (e.g., Endurance Group, namespace that contains formatted storage) that may be created by this controller in the domain described by that Domain Attributes Entry.

For an NVM subsystem that supports Endurance Groups (refer to section 3.2.3), the host may use the Identify command to access the Endurance Group List data structure (refer to section 5.1.13.2.16) to determine the Endurance Groups that are accessible by the controller. To determine the capacity

information for each Endurance Group, the host uses the Get Log Page command to access the Endurance Group Information log page (refer to section 5.1.12.1.10).

For an NVM subsystem that supports NVM Sets (refer to section 3.2.2), the host may use the Identify command to access the NVM Set List data structure (refer to section 5.1.13.2.4) to determine the NVM Sets that are accessible by the controller and the capacity information for each of those NVM Sets.

For the management of Endurance Groups, NVM Sets, and namespaces that contain formatted storage, Figure 89 describes the effects of the support of NVM Sets, Endurance Groups, and domains on which capacity information is used for each management operation.

**Figure 89: Capacity Information Field Usage**

Entity being created / deleted	NVM Sets supported	Endurance Groups supported	Domains supported	Capacity information used <sup>5</sup>
Endurance Group <sup>11</sup>	n/a	Yes	No	NVM subsystem <sup>7</sup>
			Yes	Domain <sup>8</sup>
NVM Set <sup>11</sup>	Yes	Yes <sup>6</sup>	n/a	Endurance Group <sup>9</sup>
Namespace that contains formatted storage <sup>12</sup>	No	No	No	NVM subsystem <sup>7</sup>
			Yes	Domain <sup>8</sup>
	Yes	Yes <sup>6</sup>	n/a	Endurance Group <sup>9</sup>
			n/a	NVM Set <sup>10</sup>

Notes:

- This information described in this column is used by the host for creating the entity (e.g., to determine if there is sufficient available capacity) and this information is altered by the controller as a result of the creation or deletion of the entity (e.g., unallocated capacity decreased as a result of entity creation, or unallocated capacity increased as a result of entity deletion).
- NVM Set support requires support for Endurance Groups as described in section 3.2.2.
- Capacity information in the Identify Controller data structure (i.e., TNVMCAP field, UNVMCAP field, and MEGCAP fields (refer to Figure 312)).
- Capacity information in the Domain Attributes Entry (i.e., Total Domain Capacity field, Unallocated Domain Capacity field, and Max Endurance Group Domain Capacity field (refer to Figure 324)).
- Capacity information in the Endurance Group Information log page (i.e., TEGCAP field, UEGCAP field (refer to Figure 218)).
- Capacity information in the NVM Set Attributes Entry (i.e., Total NVM Set Capacity field, and Unallocated NVM Set Capacity field (refer to Figure 318)).
- Endurance Groups and NVM Sets are created and deleted using the Capacity Management command (refer to section 5.1.3)
- Namespaces are created and deleted using the Namespace Management command (refer to section 8.1.15). Namespaces are deleted using the Capacity Management command.

### 3.9 Keep Alive

The Keep Alive capability uses the Keep Alive Timer on a controller as a watchdog timer to detect communication failures (e.g., transport failure, host failure, or controller failure) between a host and a controller. If the Keep Alive Timer feature is supported (i.e., the KAS field is set to a non-zero value (refer to Figure 312)), the controller shall support the Keep Alive command.

The term Keep Alive Timeout Time (KATT) refers to the time indicated by the value of the Keep Alive Timeout field on a controller (refer to Figure 397).

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

- the minimum Keep Alive Timeout value, if any;
- the maximum Keep Alive Timeout value, if any; and
- if the Keep Alive Timer feature is required to be supported and enabled.

NVMe Transports that do not detect a connection loss in a timely manner shall require that the Keep Alive Timer feature be supported and enabled.

### 3.9.1 Keep Alive Timer Configuration

A host configures the Keep Alive Timer feature by specifying a Keep Alive Timeout value in:

- the KATO field of a Fabric Connect command (refer to section 6.3); or
- the KATO field of a Set Features command specifying the Keep Alive Timer feature (refer to section 5.1.25.1.8).

If an NVMe Transport binding specification requires the use of the Keep Alive Timer feature, and a command attempts to disable the Keep Alive Timer by clearing the Keep Alive Timeout value to 0h, then the controller shall abort that command with a status code of Keep Alive Timeout Invalid and the Keep Alive Timeout value at the controller shall not be changed. If a command attempts to set the Keep Alive Timeout value to a value that exceeds the maximum allowed by the associated NVMe Transport binding specification, then the controller shall abort that command with a status code of Keep Alive Timeout Invalid and the Keep Alive Timeout value at the controller shall not be changed. If a command sets the Keep Alive Timeout value to a non-zero value that is less than the minimum supported by the NVMe Transport or less than the minimum supported by the specific implementation, then the controller sets the Keep Alive Timeout value to that minimum value.

As described in section 5.1.25.1.8, the controller rounds any Keep Alive Timeout value set by the host up to the nearest granularity as reported in the Keep Alive Support (KAS) field (refer to Figure 312). To retrieve the Keep Alive Timeout value being used by the controller, a host may issue a Get Features command for the Keep Alive Timer feature.

### 3.9.2 Keep Alive Timer Activation

The Keep Alive Timer is active if:

- CC.EN is set to '1';
- CSTS.RDY is set to '1';
- CC.SHN is cleared to '00b';
- CSTS.SHST is cleared to '00b'; and
- the Keep Alive Timer feature is enabled as a result of the KATO field being set to a non-zero value (refer to section 3.9.1).

Otherwise, the Keep Alive Timer is inactive, and a Keep Alive Timeout as described in sections 3.9.3.1 and 3.9.4.1 shall not occur on the controller. Activating an inactive Keep Alive Timer (e.g., a Set Features command successfully setting the Keep Alive Timeout value to a non-zero value from a value of 0h, or the host enabling a controller that supports NVMe over Fabrics where the Connect command specified a non-zero Keep Alive Timeout value (refer to Figure 545)) shall initialize the Keep Alive Timer to the Keep Alive Timeout value.

While the Keep Alive Timer is active, the host should ensure that the Admin Submission Queue has space for a Keep Alive command.

### 3.9.3 Command Based Keep Alive

For Command Based Keep Alive, the Keep Alive command is sent periodically from the host to the controller on the Admin Queue. The completion of the Keep Alive command indicates that the host and controller are able to communicate. For message-based transports, the Keep Alive Timeout is the maximum time an association remains established without processing a Keep Alive command.

#### 3.9.3.1 Command Based Keep Alive on the Controller

The controller is using Command Based Keep Alive if the controller supports the Keep Alive Timer feature and the TBKAS bit is cleared to '0' in the Controller Attributes field in the Identify Controller data structure (refer to Figure 312).

For Command Based Keep Alive:

- The controller shall start the Keep Alive Timer if the Keep Alive Timer becomes active (refer to section 3.9.2).
- The controller shall restart the Keep Alive Timer if the Keep Alive Timer is active, and:
  - a Keep Alive command completes successfully; or
  - a Set Features command specifying the Keep Alive Timer feature and a non-zero KATO field (refer to section 5.1.25.1.8) completes successfully.
- The controller shall expire the Keep Alive Timer if:
  - the Keep Alive Timer is active in the controller; and
  - KATT has elapsed since the Keep Alive Timer was most recently started or restarted.

If the Keep Alive Timer on the controller expires, then the controller shall consider a Keep Alive Timeout to have occurred. Upon the occurrence of a Keep Alive Timeout, the controller shall perform the cleanup actions described in section 3.9.5.

### 3.9.3.2 Command Based Keep Alive on the Host

The host may use Command Based Keep Alive regardless of the Keep Alive mode used by the controller. To prevent the controller from detecting a Keep Alive Timeout during the use of Command Based Keep Alive on the host, the host should send Keep Alive commands at  $KATT/2$  to account for delays (e.g., transport round-trip times, transport delays, command processing times, and the Keep Alive Timer granularity) while the Keep Alive Timer is active on the controller. If the host receives a successful completion to a Set Features command for the Keep Alive feature, then the host should adjust the time at which the host sends the next Keep Alive command because the controller restarts the Keep Alive Timer.

If a host detects a Keep Alive Timeout and has outstanding commands for which that host has not received completions (refer to section 3.4.5), then it is strongly recommended that the host take the steps described in section 9.6 to avoid possible data corruption caused by interaction between outstanding commands and subsequent commands submitted by that host to another controller.

For an example host implementation of Command Based Keep Alive, the host maintains a Keep Alive Send Timer for each controller to which the host is connected. The host uses the Keep Alive Send Timer to track when the host sends a Keep Alive command to the corresponding controller. The host does not know when the controller fetches the Keep Alive command. Conservatively, the host assumes the controller fetches the Keep Alive command immediately upon the host sending the Keep Alive command. The host tracks this time as the last expired timestamp of the Keep Alive Send Timer, for use when starting or restarting the Keep Alive Send Timer. While the host does restart the Keep Alive Send Timer after a successful Set Features command for the KATO feature, the host does not change the last expired timestamp when sending that Set Features command because at that point in time the host does not know the results of that command.

This example host implementation of Command Based Keep Alive behaves as follows:

- The host enables the Keep Alive Send Timer if the host requests activation of the Keep Alive Timer on the controller (e.g., the host enables the controller (refer to section 3.9.2)). The host records the current time as the last expired timestamp when the Keep Alive Send Timer becomes enabled.
- The host disables the Keep Alive Send Timer if the host requests deactivation of the Keep Alive Timer on the controller and the host receives a successful response from the controller (e.g., the host disables the controller (refer to section 3.9.2)).
- The host starts a Keep Alive Send Timer if:
  - the Keep Alive Send Timer becomes enabled; or
  - a successful Keep Alive command completion is processed by the host.
- The host restarts a Keep Alive Send Timer if:
  - the Keep Alive Send Timer is enabled;
  - a Keep Alive command is not outstanding; and,



- a successful Set Features command completion is processed by the host where the command specified the Keep Alive Timer feature (i.e., Feature Identifier 0Fh) and a non-zero KATO field.
- If a Keep Alive Send Timer starts or restarts, the host sets the Keep Alive Send Timer to:
  - $KATT/2$  minus the time elapsed since the last expired timestamp; or
  - zero, if the time elapsed since the last expired timestamp is greater than  $KATT/2$ .
- The host stops a Keep Alive Send Timer if the Keep Alive Send Timer becomes disabled.
- If a Keep Alive Send Timer expires, (i.e., the Keep Alive Send Timer is still enabled and time has elapsed equal to the value to which the Keep Alive Send Timer was set since the host most recently started or restarted the Keep Alive Send Timer), then the host records the current time as the last expired timestamp and sends a Keep Alive command.
- The host detects a Keep Alive Timeout if the host sends a Keep Alive command and does not receive a completion for the Keep Alive command before KATT elapses from when the Keep Alive command was sent.

### 3.9.4 Traffic Based Keep Alive

Traffic Based Keep Alive allows the host and controller to avoid a Keep Alive Timeout in the presence of Admin or I/O command processing without sending Keep Alive commands.

#### 3.9.4.1 Traffic Based Keep Alive on the Controller

If the controller supports the Keep Alive Timer feature, then support for Traffic Based Keep Alive is indicated by the TBKAS bit in the Controller Attributes field in the Identify Controller data structure (refer to Figure 312). If the Controller does not support Traffic Based Keep Alive (i.e., the TBKAS bit is cleared to '0'), then the operation of the Keep Alive Timer feature is described in section 3.9.3.

For Traffic Based Keep Alive:

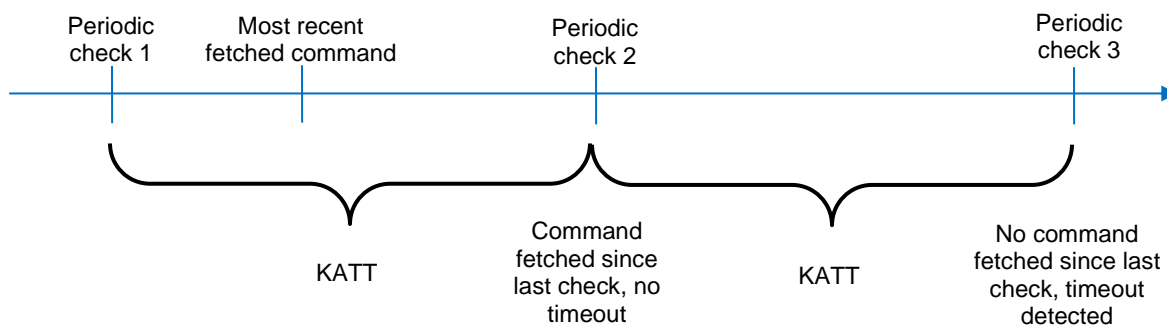
- The controller shall start a Keep Alive Timeout Interval if the Keep Alive Timer becomes active (refer to section 3.9.2).
- The controller shall consider a Keep Alive Timeout to have occurred if:
  - the Keep Alive Timer is active;
  - KATT has elapsed since the start of the most recent Keep Alive Timeout Interval; and
  - no Admin command or I/O command was fetched by the controller during the Keep Alive Timeout Interval.
- The controller shall end a Keep Alive Timeout Interval and:
  - not start a new Keep alive Timeout Interval if:
    - a Keep Alive Timeout occurs; or
    - the Keep Alive Timer becomes inactive (refer to section 3.9.2).
  - start a new Keep Alive Timeout Interval if the Keep Alive Timer is active, and:
    - a Keep Alive command completes successfully;
    - a Set Features command specifying the Keep Alive Timer feature and a non-zero KATO field (refer to section 5.1.25.1.8) completes successfully; or
    - KATT has elapsed since the start of the Keep Alive Timeout Interval and a Keep Alive Timeout did not occur during the Keep Alive Timeout Interval (e.g., an Admin command or an I/O command was fetched by the controller during the Keep Alive Timeout Interval).

Upon the occurrence of a Keep Alive Timeout, the controller shall perform the cleanup actions described in section 3.9.5.

A controller using Traffic Based Keep Alive may require up to  $2 * KATT$  after the controller fetches the most recent command to detect a Keep Alive Timeout as shown in Figure 90.

**Figure 90: Detecting Timeout Takes up to 2 \* KATT**

Figure 90 shows that periodic check 3, not periodic check 2, detects the Keep Alive Timeout. Therefore, the time between fetching the most recent command and the check that detects the timeout (i.e., periodic



check 3 in Figure 90) is up to 2 \* KATT.

### 3.9.4.2 Traffic Based Keep Alive on the Host

The host is able to use Traffic Based Keep Alive only if the controller is also using Traffic Based Keep Alive. The host should not use Traffic Based Keep Alive if the controller is not using Traffic Based Keep Alive because a controller that uses Command Based Keep Alive detects a Keep Alive Timeout based on the absence of Keep Alive commands, not the absence of all commands.

Traffic Based Keep Alive on the host is the same as Command Based Keep Alive on the host (refer to section 3.9.3.2), with two exceptions:

- The host is not required to submit a Keep Alive command if the host submitted an Admin command or I/O command and the host processed the completion for that command since the most recent time the host checked whether sending a Keep Alive command was necessary.
- To prevent the controller from detecting a Keep Alive Timeout during the use of Traffic Based Keep Alive on the host, the host should check for sending a Keep Alive command at a rate of  $KATT/4$ , instead of sending a Keep Alive command at a rate of  $KATT/2$ , while the Keep Alive Timer is active on the controller.

Like Command Based Keep Alive, if the host receives a successful completion to a Set Features command for the Keep Alive feature, then the host should adjust the time at which the host checks for sending the next Keep Alive command because the controller restarted the Keep Alive Timer.

If a host detects a Keep Alive Timeout and has outstanding commands for which that host has not received completions (refer to section 3.4.5), then it is strongly recommended that the host take the steps described in section 9.6 to avoid possible data corruption caused by interaction between outstanding commands and subsequent commands submitted by that host to another controller.

For an example host implementation of Traffic Based Keep Alive, the host maintains a Keep Alive Send Timer for each controller to which that host is connected. The host uses the Keep Alive Send Timer to track when the host checks for sending a Keep Alive command to the corresponding controller. The host does not know when the controller fetches the Keep Alive command. Conservatively, the host assumes the controller fetches the Keep Alive command immediately upon the host sending the Keep Alive command. Whether or not the host sends a Keep Alive command after the Keep Alive Send Timer expires, the host tracks this time as the last expired timestamp of the Keep Alive Send Timer, for use when starting or restarting the Keep Alive Send Timer. While the host does restart the Keep Alive Send Timer after a successful Set Features command for the KATO feature, the host does not change the last expired timestamp when sending that Set Features command because at that point in time the host does not know the results of that command.

This example host implementation of Traffic Based Keep Alive behaves as follows:

- The host enables the Keep Alive Send Timer if the host requests activation of the Keep Alive Timer on the controller (e.g., the host enables the controller (refer to section 3.9.2)). The host records the current time as the last expired timestamp when the Keep Alive Send Timer becomes enabled.
- The host disables the Keep Alive Send Timer if the host requests deactivation of the Keep Alive Timer on the controller and the host receives a successful response from the controller (e.g., the host disables the controller (refer to section 3.9.2)).
- The host starts a Keep Alive Send Timer if:
  - the Keep Alive Send Timer becomes enabled; or
  - a successful Keep Alive command completion is processed by the host.
- The host restarts a Keep Alive Send Timer if:
  - the Keep Alive Send Timer is enabled;
  - a Keep Alive command is not outstanding; and
  - a successful Set Features command completion is processed by the host where the command specified the Keep Alive Timer feature (i.e., Feature Identifier 0Fh) and a non-zero KATO field.
- If a Keep Alive Send Timer starts or restarts, the host sets the Keep Alive Send Timer to:
  - $KATT/4$  minus the time elapsed since the last expired timestamp; or
  - zero, if the time elapsed since the last expired timestamp is greater than  $KATT/4$ .
- The host stops a Keep Alive Send Timer if the Keep Alive Send Timer becomes disabled
- If a Keep Alive Send Timer expires, (i.e., the Keep Alive Send Timer is still enabled and time has elapsed equal to the value to which the Keep Alive Send Timer was set since the host most recently started or restarted the Keep Alive Send Timer), then the host records the current time as the last expired timestamp and the host either:
  - starts a new Keep Alive Send Timer, if at least one Admin command or I/O command was submitted to the controller and the completion of that command was processed by the host since the last Keep Alive Send Timer started (not restarted); or
  - sends a Keep Alive command.
- The host detects a Keep Alive Timeout if the host sends a Keep Alive command and does not receive a completion for that Keep Alive command before KATT elapses from when the Keep Alive command was submitted.

### 3.9.5 Keep Alive Timeout Cleanup

If a controller detects a Keep Alive Timeout, then the controller shall perform the following actions within the time specified by the CQT field (refer to Figure 312):

- record an Error Information Log Entry with the status code Keep Alive Timeout Expired;
- stop processing commands;
- set the Controller Fatal Status (CSTS.CFS) bit to '1'; and
- for message-based transports:
  - terminate the NVMe Transport connections for this association; and
  - break the host to controller association.

For message-based transports, after completing these steps, a controller may accept a Connect command (refer to section 6.3) for the Admin Queue from the same or another host in order to form a new association.

If a host detects a Keep Alive Timeout and has outstanding commands for which that host has not received completions (refer to section 3.4.5), then it is strongly recommended that the host takes steps described in

section 9.6 to avoid possible data corruption caused by interaction between outstanding commands and subsequent commands submitted by that host to another controller.

### 3.10 Privileged Actions

Privileged actions are actions (e.g., command, property write) that affect or have the potential to affect the state beyond the controller and attached namespaces.

Examples of privileged actions are:

- Admin commands including Namespace Management, Namespace Attachment, Virtualization Management, Format NVM, Set Features with Feature Identifier 17h (i.e., Sanitize Config, refer to section 5.1.25.1.15), Sanitize, Capacity Management, Controller Data Queue, Migration Receive, Migration Send, Track Send, and Track Receive;
- Property Writes including NVM Subsystem Reset; and
- Some Vendor specific commands and properties.

### 3.11 Firmware Update Process

The process for a firmware update to be activated in a domain (refer to section 3.2.5) by a reset is:

1. The host issues a Firmware Image Download command to download the firmware image to a controller. There may be multiple portions of the firmware image to download, thus the offset for each portion of the firmware image being downloaded on that controller is specified in the Firmware Image Download command. The data provided in the Firmware Image Download command should conform to the Firmware Update Granularity indicated in the Identify Controller data structure or the firmware update may fail;
2. After the firmware is downloaded to that controller, the next step is for the host to submit a Firmware Commit command to that controller. The Firmware Commit command verifies that the last firmware image downloaded is valid and commits that firmware image to the firmware slot indicated for future use. A firmware image that does not start at offset zero, contains gaps, or contains overlapping regions is considered invalid. A controller may employ additional vendor specific means (e.g., checksum, CRC, cryptographic hash, or a digital signature) to determine the validity of a firmware image:
  - a. The Firmware Commit command may also be used to activate a firmware image associated with a previously committed firmware slot;
3. The host performs an action that results in a 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:
  - 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 5.1.8.1);

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

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

1. The host issues a Firmware Image Download command to download the firmware image to a controller. There may be multiple portions of the firmware image to download, thus the offset for each portion of the firmware image being downloaded on that controller is specified in the Firmware Image Download command. The data provided in the Firmware Image Download command should conform to the Firmware Update Granularity indicated in the Identify Controller data structure or the firmware update may fail;

2. The host submits a Firmware Commit command on that controller with a Commit Action of 011b which specifies that the firmware image should be activated immediately without reset. The downloaded firmware image shall replace the firmware image in the firmware slot. If no firmware image was downloaded since the last reset or Firmware Commit command, (i.e., the first step was skipped), then that controller shall verify and activate the firmware image in the specified slot. If that controller starts to activate the firmware image, any controllers affected by the new firmware image send a Firmware Activation Starting asynchronous event to the host if Firmware Activation Notices are enabled (refer to Figure 391):
  - a. The Firmware Commit command may also be used to activate a firmware image associated with a previously committed firmware slot;
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 the MTFA field 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.

If the controller transitions to the D3<sub>cold</sub> state (refer to the PCI Express Base Specification) after the submission of a Firmware Commit command that attempts to activate a firmware image and before the completion of that command, then the controller may resume operation with either the firmware image active at the time the Firmware Commit command was submitted or the firmware image that was activated by that command.

If the firmware image is not able to be successfully loaded, then the controller shall revert to the firmware image present in the most recently activated firmware slot or the baseline read-only firmware image, if available, and indicate the failure as an asynchronous event with a Firmware Image Load Error. If the controller changes (e.g., reverts) the firmware image while a sanitize operation is in progress, then that sanitize operation fails (refer to section 8.1.24.3).

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

- this is a pending firmware activation with reset; and
- a controller aborts subsequent Sanitize commands with a status code indicating the appropriate reset required to activate the pending activation as defined in section 5.1.22.

Host software should not overlap firmware/boot partition image update command sequences (refer to section 1.5.41). During a firmware image update command sequence, if a Firmware Image Download command or a Firmware Commit command is submitted for another firmware/boot partition image update command sequence, the results of both that command and the in-progress firmware image update are undefined.

Host software should use the same controller or Management Endpoint (refer to the NVM Express Management Interface Specification) for all commands that are part of a firmware image update command sequence. If the commands for a single firmware/boot partition image update command sequence are submitted to more than one controller and/or Management Endpoint, the controller may abort the Firmware Commit command with Invalid Firmware Image status.

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

## 4 Data Structures

This section describes data structures used by the NVM Express Interface.

### 4.1 Submission Queue Entry

#### 4.1.1 Admin Command and I/O Command Common SQE

Each Common Command Format command is 64 bytes in size.

Command Dword 0, Namespace Identifier, Metadata Pointer, PRP Entry 1, PRP Entry 2, SGL Entry 1, and Metadata SGL Segment Pointer have common definitions for all Admin commands and I/O commands for all I/O Command Sets. Metadata Pointer, PRP Entry 1, PRP Entry 2, and Metadata SGL Segment Pointer are not used by all commands. Command Dword 0 is defined in Figure 91.

**Figure 91: Command Dword 0**

Bits	Description										
31:16	<p><b>Command Identifier (CID):</b> This field specifies a unique identifier for the command when combined with the Submission Queue identifier.</p> <p>The value of FFFFh should not be used as the Error Information log page (refer to section 5.1.12.1.2) uses this value to indicate an error is not associated with a particular command.</p>										
15:14	<p><b>PRP or SGL for Data Transfer (PSDT):</b> This field specifies whether PRPs or SGLs are used for any data transfer associated with the command. PRPs shall be used for all Admin commands for NVMe over PCIe implementations. SGLs shall be used for all Admin and I/O commands for NVMe over Fabrics implementations (i.e., this field set to 01b). An NVMe Transport may support only specific values (refer to the applicable NVMe Transport binding specification for details).</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td><b>PRPS Used:</b> PRPs are used for this transfer.</td> </tr> <tr> <td>01b</td> <td><b>SGLs Used MPTR Address:</b> SGLs are used for this transfer. If used, Metadata Pointer (MPTR) contains an address of a single contiguous physical buffer. Refer to the Metadata Buffer Alignment (MBA) bit of the SGLS field in the Identify Controller data structure (refer to Figure 312) for alignment requirements.</td> </tr> <tr> <td>10b</td> <td><b>SGLs Used MPTR SGL Segment:</b> SGLs are used for this transfer. If used, Metadata Pointer (MPTR) contains an address of an SGL segment containing exactly one SGL Descriptor that is qword aligned.</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table> <p>If there is metadata that is not interleaved with the user data, as specified in the Format NVM command, then the Metadata Pointer (MPTR) field is used to point to the metadata. The definition of the Metadata Pointer field is dependent on the setting in this field. Refer to Figure 92.</p>	Value	Definition	00b	<b>PRPS Used:</b> PRPs are used for this transfer.	01b	<b>SGLs Used MPTR Address:</b> SGLs are used for this transfer. If used, Metadata Pointer (MPTR) contains an address of a single contiguous physical buffer. Refer to the Metadata Buffer Alignment (MBA) bit of the SGLS field in the Identify Controller data structure (refer to Figure 312) for alignment requirements.	10b	<b>SGLs Used MPTR SGL Segment:</b> SGLs are used for this transfer. If used, Metadata Pointer (MPTR) contains an address of an SGL segment containing exactly one SGL Descriptor that is qword aligned.	11b	Reserved
Value	Definition										
00b	<b>PRPS Used:</b> PRPs are used for this transfer.										
01b	<b>SGLs Used MPTR Address:</b> SGLs are used for this transfer. If used, Metadata Pointer (MPTR) contains an address of a single contiguous physical buffer. Refer to the Metadata Buffer Alignment (MBA) bit of the SGLS field in the Identify Controller data structure (refer to Figure 312) for alignment requirements.										
10b	<b>SGLs Used MPTR SGL Segment:</b> SGLs are used for this transfer. If used, Metadata Pointer (MPTR) contains an address of an SGL segment containing exactly one SGL Descriptor that is qword aligned.										
11b	Reserved										
13:10	Reserved										
09:08	<p><b>Fused Operation (FUSE):</b> In a fused operation, a complex command is created by “fusing” together two simpler commands. Refer to section 3.4.2. This field specifies whether this command is part of a fused operation and if so, which command it is in the sequence.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Normal operation</td> </tr> <tr> <td>01b</td> <td>First command of Fused operation</td> </tr> <tr> <td>10b</td> <td>Second command of Fused operation</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	Normal operation	01b	First command of Fused operation	10b	Second command of Fused operation	11b	Reserved
Value	Definition										
00b	Normal operation										
01b	First command of Fused operation										
10b	Second command of Fused operation										
11b	Reserved										

**Figure 91: Command Dword 0**

Bits	Description																	
07:00	<b>Opcode (OPC):</b> This field specifies the opcode of the command to be executed as shown here:																	
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>07:02</td> <td><b>Function (FN):</b> This field contains a value that, in combination with the other fields in the Opcode data structure, creates a unique combined opcode value.</td> </tr> <tr> <td rowspan="4">01:00</td> <td><b>Data Transfer Direction (DTD):</b> This field indicates the direction of a data transfer, if any. All options of the command shall transfer data as specified or transfer no data. All commands, including vendor specific commands, shall follow this convention.</td> </tr> <tr> <td> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td><b>No Data Transfer:</b> No data is transferred.</td> </tr> <tr> <td>01b</td> <td><b>Host to Controller Transfer:</b> Data is transferred from the host to the controller.</td> </tr> <tr> <td>10b</td> <td><b>Controller to Host Transfer:</b> Data is transferred from the controller to the host.</td> </tr> <tr> <td>11b</td> <td><b>Bi-Directional Transfer:</b> Data is transferred from the host to the controller and from the controller to the host.</td> </tr> </tbody> </table> </td> </tr> </tbody> </table>	Bits	Description	07:02	<b>Function (FN):</b> This field contains a value that, in combination with the other fields in the Opcode data structure, creates a unique combined opcode value.	01:00	<b>Data Transfer Direction (DTD):</b> This field indicates the direction of a data transfer, if any. All options of the command shall transfer data as specified or transfer no data. All commands, including vendor specific commands, shall follow this convention.	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td><b>No Data Transfer:</b> No data is transferred.</td> </tr> <tr> <td>01b</td> <td><b>Host to Controller Transfer:</b> Data is transferred from the host to the controller.</td> </tr> <tr> <td>10b</td> <td><b>Controller to Host Transfer:</b> Data is transferred from the controller to the host.</td> </tr> <tr> <td>11b</td> <td><b>Bi-Directional Transfer:</b> Data is transferred from the host to the controller and from the controller to the host.</td> </tr> </tbody> </table>	Value	Definition	00b	<b>No Data Transfer:</b> No data is transferred.	01b	<b>Host to Controller Transfer:</b> Data is transferred from the host to the controller.	10b	<b>Controller to Host Transfer:</b> Data is transferred from the controller to the host.	11b	<b>Bi-Directional Transfer:</b> Data is transferred from the host to the controller and from the controller to the host.
	Bits	Description																
	07:02	<b>Function (FN):</b> This field contains a value that, in combination with the other fields in the Opcode data structure, creates a unique combined opcode value.																
	01:00	<b>Data Transfer Direction (DTD):</b> This field indicates the direction of a data transfer, if any. All options of the command shall transfer data as specified or transfer no data. All commands, including vendor specific commands, shall follow this convention.																
<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td><b>No Data Transfer:</b> No data is transferred.</td> </tr> <tr> <td>01b</td> <td><b>Host to Controller Transfer:</b> Data is transferred from the host to the controller.</td> </tr> <tr> <td>10b</td> <td><b>Controller to Host Transfer:</b> Data is transferred from the controller to the host.</td> </tr> <tr> <td>11b</td> <td><b>Bi-Directional Transfer:</b> Data is transferred from the host to the controller and from the controller to the host.</td> </tr> </tbody> </table>		Value	Definition	00b	<b>No Data Transfer:</b> No data is transferred.	01b	<b>Host to Controller Transfer:</b> Data is transferred from the host to the controller.	10b	<b>Controller to Host Transfer:</b> Data is transferred from the controller to the host.	11b	<b>Bi-Directional Transfer:</b> Data is transferred from the host to the controller and from the controller to the host.							
Value		Definition																
00b		<b>No Data Transfer:</b> No data is transferred.																
01b	<b>Host to Controller Transfer:</b> Data is transferred from the host to the controller.																	
10b	<b>Controller to Host Transfer:</b> Data is transferred from the controller to the host.																	
11b	<b>Bi-Directional Transfer:</b> Data is transferred from the host to the controller and from the controller to the host.																	

The Common Command Format is defined in Figure 92. Any additional I/O Command Set defined in the future may use an alternate command size or format.

SGLs shall not be used for Admin commands in NVMe over PCIe implementations.

**Figure 92: Common Command Format**

Bytes	Description
03:00	<b>Command Dword 0 (CDW0):</b> This field is common to all commands and is defined in Figure 91.
07:04	<p><b>Namespace Identifier (NSID):</b> This field specifies the namespace that this command applies to. If the namespace identifier is not used for the command, then this field shall be cleared to 0h. The value FFFFFFFFh in this field is a broadcast value (refer to section 3.2.1.2), where the scope (e.g., the NVM subsystem, all attached namespaces, or all namespaces in the NVM subsystem) is dependent on the command. Refer to Figure 141 and Figure 555 for commands that support the use of the value FFFFFFFFh in this field.</p> <p>Specifying an inactive namespace identifier (refer to section 3.2.1.4) in a command that uses the namespace identifier shall cause the controller to abort the command with a status code of Invalid Field in Command, unless otherwise specified. Specifying an invalid namespace identifier (refer to section 3.2.1.2) in a command that uses the namespace identifier shall cause the controller to abort the command with a status code of Invalid Namespace or Format, unless otherwise specified.</p> <p>If the namespace identifier is used for the command (refer to Figure 141), the value FFFFFFFFh is not supported for that command, and the host specifies a value of FFFFFFFFh, then the controller shall abort the command with a status code of Invalid Field in Command, unless otherwise specified.</p> <p>If the namespace identifier is not used for the command and the host specifies a value from 1h to FFFFFFFFh, then the controller shall abort the command with a status code of Invalid Field in Command, unless otherwise specified.</p>
11:08	<b>Command Dword 2 (CDW2):</b> This field is command specific Dword2.
15:12	<b>Command Dword 3 (CDW3):</b> This field is command specific Dword3.



**Figure 92: Common Command Format**

Bytes	Description
23:16	<p><b>Metadata Pointer (MPTR):</b> If CDW0.PSDT (refer to Figure 91) is cleared to 00b, then this field shall contain the address of a contiguous physical buffer of metadata and that address shall be dword aligned (i.e., bits 1:0 cleared to 00b). The controller is not required to check that bits 1:0 are cleared to 00b. The controller may report an error of Invalid Field in Command if bits 1:0 are not cleared to 00b. If the controller does not report an error of Invalid Field in Command, then the controller shall operate as if bits 1:0 are cleared to 00b.</p> <p>If CDW0.PSDT is set to 01b, then this field shall contain the address of a contiguous physical buffer of metadata. Refer to the Metadata Buffer Alignment (MBA) bit of the SGLS field in the Identify Controller data structure for alignment requirements.</p> <p>If CDW0.PSDT is set to 10b, then this field shall contain the address of an SGL segment that contains exactly one SGL Descriptor. The address of that SGL segment shall be qword aligned (i.e., bits 2:0 cleared to 000b). The SGL Descriptor contained in that SGL segment is the first SGL Descriptor of the metadata for the command. If the SGL Descriptor contained in that SGL segment is an SGL Data Block descriptor, then that SGL Data Block Descriptor is the only SGL Descriptor and therefore describes the entire metadata data transfer. Refer to section 4.3.2. The controller is not required to check that bits 2:0 are cleared to 000b. The controller may report an error of Invalid Field in Command if bits 2:0 are not cleared to 000b. If the controller does not report an error of Invalid Field in Command, then the controller shall operate as if bits 2:0 are cleared to 000b.</p>

**Figure 92: Common Command Format**

Bytes	Description				
39:24	<p><b>Data Pointer (DPTR):</b> This field specifies the data used in the command. If CDW0.PSDT is cleared to 00b, then the definition of this field is:</p> <table border="1"> <tr> <td>39:32</td> <td> <p><b>PRP Entry 2 (PRP2):</b> This field:</p> <ul style="list-style-type: none"> <li>• is reserved if the data transfer does not cross a memory page boundary;</li> <li>• specifies the Page Base Address of the second memory page if the data transfer crosses exactly one memory page boundary. E.g.,:                             <ul style="list-style-type: none"> <li>○ the command data transfer length is equal in size to one memory page and the offset portion of the PBAO field of PRP1 is non-zero; or</li> <li>○ the Offset portion of the PBAO field of PRP1 is equal to 0h and the command data transfer length is greater than one memory page and less than or equal to two memory pages in size;</li> </ul> </li> </ul> <p>and</p> <ul style="list-style-type: none"> <li>• is a PRP List pointer if the data transfer crosses more than one memory page boundary. E.g.,:                             <ul style="list-style-type: none"> <li>○ the command data transfer length is greater than or equal to two memory pages in size but the offset portion of the PBAO field of PRP1 is non-zero; or</li> <li>○ the command data transfer length is equal in size to more than two memory pages and the Offset portion of the PBAO field of PRP1 is equal to 0h.</li> </ul> </li> </ul> </td> </tr> <tr> <td>31:24</td> <td> <p><b>PRP Entry 1 (PRP1):</b> This field contains:</p> <ul style="list-style-type: none"> <li>• the first PRP entry for the command; or</li> <li>• a PRP List pointer,</li> </ul> <p>depending on the command (e.g., the Create I/O Completion Queue command (refer to Figure 473)).</p> </td> </tr> </table>	39:32	<p><b>PRP Entry 2 (PRP2):</b> This field:</p> <ul style="list-style-type: none"> <li>• is reserved if the data transfer does not cross a memory page boundary;</li> <li>• specifies the Page Base Address of the second memory page if the data transfer crosses exactly one memory page boundary. E.g.,:                             <ul style="list-style-type: none"> <li>○ the command data transfer length is equal in size to one memory page and the offset portion of the PBAO field of PRP1 is non-zero; or</li> <li>○ the Offset portion of the PBAO field of PRP1 is equal to 0h and the command data transfer length is greater than one memory page and less than or equal to two memory pages in size;</li> </ul> </li> </ul> <p>and</p> <ul style="list-style-type: none"> <li>• is a PRP List pointer if the data transfer crosses more than one memory page boundary. E.g.,:                             <ul style="list-style-type: none"> <li>○ the command data transfer length is greater than or equal to two memory pages in size but the offset portion of the PBAO field of PRP1 is non-zero; or</li> <li>○ the command data transfer length is equal in size to more than two memory pages and the Offset portion of the PBAO field of PRP1 is equal to 0h.</li> </ul> </li> </ul>	31:24	<p><b>PRP Entry 1 (PRP1):</b> This field contains:</p> <ul style="list-style-type: none"> <li>• the first PRP entry for the command; or</li> <li>• a PRP List pointer,</li> </ul> <p>depending on the command (e.g., the Create I/O Completion Queue command (refer to Figure 473)).</p>
	39:32	<p><b>PRP Entry 2 (PRP2):</b> This field:</p> <ul style="list-style-type: none"> <li>• is reserved if the data transfer does not cross a memory page boundary;</li> <li>• specifies the Page Base Address of the second memory page if the data transfer crosses exactly one memory page boundary. E.g.,:                             <ul style="list-style-type: none"> <li>○ the command data transfer length is equal in size to one memory page and the offset portion of the PBAO field of PRP1 is non-zero; or</li> <li>○ the Offset portion of the PBAO field of PRP1 is equal to 0h and the command data transfer length is greater than one memory page and less than or equal to two memory pages in size;</li> </ul> </li> </ul> <p>and</p> <ul style="list-style-type: none"> <li>• is a PRP List pointer if the data transfer crosses more than one memory page boundary. E.g.,:                             <ul style="list-style-type: none"> <li>○ the command data transfer length is greater than or equal to two memory pages in size but the offset portion of the PBAO field of PRP1 is non-zero; or</li> <li>○ the command data transfer length is equal in size to more than two memory pages and the Offset portion of the PBAO field of PRP1 is equal to 0h.</li> </ul> </li> </ul>			
	31:24	<p><b>PRP Entry 1 (PRP1):</b> This field contains:</p> <ul style="list-style-type: none"> <li>• the first PRP entry for the command; or</li> <li>• a PRP List pointer,</li> </ul> <p>depending on the command (e.g., the Create I/O Completion Queue command (refer to Figure 473)).</p>			
<p>If CDW0.PSDT is set to 01b or 10b, then the definition of this field is:</p> <table border="1"> <tr> <td>39:24</td> <td> <p><b>SGL Entry 1 (SGL1):</b> This field contains the first SGL segment for the command. If the SGL segment is an SGL Data Block or Keyed SGL Data Block or Transport SGL Data Block descriptor, then it describes the entire data transfer. If more than one SGL segment is needed to describe the data transfer, then the first SGL segment is a Segment, or Last Segment descriptor. Refer to section 4.3.2 for the definition of SGL segments and descriptor types.</p> <p>The NVMe Transport may support a subset of SGL Descriptor types and features as defined in the NVMe Transport binding specification.</p> </td> </tr> </table>		39:24	<p><b>SGL Entry 1 (SGL1):</b> This field contains the first SGL segment for the command. If the SGL segment is an SGL Data Block or Keyed SGL Data Block or Transport SGL Data Block descriptor, then it describes the entire data transfer. If more than one SGL segment is needed to describe the data transfer, then the first SGL segment is a Segment, or Last Segment descriptor. Refer to section 4.3.2 for the definition of SGL segments and descriptor types.</p> <p>The NVMe Transport may support a subset of SGL Descriptor types and features as defined in the NVMe Transport binding specification.</p>		
39:24	<p><b>SGL Entry 1 (SGL1):</b> This field contains the first SGL segment for the command. If the SGL segment is an SGL Data Block or Keyed SGL Data Block or Transport SGL Data Block descriptor, then it describes the entire data transfer. If more than one SGL segment is needed to describe the data transfer, then the first SGL segment is a Segment, or Last Segment descriptor. Refer to section 4.3.2 for the definition of SGL segments and descriptor types.</p> <p>The NVMe Transport may support a subset of SGL Descriptor types and features as defined in the NVMe Transport binding specification.</p>				
43:40	<b>Command Dword 10 (CDW10):</b> This field is command specific Dword 10.				
47:44	<b>Command Dword 11 (CDW11):</b> This field is command specific Dword 11.				
51:48	<b>Command Dword 12 (CDW12):</b> This field is command specific Dword 12.				
55:52	<b>Command Dword 13 (CDW13):</b> This field is command specific Dword 13.				
59:56	<b>Command Dword 14 (CDW14):</b> This field is command specific Dword 14.				
63:60	<b>Command Dword 15 (CDW15):</b> This field is command specific Dword 15.				

In addition to the fields commonly defined for the Common Command Format, 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 Vendor Specific commands are defined in Figure 93. For more details, refer to section 8.1.26.

**Figure 93: Common Command Format – Vendor Specific Commands (Optional)**

Bytes	Description
03:00	<b>Command Dword 0 (CDW0):</b> This field is common to all commands and is defined in Figure 91.
07:04	<b>Namespace Identifier (NSID):</b> This field indicates the namespace ID that this command applies to. If the namespace ID is not used for the command, then this field shall be cleared to 0h. Setting this value to FFFFFFFFh causes the command to be applied to all namespaces attached to the controller processing the command, unless otherwise specified.  The behavior of a controller in response to an inactive namespace ID for a vendor specific command is vendor specific. Specifying an invalid namespace ID in a command that uses the namespace ID shall cause the controller to abort the command with a status code of Invalid Namespace or Format, unless otherwise specified.
15:08	Reserved
39:16	<b>Metadata and Data Pointers (MDPTR):</b> Refer to Figure 92 for the definition of these fields.
43:40	<b>Number of Dwords in Data Transfer (NDT):</b> This field indicates the number of dwords in the data transfer.
47:44	<b>Number of Dwords in Metadata Transfer (NDM):</b> This field indicates the number of dwords in the metadata transfer.
51:48	<b>Command Dword 12 (CDW12):</b> This field is command specific Dword 12.
55:52	<b>Command Dword 13 (CDW13):</b> This field is command specific Dword 13.
59:56	<b>Command Dword 14 (CDW14):</b> This field is command specific Dword 14.
63:60	<b>Command Dword 15 (CDW15):</b> This field is command specific Dword 15.

#### 4.1.2 Fabrics Command Common SQE

The common submission queue entry for Fabrics commands is shown in Figure 94.

**Figure 94: Fabrics Command – Submission Queue Entry Format**

Bytes	Description																
03:00	<b>Command Dword 0 (CDW0):</b> Refer to Figure 95.																
04	<p><b>Fabrics Command Type (FCTYPE):</b> This field specifies the Fabrics command transferred in the capsule. The Fabrics command types are defined in Figure 540. If this field is set to a reserved value, then the command shall be aborted with a status code of Invalid Field in Command. The format of the FCTYPE field is shown here:</p> <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>07:02</td> <td><b>Function (FN):</b> This field contains a value that, in combination with the other fields in the Fabrics Command Type data structure, creates a unique Combined Fabrics Command Type.</td> </tr> <tr> <td>01:00</td> <td> <p><b>Data Transfer Direction (DTD):</b> This field indicates the direction of a data transfer, if any. All options of the command shall transfer data as specified or transfer no data. All commands, including vendor specific commands, shall follow this convention.</p> <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td><b>No Data Transfer:</b> No data is transferred.</td> </tr> <tr> <td>01b</td> <td><b>Host to Controller Transfer:</b> Data is transferred from the host to the controller.</td> </tr> <tr> <td>10b</td> <td><b>Controller to Host Transfer:</b> Data is transferred from the controller to the host.</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table> </td> </tr> </tbody> </table>	Bits	Description	07:02	<b>Function (FN):</b> This field contains a value that, in combination with the other fields in the Fabrics Command Type data structure, creates a unique Combined Fabrics Command Type.	01:00	<p><b>Data Transfer Direction (DTD):</b> This field indicates the direction of a data transfer, if any. All options of the command shall transfer data as specified or transfer no data. All commands, including vendor specific commands, shall follow this convention.</p> <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td><b>No Data Transfer:</b> No data is transferred.</td> </tr> <tr> <td>01b</td> <td><b>Host to Controller Transfer:</b> Data is transferred from the host to the controller.</td> </tr> <tr> <td>10b</td> <td><b>Controller to Host Transfer:</b> Data is transferred from the controller to the host.</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	<b>No Data Transfer:</b> No data is transferred.	01b	<b>Host to Controller Transfer:</b> Data is transferred from the host to the controller.	10b	<b>Controller to Host Transfer:</b> Data is transferred from the controller to the host.	11b	Reserved
Bits	Description																
07:02	<b>Function (FN):</b> This field contains a value that, in combination with the other fields in the Fabrics Command Type data structure, creates a unique Combined Fabrics Command Type.																
01:00	<p><b>Data Transfer Direction (DTD):</b> This field indicates the direction of a data transfer, if any. All options of the command shall transfer data as specified or transfer no data. All commands, including vendor specific commands, shall follow this convention.</p> <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td><b>No Data Transfer:</b> No data is transferred.</td> </tr> <tr> <td>01b</td> <td><b>Host to Controller Transfer:</b> Data is transferred from the host to the controller.</td> </tr> <tr> <td>10b</td> <td><b>Controller to Host Transfer:</b> Data is transferred from the controller to the host.</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	<b>No Data Transfer:</b> No data is transferred.	01b	<b>Host to Controller Transfer:</b> Data is transferred from the host to the controller.	10b	<b>Controller to Host Transfer:</b> Data is transferred from the controller to the host.	11b	Reserved						
Value	Definition																
00b	<b>No Data Transfer:</b> No data is transferred.																
01b	<b>Host to Controller Transfer:</b> Data is transferred from the host to the controller.																
10b	<b>Controller to Host Transfer:</b> Data is transferred from the controller to the host.																
11b	Reserved																
23:05	Reserved																
39:24	<p><b>SGL Descriptor 1 (SGL1):</b> This field contains a Transport SGL Data Block descriptor or a Keyed SGL Data Block descriptor that describes the entire data transfer. Refer to section 4.3.2 for the definition of SGL descriptors.</p> <p>This field is used for Fabrics commands that transfer data. If a Fabrics command does not transfer data, then this field is reserved.</p>																

**Figure 94: Fabrics Command – Submission Queue Entry Format**

Bytes	Description
63:40	<b>Fabrics Command Type Specific (FCTS):</b> This field is Fabrics command type specific.

**Figure 95: Fabrics Command – Command Dword 0**

Bits	Description										
31:16	<b>Command Identifier (CID):</b> Refer to Figure 91.										
15:14	<b>PRP or SGL for Data Transfer (PSDT):</b>										
	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td><b>No Data Transferred:</b> This value is used for Fabrics commands that do not transfer data. If this value is used for Fabrics commands that transfer data, then SGLs are used for this transfer. A host should use the value 10b rather than 00b for Fabrics commands that transfer data.</td> </tr> <tr> <td>01b</td> <td>Reserved</td> </tr> <tr> <td>10b</td> <td><b>Data Transferred:</b> This value is used for Fabrics commands that transfer data. SGLs are used for this transfer.</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Description	00b	<b>No Data Transferred:</b> This value is used for Fabrics commands that do not transfer data. If this value is used for Fabrics commands that transfer data, then SGLs are used for this transfer. A host should use the value 10b rather than 00b for Fabrics commands that transfer data.	01b	Reserved	10b	<b>Data Transferred:</b> This value is used for Fabrics commands that transfer data. SGLs are used for this transfer.	11b	Reserved
	Value	Description									
	00b	<b>No Data Transferred:</b> This value is used for Fabrics commands that do not transfer data. If this value is used for Fabrics commands that transfer data, then SGLs are used for this transfer. A host should use the value 10b rather than 00b for Fabrics commands that transfer data.									
	01b	Reserved									
10b	<b>Data Transferred:</b> This value is used for Fabrics commands that transfer data. SGLs are used for this transfer.										
11b	Reserved										
13:10	Reserved										
09:08	<b>Fused Operation (FUSE):</b> Refer to Figure 91. There are no fused Fabrics commands and as a result this field is cleared to 00b.										
07:00	<b>Opcode (OPC):</b> This field is set to 7Fh to specify a Fabrics command.										

## 4.2 Completion Queue Entry

### 4.2.1 Admin Command and I/O Command Common CQE

The Common Completion Queue Entry Layout is at least 16 bytes in size. Figure 96 describes the layout of the first 16 bytes of the completion queue entry data structure which follows the Common Completion Queue Entry Layout. The contents of Dword 0 and Dword 1 are command specific. If a command uses Dword 0 or Dword 1, then the definition of these dwords is contained within the associated command definition. If a command does not use Dword 0 or Dword 1, then the unused field(s) are reserved. Dword 2 is defined in Figure 97 and Dword 3 is defined in Figure 98.

If a completion queue entry is constructed via multiple writes, the Phase Tag bit shall be updated in the last write of that completion queue entry.

**Figure 96: Common Completion Queue Entry Layout – Admin and All I/O Command Sets**

	31	23	15	7	0
<b>DW0</b>	Command Specific				
<b>DW1</b>	Command Specific				
<b>DW2</b>	SQ Identifier		SQ Head Pointer		
<b>DW3</b>	Status	P	Command Identifier		

**Figure 97: Completion Queue Entry: DW 2**

Bits	Description
31:16	<b>SQ Identifier (SQID):</b> Indicates the Submission Queue to which the associated command was issued. This field is used by host software when more than one Submission Queue shares a single Completion Queue to uniquely determine the command completed in combination with the Command Identifier (CID).  This is a reserved field in NVMe over Fabrics implementations.

**Figure 97: Completion Queue Entry: DW 2**

Bits	Description
15:00	<p><b>SQ Head Pointer (SQHD):</b> Indicates the current Submission Queue Head pointer for the Submission Queue indicated in the SQ Identifier field. This is used to indicate to the host the submission queue entries that have been consumed and may be re-used for new entries.</p> <p>Note: The value returned is the value of the SQ Head pointer when the completion queue entry was created. By the time host software consumes the completion queue entry, the controller may have an SQ Head pointer that has advanced beyond the value indicated.</p>

**Figure 98: Completion Queue Entry: DW 3**

Bits	Description
31:17	<b>Status (STATUS):</b> Indicates the status for the command that is being completed. Refer to section 4.2.3.
16	<p><b>Phase Tag (P):</b> Identifies whether a completion queue entry is new. Refer to section 4.2.4.</p> <p>This is a reserved bit in NVMe over Fabrics implementations.</p>
15:00	<b>Command Identifier (CID):</b> Indicates the identifier of the command that is being completed. This identifier is assigned by host software when the command is submitted to the Submission Queue. The combination of the SQ Identifier and Command Identifier uniquely identifies the command that is being completed. The maximum number of requests outstanding for a Submission Queue at one time is 65,535.

#### 4.2.2 Fabrics Command Common CQE

The common completion queue entry for Fabrics commands is shown in Figure 99.

**Figure 99: Fabrics Response – Completion Queue Entry Format**

Bytes	Description		
07:00	<b>Fabrics Response Type Specific (FRTS):</b> The definition of this field is Fabrics response type specific.		
09:08	<b>SQ Head Pointer (SQHD):</b> This field indicates the current Submission Queue Head pointer for the associated Submission Queue. This field is reserved if SQ flow control is disabled for the queue pair (refer to section 6.3).		
11:10	Reserved		
13:12	<b>Command Identifier (CID):</b> This field indicates the identifier of the command that is being completed.		
15:14	<b>Status Info (STS):</b> This field indicates the status for the associated Fabrics command.	<b>Bits</b>	<b>Description</b>
		15:01	<b>Status (STATUS):</b> This field as defined in section 4.2.3.
		00	Reserved

#### 4.2.3 Status Field Definition

The Status field defines the status for the command indicated in the completion queue entry, defined in Figure 100.

A value of 0h for the Status field indicates a successful command completion, with no fatal or non-fatal error conditions. Unless otherwise noted, if a command fails to complete successfully for multiple reasons, then the particular status code returned is chosen by the vendor.

**Figure 100: Completion Queue Entry: Status Field**

Bits	Description
31	<b>Do Not Retry (DNR):</b> If this bit is set to '1', then this bit indicates that if the same command is re-submitted to any controller in the NVM subsystem, then that re-submitted command is expected to fail. If this bit is cleared to '0', then this bit indicates that the same command may succeed if retried. If a command is aborted due to time limited error recovery (refer to the Error Recovery section in the NVM Command Set Specification), this bit should be cleared to '0'. If the SCT and SC fields are cleared to 0h, then this bit should be cleared to '0'.
30	<b>More (M):</b> If this bit is set to '1', then there is more status information for this command as part of the Error Information log page that may be retrieved with the Get Log Page command. If this bit is cleared to '0', then there is no additional status information for this command. Refer to section 5.1.12.1.2.
29:28	<p><b>Command Retry Delay (CRD):</b> If the DNR bit is cleared to '0' and the host has set the Advanced Command Retry Enable (ACRE) field to 1h in the Host Behavior Support feature (refer to section 5.1.25.1.14), then:</p> <ul style="list-style-type: none"> <li>a) a 00b CRD value indicates a command retry delay time of zero (i.e., the host may retry the command immediately); and</li> <li>b) a non-zero CRD value selects a field in the Identify Controller data structure (refer to Figure 312) that indicates the command retry delay time: <ul style="list-style-type: none"> <li>• a 01b CRD value selects the Command Retry Delay Time 1 (CRDT1) field;</li> <li>• a 10b CRD value selects the Command Retry Delay Time 2 (CRDT2) field; and</li> <li>• a 11b CRD value selects the Command Retry Delay Time 3 (CRDT3) field.</li> </ul> </li> </ul> <p>The host should not retry the command until at least the amount of time indicated by the selected field has elapsed. It is not an error for the host to retry the command prior to that time.</p> <p>If the DNR bit is set to '1' in the Status field or the ACRE field is cleared to 0h in the Host Behavior Support feature, then this field is reserved.</p> <p>If the SCT and SC fields are cleared to 0h, then this field should be cleared to 00b.</p>
27:25	<b>Status Code Type (SCT):</b> Indicates the status code type of the completion queue entry. This indicates the type of status code the controller is returning.
24:17	<b>Status Code (SC):</b> Indicates a status code identifying any error or status information for the command indicated.

Completion queue entries indicate a Status Code Type (SCT) for the type of completion being reported. Figure 101 specifies the status code type values and descriptions.

<b>Figure 101: Status Code – Status Code Type Values</b>		
Value	Description	Reference
0h	<b>Generic Command Status:</b> Indicates that the command specified by the Command and Submission Queue identifiers in the completion queue entry has completed. These status values are generic across all command types, and include such conditions as success, opcode not supported, and invalid field.	4.2.3.1
1h	<b>Command Specific Status:</b> Indicates a status value that is specific to a particular command opcode. These values may indicate additional processing is required. Status values such as invalid firmware image or exceeded maximum number of queues is reported with this type.	4.2.3.2
2h	<b>Media and Data Integrity Errors:</b> Any media specific errors that occur in the NVM or data integrity type errors shall be of this type.	4.2.3.3
3h	<p><b>Path Related Status:</b> Indicates that the command specified by the Command and Submission Queue identifier in the completion queue entry has completed. These status values are generic across all command types. These values may indicate that additional process is required and indicate a status value that is specific to:</p> <ul style="list-style-type: none"> <li>a) the connection between the host and the controller processing the command; or</li> <li>b) the characteristics that support Asymmetric Namespace Access Reporting (refer to section 8.1.1), the characteristics of the relationship between the controller processing the command and the specified namespace.</li> </ul>	4.2.3.4
4h to 6h	Reserved	

**Figure 101: Status Code – Status Code Type Values**

Value	Description	Reference
7h	Vendor Specific	

The Status Code (SC) field in the completion queue entry indicates more detailed status information about the completion being reported.

Each Status Code set of values is split into three ranges:

- 00h to 7Fh: Applicable to Admin Command Set, or across multiple command sets;
- 80h to BFh: I/O Command Set specific status codes; and
- C0h to FFh: Vendor Specific status codes.

Unless otherwise specified, if multiple status codes apply, then the controller selects the status code that is returned.

#### 4.2.3.1 Generic Command Status Definition

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

**Figure 102: Status Code – Generic Command Status Values**

Value	Description	I/O Command Set(s) <sup>1</sup>
00h	<b>Successful Completion:</b> The command completed without error.	
01h	<b>Invalid Command Opcode:</b> A reserved coded value or an unsupported value in the command opcode field.	
02h	<b>Invalid Field in Command:</b> A reserved coded value or an unsupported value in a defined field (other than the opcode field). This status code should be used unless another status code is explicitly specified for a particular condition. The field may be in the command parameters as part of the submission queue entry or in data structures pointed to by the command parameters.	
03h	<b>Command ID Conflict:</b> The command identifier is already in use. Note: It is implementation specific how many commands are searched for a conflict.	
04h	<b>Data Transfer Error:</b> Transferring the data or metadata associated with a command had an error.	
05h	<b>Commands Aborted due to Power Loss Notification:</b> Indicates that the command was aborted due to a power loss notification.	
06h	<b>Internal Error:</b> The command was not completed successfully due to an internal error. Details on the internal device error should be reported as an asynchronous event. Refer to Figure 149 for Internal Error Asynchronous Event Information.	
07h	<b>Command Abort Requested:</b> The command was aborted due to: <ul style="list-style-type: none"> <li>• an Abort command being received that specified this command (refer to section 5.1); or</li> <li>• a Cancel command being received that specified this command (refer to section 7.1).</li> </ul>	
08h	<b>Command Aborted due to SQ Deletion:</b> The command was aborted due to a Delete I/O Submission Queue request received for the Submission Queue to which the command was submitted.	
09h	<b>Command Aborted due to Failed Fused Command:</b> The command was aborted due to the other command in a fused operation failing.	
0Ah	<b>Command Aborted due to Missing Fused Command:</b> The fused command was aborted due to the adjacent submission queue entry not containing a fused command that is the other command in a supported fused operation (refer to section 3.4.2).	
0Bh	<b>Invalid Namespace or Format:</b> The namespace or the format of that namespace is invalid.	

Figure 102: Status Code – Generic Command Status Values

Value	Description	I/O Command Set(s) <sup>1</sup>
0Ch	<b>Command Sequence Error:</b> The command was aborted due to a protocol violation in a multi-command sequence (e.g., a violation of the Security Send and Security Receive sequencing rules in the TCG Storage Synchronous Interface Communications protocol (refer to TCG Storage Architecture Core Specification)).	
0Dh	<b>Invalid SGL Segment Descriptor:</b> The command includes an invalid SGL Last Segment or SGL Segment descriptor. This may occur under various conditions, including: <ul style="list-style-type: none"> <li>a) the SGL segment pointed to by an SGL Last Segment descriptor contains an SGL Segment descriptor or an SGL Last Segment descriptor;</li> <li>b) an SGL Last Segment descriptor contains an invalid length (i.e., a length of 0h or 1h that is not a multiple of 16); or</li> <li>c) an SGL Segment descriptor or an SGL Last Segment descriptor contains an invalid address (e.g., an address that is not qword aligned).</li> </ul>	
0Eh	<b>Invalid Number of SGL Descriptors:</b> There is an SGL Last Segment descriptor or an SGL Segment descriptor in a location other than the last descriptor of a segment based on the length indicated. This is also used for invalid SGLs in a command capsule.	
0Fh	<b>Data SGL Length Invalid:</b> This may occur if the length of a data SGL is too short. This may occur if the length of a data SGL is too long and the controller does not support SGL transfers longer than the amount of data to be transferred as indicated in the SGL Support field of the Identify Controller data structure.	
10h	<b>Metadata SGL Length Invalid:</b> This may occur if the length of a metadata SGL is too short. This may occur if the length of a metadata SGL is too long and the controller does not support SGL transfers longer than the amount of data to be transferred as indicated in the SGL Support field of the Identify Controller data structure.	
11h	<b>SGL Descriptor Type Invalid:</b> The type of an SGL Descriptor is a type that is not supported by the controller, or the combination of type and subtype is not supported by the controller.	
12h	<b>Invalid Use of Controller Memory Buffer:</b> The attempted use of the Controller Memory Buffer is not supported by the controller. Refer to section 8.2.1.	
13h	<b>PRP Offset Invalid:</b> The Offset field for a PRP entry is invalid. This may occur when there is a PRP entry with a non-zero offset after the first entry or when the Offset field in any PRP entry is not dword aligned (i.e., bits 1:0 are not cleared to 00b).	
14h	<b>Atomic Write Unit Exceeded:</b> See the applicable I/O Command Set specification for the description.	NVM, ZNS
15h	<b>Operation Denied:</b> The command was denied due to lack of access rights. Refer to the appropriate security specification (e.g., TCG Storage Interface Interactions specification). For media access commands, the Access Denied status code should be used instead.	
16h	<b>SGL Offset Invalid:</b> The offset specified in an SGL descriptor is invalid. This may occur when using capsules for data transfers in NVMe over Fabrics implementations and an invalid offset in the capsule is specified.	
17h	Reserved	
18h	<b>Host Identifier Inconsistent Format:</b> The NVM subsystem detected the simultaneous use of 64-bit and 128-bit Host Identifier values on different controllers.	
19h	<b>Keep Alive Timer Expired:</b> The Keep Alive Timer expired.	
1Ah	<b>Keep Alive Timeout Invalid:</b> The Keep Alive Timeout value specified is invalid. This may be due to an attempt to specify a value of 0h on a transport that requires the Keep Alive Timer feature to be enabled. This may be due to the value specified being too large for the associated NVMe Transport as defined in the NVMe Transport binding specification.	
1Bh	<b>Command Aborted due to Preempt and Abort:</b> The command was aborted due to a Reservation Acquire command with the Reservation Acquire Action (RACQA) set to 010b (Preempt and Abort).	
1Ch	<b>Sanitize Failed:</b> The most recent sanitize operation failed and no recovery action has been successfully completed.	



**Figure 102: Status Code – Generic Command Status Values**

Value	Description	I/O Command Set(s) <sup>1</sup>
1Dh	<b>Sanitize In Progress:</b> The requested function (e.g., command) is prohibited while a sanitize operation is in progress. Refer to section 8.1.24.3.	
1Eh	<b>SGL Data Block Granularity Invalid:</b> See the applicable I/O Command Set specification for the description.	NVM, ZNS
1Fh	<b>Command Not Supported for Queue in CMB:</b> The implementation does not support submission of the command to a Submission Queue in the Controller Memory Buffer or command completion to a Completion Queue in the Controller Memory Buffer. Note: NVM Express revision 1.3 and later use this status code only for Sanitize commands.	
20h	<b>Namespace is Write Protected:</b> The command is prohibited while the namespace is write protected as a result of a change in the namespace write protection state as defined by the Namespace Write Protection State Machine (refer to Figure 622).	
21h	<b>Command Interrupted:</b> Command processing was interrupted and the controller is unable to successfully complete the command. The host should retry the command.  If this status code is returned, then the controller shall clear the Do Not Retry bit to '0' in the Status field of the CQE (refer to Figure 100). The controller shall not return this status code unless the host has set the Advanced Command Retry Enable (ACRE) field to 1h in the Host Behavior Support feature (refer to section 5.1.25.1.14).	
22h	<b>Transient Transport Error:</b> A transient transport error was detected. If the command is retried on the same controller, the command is likely to succeed. A command that fails with a transient transport error four or more times should be treated as a persistent transport error that is not likely to succeed if retried on the same controller.	
23h	<b>Command Prohibited by Command and Feature Lockdown:</b> The command was aborted due to command execution being prohibited by the Command and Feature Lockdown (refer to section 8.1.5).	
24h	<b>Admin Command Media Not Ready:</b> The Admin command requires access to media and the media is not ready. The Do Not Retry bit indicates whether re-issuing the command at a later time may succeed. This status code shall only be returned: <ul style="list-style-type: none"> <li>a) for Admin commands; and</li> <li>b) if the controller is in Controller Ready Independent of Media mode (i.e., CC.CRIME bit is set to '1').</li> </ul> <p>This status code shall not be returned with the Do Not Retry bit cleared to '0' after the amount of time indicated by the Controller Ready With Media Timeout (CRTO.CRWMT) field after the controller is enabled (i.e., CC.EN transitions from '0' to '1').</p> <p>Refer to Figure 84 for the list of Admin commands permitted to return this status code.</p>	
25h	<b>Invalid Key Tag:</b> The command was aborted due to an invalid KEYTAG field value (refer to Figure 620) as: <ul style="list-style-type: none"> <li>a) the value of the specified KEYTAG field is greater than the Maximum Key Tag (MAXKT) field in the I/O Command Set Independent Identify Namespace data structure (refer to Figure 319); or</li> <li>b) defined by the appropriate security specification (e.g., TCG Storage Interface Interactions specification).</li> </ul>	
26h	<b>Host Dispersed Namespace Support Not Enabled:</b> The command is prohibited while the Host Dispersed Namespace Support (HDISNS) field is not set to 1h in the Host Behavior Support feature (refer to Figure 407).	
27h	<b>Host Identifier Not Initialized</b>	
28h	<b>Incorrect Key:</b> The command was aborted due to the key associated with the KEYTAG field being incorrect.  The specific conditions under which a key is considered incorrect are defined by the appropriate security specification (e.g., TCG Storage Interface Interactions specification).	

Figure 102: Status Code – Generic Command Status Values

Value	Description	I/O Command Set(s) <sup>1</sup>
29h	<b>FDP Disabled:</b> The command is not allowed when Flexible Data Placement is disabled.	
2Ah	<b>Invalid Placement Handle List:</b> The Placement Handle List is invalid due to: <ul style="list-style-type: none"> <li>• a Reclaim Unit Handle Identifier that is: <ul style="list-style-type: none"> <li>○ valid but restricted to be used by the command; or</li> <li>○ invalid;</li> </ul> </li> <li>or</li> <li>• the Placement Handle List number of entries exceeded the maximum number allowed.</li> </ul>	
2Bh to 7Fh	Reserved	
80h	<b>LBA Out of Range:</b> See the applicable I/O Command Set specification for the description.	NVM, ZNS
81h	<b>Capacity Exceeded:</b> The command attempted an operation that exceeds the capacity of the namespace.	
82h	<b>Namespace Not Ready:</b> The namespace is not ready to be accessed as a result of a condition other than a condition that is reported as an Asymmetric Namespace Access condition. The Do Not Retry bit indicates whether re-issuing the command at a later time may succeed.	
83h	<b>Reservation Conflict:</b> The command was aborted due to a conflict with a reservation held on the accessed namespace. Refer to section 8.1.22.	
84h	<b>Format In Progress:</b> A Format NVM command is in progress on the namespace. The Do Not Retry bit shall be cleared to '0' to indicate that the command may succeed if resubmitted.	NVM, ZNS
85h	<b>Invalid Value Size:</b> See the applicable I/O Command Set specification for the description.	KV
86h	<b>Invalid Key Size:</b> See the applicable I/O Command Set specification for the description.	KV
87h	<b>KV Key Does Not Exist:</b> See the applicable I/O Command Set specification for the description.	KV
88h	<b>Unrecovered Error:</b> See the applicable I/O Command Set specification for the description.	KV
89h	<b>Key Exists:</b> See the applicable I/O Command Set specification for the description.	KV
90h to BFh	Reserved	
C0h to FFh	Vendor Specific	
Key: NVM – NVM Command Set ZNS – Zoned Namespace Command Set KV – Key Value Command Set  Notes: 1. This column is blank unless the value is I/O Command Set specific		

#### 4.2.3.2 Command Specific Status Definition

Completion queue entries with a Status Code Type (SCT) of Command Specific Errors indicate an error that is specific to a particular command opcode. Status codes of 00h to 7Fh are for Admin command errors. Status codes of 80h to BFh are specific to the selected I/O command sets.

Figure 103: Status Code – Command Specific Status Values

Value	Description	Commands Affected
00h	Completion Queue Invalid	Create I/O Submission Queue

**Figure 103: Status Code – Command Specific Status Values**

Value	Description	Commands Affected
01h	Invalid Queue Identifier	Create I/O Submission Queue, Create I/O Completion Queue, Delete I/O Completion Queue, Delete I/O Submission Queue
02h	Invalid Queue Size	Create I/O Submission Queue, Create I/O Completion Queue
03h	Abort Command Limit Exceeded	Abort
04h	Reserved	
05h	Asynchronous Event Request Limit Exceeded	Asynchronous Event Request
06h	Invalid Firmware Slot	Firmware Commit
07h	Invalid Firmware Image	Firmware Commit
08h	Invalid Interrupt Vector	Create I/O Completion Queue
09h	Invalid Log Page	Get Log Page
0Ah	Invalid Format	Format NVM, Namespace Management
0Bh	Firmware Activation Requires Conventional Reset	Firmware Commit, Sanitize
0Ch	Invalid Queue Deletion	Delete I/O Completion Queue
0Dh	Feature Identifier Not Saveable	Set Features
0Eh	Feature Not Changeable	Set Features
0Fh	Feature Not Namespace Specific	Set Features
10h	Firmware Activation Requires NVM Subsystem Reset	Firmware Commit, Sanitize
11h	Firmware Activation Requires Controller Level Reset	Firmware Commit, Sanitize
12h	Firmware Activation Requires Maximum Time Violation	Firmware Commit
13h	Firmware Activation Prohibited	Firmware Commit
14h	Overlapping Range	Firmware Commit, Firmware Image Download, Set Features
15h	Namespace Insufficient Capacity	Namespace Management
16h	Namespace Identifier Unavailable	Namespace Management
17h	Reserved	
18h	Namespace Already Attached	Namespace Attachment
19h	Namespace Is Private	Namespace Attachment
1Ah	Namespace Not Attached	Namespace Attachment
1Bh	Thin Provisioning Not Supported	Namespace Management
1Ch	Controller List Invalid	Namespace Attachment
1Dh	Device Self-test In Progress	Device Self-test
1Eh	Boot Partition Write Prohibited	Firmware Commit
1Fh	Invalid Controller Identifier	Virtualization Management, Track Send, Track Receive, Migration Send, Migration Receive, Controller Data Queue
20h	Invalid Secondary Controller State	Virtualization Management
21h	Invalid Number of Controller Resources	Virtualization Management
22h	Invalid Resource Identifier	Virtualization Management
23h	Sanitize Prohibited While Persistent Memory Region is Enabled	Sanitize
24h	ANA Group Identifier Invalid	Namespace Management
25h	ANA Attach Failed	Namespace Attachment
26h	Insufficient Capacity	Capacity Management
27h	Namespace Attachment Limit Exceeded	Namespace Attachment
28h	Prohibition of Command Execution Not Supported	Lockdown
29h	I/O Command Set Not Supported	Namespace Attachment, Namespace Management
2Ah	I/O Command Set Not Enabled	Namespace Attachment
2Bh	I/O Command Set Combination Rejected	Set Features
2Ch	Invalid I/O Command Set	Identify
2Dh	Identifier Unavailable	Capacity Management

**Figure 103: Status Code – Command Specific Status Values**

Value	Description	Commands Affected
2Eh	Namespace Is Dispersed	Reservation Acquire, Reservation Register, Reservation Release, Reservation Report
2Fh	Invalid Discovery Information	Discovery Information Management
30h	Zoning Data Structure Locked	Fabric Zoning Lookup, Fabric Zoning Send, Fabric Zoning Receive
31h	Zoning Data Structure Not Found	Fabric Zoning Lookup, Fabric Zoning Send, Fabric Zoning Receive
32h	Insufficient Discovery Resources	Discovery Information Management
33h	Requested Function Disabled	Fabric Zoning Lookup, Fabric Zoning Send, Fabric Zoning Receive
34h	ZoneGroup Originator Invalid	Fabric Zoning Send
35h	Invalid Host	Manage Exported NVM Subsystem
36h	Invalid NVM Subsystem	Manage Exported NVM Subsystem
37h	Invalid Controller Data Queue	Set Features, Get Features, Track Send, Controller Data Queue
38h	Not Enough Resources	Track Send, Controller Data Queue
39h	Controller Suspended	Track Send, Sanitize
3Ah	Controller Not Suspended	Track Send
3Bh	Controller Data Queue Full	Track Send
3Ch to 6Fh	Reserved	
70h to 7Fh	Directive Specific	NOTE 1
80h to BFh	I/O Command Set Specific	Refer to Figure 104
C0h to FFh	Vendor Specific	
Notes:		
1. The Directives Specific range defines Directives specific status values. Refer to section 8.1.8.		

**Figure 104: Status Code – Command Specific Status Values, I/O Commands**

Value	Description
80h	Conflicting Attributes
81h	Invalid Protection Information
82h	Attempted Write to Read Only Range
83h	Command Size Limit Exceeded
84h	Invalid Command ID
85h	Incompatible Namespace or Format
86h	Fast Copy Not Possible
87h	Overlapping I/O Range
88h	Namespace Not Reachable
89h	Insufficient Resources
8Ah	Insufficient Program Resources
8Bh	Invalid Memory Namespace
8Ch	Invalid Memory Range Set
8Dh	Invalid Memory Range Set Identifier
8Eh	Invalid Program Data
8Fh	Invalid Program Index
90h	Invalid Program Type
91h	Maximum Memory Ranges Exceeded
92h	Maximum Memory Range Sets Exceeded
93h	Maximum Programs Activated
94h	Maximum Program Bytes Exceeded
95h	Memory Range Set In Use
96h	No Program
97h	Overlapping Memory Ranges

**Figure 104: Status Code – Command Specific Status Values, I/O Commands**

Value	Description
98h	Program Not Activated
99h	Program In Use
9Ah	Program Index Not Downloadable
9Bh	Program Too Big
9Ch	Successful Media Verification Read
9Dh to B7h	Reserved
B8h	Zoned Boundary Error
B9h	Zone Is Full
BAh	Zone Is Read Only
BBh	Zone Is Offline
BCh	Zone Invalid Write
BDh	Too Many Active Zones
BEh	Too Many Open Zones
BFh	Invalid Zone State Transition

**Figure 105: Status Code – Command Specific Status Values, Fabrics Commands**

Value	Description	Commands Affected
80h	<b>Incompatible Format:</b> The NVM subsystem does not support the record format specified by the host.	Connect, Disconnect
81h	<b>Controller Busy:</b> The controller is already associated with a host (Connect command). This value is also returned if there is no available controller (Connect command).  The controller is not able to disconnect the I/O Queue at the current time (Disconnect command).	Connect, Disconnect
82h	<b>Connect Invalid Parameters:</b> One or more of the command parameters (e.g., Host NQN, NVM Subsystem NQN, Host Identifier, Controller ID, Queue ID) specified are not valid.	Connect
83h	<b>Connect Restart Discovery:</b> The NVM subsystem requested is not available. The host should restart the discovery process.	Connect
84h	<b>Connect Invalid Host:</b> The host is not allowed to establish an association to any controller in the NVM subsystem or the host is not allowed to establish an association to the specified controller.	Connect
85h	<b>Invalid Queue Type:</b> The command was sent on the wrong queue type (e.g., a Disconnect command was sent on the Admin queue).	Disconnect
86h to 8Fh	Reserved	
90h	<b>Discover Restart:</b> The snapshot of the records is now invalid or out of date. If the Discovery log page was requested, then the host or Discovery controller should re-read the Discovery log page. If the Host Discovery log page was requested, then the host or Discovery controller should re-read the Host Discovery log page.	Get Log Page
91h	<b>Authentication Required:</b> NVMe in-band authentication is required and the queue has not yet been authenticated.	NOTE 1
92h to AFh	Reserved	
B0h to BFh	<b>Transport Specific:</b> The status values in this range are NVMe Transport specific. Refer to the appropriate NVMe Transport binding specification for the definition of these status values.	
Notes:		
1. All commands other than Connect, Authenticate Send, and Authenticate Receive.		

#### 4.2.3.3 Media and Data Integrity Errors Definition

Completion queue entries with a Status Code Type (SCT) of Media and Data Integrity Errors indicate an error associated with the command that is due to an error associated with the NVM media or a data integrity type error.

**Figure 106: Status Code – Media and Data Integrity Error Values**

Value	Description	Command Set(s) <sup>1</sup>
00h to 7Fh	Reserved	
80h	<b>Write Fault:</b> The write data could not be committed to the media.	
81h	<b>Unrecovered Read Error:</b> The read data could not be recovered from the media.	
82h	<b>End-to-end Guard Check Error:</b> The command was aborted due to an end-to-end guard check failure.	
83h	<b>End-to-end Application Tag Check Error:</b> The command was aborted due to an end-to-end application tag check failure.	
84h	<b>End-to-end Reference Tag Check Error:</b> The command was aborted due to an end-to-end reference tag check failure.	
85h	<b>Compare Failure:</b> See the NVM Command Set Specification for the description.	NVM
86h	<b>Access Denied:</b> Access to the namespace and/or user data is denied due to lack of access rights. Refer to the appropriate security specification (e.g., TCG Storage Interface Interactions Specification).	
87h	<b>Deallocated or Unwritten Logical Block:</b> See the NVM Command Set Specification for the description.	NVM
88h	<b>End-to-End Storage Tag Check Error:</b> The command was aborted due to an end-to-end storage tag check failure.	
89h to BFh	Reserved	
C0h to FFh	Vendor Specific	
Key: NVM – NVM Command Set		
Notes: 1. This column is blank unless the value is I/O Command Set specific.		

#### 4.2.3.4 Path Related Status Definition

Completion queue entries with a Status Code Type (SCT) of Path Related Status (refer to Figure 107) indicate a status value associated with the command that is generic across many different types of commands and applies to a specific connection between the host and controller processing the command or between the controller and the namespace. The command for which this status is returned may be retried on a different controller in the same NVM subsystem if more than one controller is available to the host.

In a multipath environment, unless otherwise specified, errors of this type should be retried using a different path, if one is available.

**Figure 107: Status Code – Path Related Status Values**

Value	Description
00h	<b>Internal Path Error:</b> The command was not completed as the result of a controller internal error that is specific to the controller processing the command. Retries for the request function should be based on the setting of the DNR bit (refer to Figure 100).
01h	<b>Asymmetric Access Persistent Loss:</b> The requested function (e.g., command) is not able to be performed as a result of the relationship between the controller and the namespace, NVM Set, or Endurance Group being in the ANA Persistent Loss state (refer to section 8.1.1.7). The command should not be re-submitted to the same controller.
02h	<b>Asymmetric Access Inaccessible:</b> The requested function (e.g., command) is not able to be performed as a result of the relationship between the controller and the namespace, NVM Set, or Endurance Group being in the ANA Inaccessible state (refer to section 8.1.1.6). The command should not be re-submitted to the same controller.
03h	<b>Asymmetric Access Transition:</b> The requested function (e.g., command) is not able to be performed as a result of the relationship between the controller and the namespace, NVM Set, or Endurance Group transitioning between Asymmetric Namespace Access states (refer to section 8.1.1.8). The requested function should be retried after the transition is complete.

**Figure 107: Status Code – Path Related Status Values**

Value	Description
04h to 5Fh	Reserved
<b>Controller detected Pathing errors</b>	
60h	<b>Controller Pathing Error:</b> A pathing error was detected by the controller.
61h to 6Fh	Reserved
<b>Host detected Pathing errors</b>	
70h	<b>Host Pathing Error:</b> A pathing error was detected by the host.
71h	<b>Command Aborted By Host:</b> The command was aborted as a result of host action (e.g., the host disconnected the fabric connection).
72h to 7Fh	Reserved
<b>Other Pathing errors</b>	
80h to BFh	I/O Command Set Specific
C0h to FFh	Vendor Specific

#### 4.2.4 Phase Tag

The Phase Tag bit indicates whether a completion queue entry is new. The Phase Tag bit for each completion queue entry in:

- the Admin Completion Queue shall be initialized to '0' by the host prior to setting CC.EN (refer to Figure 41) to '1'; and
- an I/O Completion Queue shall be initialized to '0' by the host prior to submitting the Create I/O Completion Queue command for that queue.

When the controller posts a new completion queue entry to the Completion Queue, the controller shall invert the Phase Tag bit in that completion queue entry (i.e., the inverting of the Phase Tag bit enables the host to detect the new completion queue entry).

When a completion queue entry is posted to a completion queue slot in:

- the Admin Queue for the first time after CC.EN is set to '1', the Phase Tag bit for that completion queue entry is set to '1'; and
- an I/O Completion Queue for the first time after the Create I/O Completion Queue command completed for that queue, the Phase Tag bit for that completion queue entry is set to '1'.

This continues for each completion queue entry that is posted until the controller posts a completion queue entry to the highest numbered completion queue slot and wraps to completion queue slot number 0 as described in section 3.3.1.2. When that queue wrap condition occurs, the Phase Tag bit is then cleared to '0' in each completion queue entry that is posted. This continues until another queue wrap condition occurs. Each time a queue wrap condition occurs, the value of the Phase Tag bit is inverted (i.e., changes from '1' to '0' or changes from '0' to '1').

##### 4.2.4.1 Phase Tag Example

Figure 108 shows an example of how the Phase Tag bit changes over time as a memory-based controller completes commands and the host processes those completions. This example shows a Completion Queue consisting of 6 entries.

Figure 108: Phase Tag bit Transition Example

T <sup>1</sup>	Condition	Completion Queue Entry/Slot number					
		0	1	2	3	4	5
0	Admin Queue: Host initializes Completion Queue and sets CC.EN to '1' I/O Queue: Host initializes Completion Queue and submits Create I/O Completion Queue command	P(0) (E) HEAD-> TAIL->	P(0) (E)	P(0) (E)	P(0) (E)	P(0) (E)	P(0) (E)
1	Controller has completed 1 command and the host has consumed 0 completions	P(1) HEAD->	P(0) (E) TAIL->	P(0) (E)	P(0) (E)	P(0) (E)	P(0) (E)
2	Controller has completed 6 commands and the host has consumed 2 completions	P(1) (E) TAIL->	P(1) (E)	P(1) HEAD->	P(1)	P(1)	P(1)
3	Controller has completed 7 commands and the host has consumed 2 completions	P(0)	P(1) (E) TAIL->	P(1) HEAD->	P(1)	P(1)	P(1)
4	Controller has completed 7 commands and the host has consumed 4 completions	P(0)	P(1) (E) TAIL->	P(1) (E)	P(1) (E)	P(1) HEAD->	P(1)
5	Controller has completed 11 commands and the host has consumed 8 completions	P(0) (E)	P(0) (E)	P(0) HEAD->	P(0)	P(0)	P(1) (E) TAIL->
6	Controller has completed 11 commands and the host has consumed 11 completions	P(0) (E)	P(0) (E)	P(0) (E)	P(0) (E)	P(0) (E)	P(1) (E) HEAD-> TAIL->

Key:  
P(0) = Phase Tag bit for this completion queue entry is cleared to the value '0'.  
P(1) = Phase Tag bit for this completion queue entry is set to the value '1'.  
(E) = The Entry/Slot is empty.  
HEAD-> = Completion Queue Head Pointer for this completion queue is set to indicate this slot.  
TAIL-> = Completion Queue Tail Pointer for this completion queue is used within the controller to indicate this slot.

Note:  
1. T = Time sequence.

At time 0, the host initializes the Completion queue (i.e., clearing the Phase Tag bit to '0' in each completion queue entry in the completion queue). For the Admin Completion Queue, the host then sets CC.EN to '1' to enable the controller. For an I/O Completion Queue, the host then sends the Create I/O Completion Queue command. The queue, at this time, is in the Empty condition (refer to section 3.3.1.4).

At time 1, the controller has completed a command, but the host has not consumed that completion queue entry. As a result of the command completion, the Phase Tag bit in completion queue entry 0 has been inverted to '1'. Since no completion queue entries have been consumed, the Completion Queue Head pointer still indicates completion queue entry 0. The controller has updated the internal Completion Queue Tail Pointer to indicate that completion queue slot 1 is the next completion queue slot into which the controller posts a completion queue entry.

At time 2, the controller has completed 5 additional commands (i.e., 6 commands have been completed) and the host has consumed 2 of the completion queue entries. As a result of the 5 additional commands having been completed, the Phase Tag bit has been inverted to '1' in completion queue entry 1 through completion queue entry 5. As a result of 2 completion queue entries having been consumed, the host has updated the Completion Queue Head Pointer to indicate that completion queue entry 0 and completion queue entry 1 have been consumed (i.e., completion queue entry 2 is the next completion queue entry for the host to consume). The controller has updated the internal Completion Queue Tail Pointer to indicate



that completion queue slot 0 is the next completion queue slot into which the controller posts a completion queue entry.

At time 3, the controller has completed 1 additional command (i.e., 7 commands have been completed) and no additional completion queue entries have been consumed by the host (i.e., 2 completion queue entries have been consumed). As a result of the additional command having been completed, the Phase Tag bit has been inverted to '0' in completion queue entry 0 (i.e., accounting for the queue wrap condition). The controller has updated the internal Completion Queue Tail Pointer to indicate that completion queue slot 1 is the next completion queue slot into which the controller posts a completion queue entry. The queue, at this time, is in the Full condition (refer to section 3.3.1.5).

At time 4, the controller has completed no additional commands (i.e., 7 commands have been completed) and the host has consumed 2 additional completion queue entries (i.e., 4 completion queue entries have been consumed). As a result of 2 additional completion queue entries having been consumed, the host has updated the Completion Queue Head Pointer to indicate that completion queue entry 2 and completion queue entry 3 have now been consumed (i.e., completion queue entry 4 is the next completion queue entry for the host to consume). The controller internal Completion Queue Tail Pointer has not changed.

At time 5, the controller has completed 11 commands and the host has consumed 8 of the completion queue entries. As a result of the 4 additional commands having been completed, the Phase Tag bit has been inverted to '0' in completion queue entry 1 through completion queue entry 4. As a result of the 4 additional completion queue entries having been consumed, the host has updated the Completion Queue Head Pointer to indicate that completion queue entry 5 through completion queue entry 1 (i.e., accounting for the queue wrap condition) have now been consumed (i.e., completion queue entry 2 is the next completion queue entry for the host to consume). The controller has updated the internal Completion Queue Tail Pointer to indicate that completion queue slot 5 is the next completion queue slot into which the controller posts a completion queue entry.

At time 6, the controller has completed 11 commands and the host has consumed all 11 of the completion queue entries. As a result of no new command completions, there are no changes to the Phase Tag bit values. As a result of the 3 additional completion queue entries having been consumed, the host has updated the Completion Queue Head Pointer to indicate that completion queue entry 2 through completion queue entry 4 have now been consumed (i.e., completion queue slot 5 is the next completion queue slot from which the host consumes a completion queue entry). The queue, at this time, is in the Empty condition (refer to section 3.3.1.4).

### 4.3 Data Pointer Layouts (PRPs and SGLs)

This section describes the data structures used to describe the layout of data that can be understood by the controller and the host.

#### 4.3.1 Physical Region Page Entry and List

A physical region page (PRP) entry is a pointer to a physical memory page. PRPs are used as a scatter/gather mechanism for data transfers between the controller and memory. To enable efficient out of order data transfers between the controller and the host, PRP entries are a fixed size.

The size of the physical memory page is configured by host software in CC.MPS. Figure 109 shows the layout of a PRP entry that consists of a Page Base Address and an Offset. The size of the Offset field is determined by the physical memory page size configured in CC.MPS.

**Figure 109: PRP Entry Layout**



The definition of a PRP entry is described in Figure 110.

**Figure 110: PRP Entry – Page Base Address and Offset**

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

A physical region page list (PRP List) is a set of PRP entries in a single page of contiguous memory. A PRP List describes additional PRP entries that could not be described within the command itself. Any PRP entries described within the command are not duplicated in a PRP List. If the amount of data to transfer requires multiple PRP List memory pages, then the last PRP entry before the end of the memory page shall be a pointer to the next PRP List, indicating the next segment of the PRP List. Figure 111 shows the layout of a PRP List where each PRP entry identifies memory pages that are physically contiguous. Figure 112 shows the layout of a PRP List where each PRP entry identifies a different memory page (i.e., the memory pages are not physically contiguous).

**Figure 111: PRP List Layout for Physically Contiguous Memory Pages**

63	$n+1$	$n$	0
Page Base Address $p$		0h	
Page Base Address $p+1$		0h	
...			
Page Base Address $p+q$		0h	
Page Base Address $p+q+1$		0h	

**Figure 112: PRP List Layout for Physically Non-Contiguous Memory Pages**

63	$n+1$	$n$	0
Page Base Address $p$		0h	
Page Base Address $q$		0h	
...			
Page Base Address $r$		0h	
Page Base Address $s$		0h	

Dependent on the command definition, the first PRP entry contained within the command may have a non-zero offset within the memory page. The first PRP List entry (i.e., the first pointer to a memory page containing additional PRP entries) that if present is typically contained in the PRP Entry 2 location within the command, shall be qword aligned and may also have a non-zero offset within the memory page.

PRP entries contained within a PRP List shall have a memory page offset of 0h. If a second PRP entry is present within a command, it shall have a memory page offset of 0h. In both cases, the entries are memory page aligned based on the value in CC.MPS. If the controller receives a non-zero offset for these PRP entries the controller should return an error of PRP Offset Invalid.

PRP Lists shall be minimally sized with packed entries starting with entry 0. If more PRP List pages are required, then the last entry of the PRP List contains the Page Base Address of the next PRP List page. The next PRP List page shall be memory page aligned. The total number of PRP entries required by a command is implied by the command parameters and memory page size.

### 4.3.2 Scatter Gather List (SGL)

A Scatter Gather List (SGL) is a data structure in memory address space used to describe a data buffer. The controller indicates the SGL types that the controller supports in the Identify Controller data structure. A data buffer is either a source buffer or a destination buffer. An SGL contains one or more SGL segments. The total length of the Data Block and Bit Bucket descriptors in an SGL shall be equal to or exceed the amount of data requested to be transferred. If the amount of data requested to be transferred exceeds the total length of the Data Block and Bit Bucket descriptors in an SGL, data shall not be transferred to or from locations that are not described by the SGL.

An SGL segment is a qword aligned data structure in a contiguous region of physical memory describing all, part of, or none of a data buffer and the next SGL segment, if any. An SGL segment consists of an array of one or more SGL descriptors. Only the last descriptor in an SGL segment may be an SGL Segment descriptor or an SGL Last Segment descriptor.

A last SGL segment is an SGL segment that does not contain an SGL Segment descriptor, or an SGL Last Segment descriptor.

A controller may support byte or dword alignment and granularity of Data Blocks. If a controller supports only dword alignment and granularity as indicated in the SGL Support field of the Identify Controller data structure (refer to Figure 312), then the values in the Address and Length fields of all Data Block descriptors shall have their two least significant bits cleared to 00b. This requirement applies to Data Block descriptors that indicate data and/or metadata memory regions.

The SGL Descriptor Threshold (SDT) field in the Identify Controller data structure (refer to Figure 312) indicates the recommended maximum number of SGL descriptors for a command. If the SDT field is set to a non-zero value, and a command is submitted for which the sum of:

- a) the number of SGL Bit Bucket descriptors with non-zero Length field contents; and
- b) the number of SGL Data Block descriptors with a non-zero Length field contents,

exceeds the value of the SDT field, then the performance of the controller may be reduced.

The value of the SDT field shall be less than or equal to the value of the Maximum SGL Data Block Descriptors (MSDBD) field in the Identify Controller data structure (refer to Figure 312 for the definition of the MSDBD field).

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

A Transport SGL Data Block descriptor is a Data Block descriptor that specifies a data block that is transferred by the NVMe Transport using a transfer mechanism and data buffers that are specific to the NVMe Transport.

The SGL Identifier Descriptor Sub Type field may indicate additional information about a descriptor. As an example, the Sub Type may indicate that the Address field is an offset rather than an absolute address. The Sub Type may also indicate NVMe Transport specific information.

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.

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 312.

Figure 113 defines the SGL segment.

**Figure 113: SGL Segment**

Bytes	Description
<b>SGL Descriptor List</b>	
15:00	<b>SGL Descriptor 0:</b> This field is the first SGL descriptor.
31:16	<b>SGL Descriptor 1:</b> This field is the second SGL descriptor.
...	...
((n*16)+15):(n*16)	<b>SGL Descriptor n:</b> This field is the last SGL descriptor.

An SGL segment contains one or more SGL descriptors. Figure 114 defines the generic SGL descriptor format.

**Figure 114: Generic SGL Descriptor Format**

Bytes	Description						
14:00	<b>Descriptor Type Specific (DTS):</b> This field is descriptor type specific.						
15	<b>SGL Identifier (SGLID):</b> The definition of this field is described in the table below.						
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>07:04</td> <td><b>SGL Descriptor Type (SGLDT):</b> Refer to Figure 115.</td> </tr> <tr> <td>03:00</td> <td><b>SGL Descriptor Sub Type (SGLDST):</b> Refer to Figure 116.</td> </tr> </tbody> </table>	Bits	Description	07:04	<b>SGL Descriptor Type (SGLDT):</b> Refer to Figure 115.	03:00	<b>SGL Descriptor Sub Type (SGLDST):</b> Refer to Figure 116.
	Bits	Description					
07:04	<b>SGL Descriptor Type (SGLDT):</b> Refer to Figure 115.						
03:00	<b>SGL Descriptor Sub Type (SGLDST):</b> Refer to Figure 116.						

The SGL Descriptor Type field defined in Figure 115 specifies the SGL descriptor type. If the SGL Descriptor Type field is set to a reserved value or an unsupported value, then the SGL descriptor shall be processed as having an SGL Descriptor Type error. If the SGL Descriptor Sub Type field is set to a reserved value or an unsupported value, then the descriptor shall be processed as having an SGL Descriptor Type error.

An SGL descriptor set to all zeroes is an SGL Data Block descriptor with the Address field cleared to 0h and the Length field cleared to 0h may be used as a NULL descriptor.

**Figure 115: SGL Descriptor Type**

Value	Descriptor
0h	SGL Data Block descriptor
1h	SGL Bit Bucket descriptor
2h	SGL Segment descriptor
3h	SGL Last Segment descriptor
4h	Keyed SGL Data Block descriptor
5h	Transport SGL Data Block descriptor
6h to Eh	Reserved
Fh	Vendor specific

Figure 116 defines the SGL Descriptor Sub Type values and indicates the SGL Descriptor Types to which each SGL Descriptor Sub Type applies.

**Figure 116: SGL Descriptor Sub Type Values**

SGL Descriptor Sub Type	SGL Descriptor Types	Sub Type Description
0h	0h, 2h, 3h, 4h	<b>Address:</b> The Address field specifies the starting 64-bit memory byte address of the Data Block, Segment, or Last Segment descriptor.
	1h	For Type 1h, the Sub Type field shall be cleared to 0h.
	All other values	Reserved
1h	0h, 2h, 3h	<b>Offset:</b> The Address field contains an offset from the beginning of the location where data may be transferred. For NVMe over PCIe implementations, this Sub Type is reserved. For NVMe over Fabrics implementations, refer to section 3.3.2.1.3.1.

**Figure 116: SGL Descriptor Sub Type Values**

SGL Descriptor Sub Type	SGL Descriptor Types	Sub Type Description
	1h	The controller shall abort the command with the status code of SGL Descriptor Type Invalid.
	4h	The controller shall abort the command with the status code of SGL Descriptor Type Invalid.
	All other values	Reserved
Ah to Fh	All	<b>NVMe Transport Specific:</b> The definitions for this range of Sub Types are defined by the binding section for the associated NVMe Transport.
All other values	All	Reserved

The SGL Data Block descriptor, defined in Figure 117, describes a data block.

**Figure 117: SGL Data Block descriptor**

Bytes	Description						
07:00	<p><b>Address (ADDR):</b> If the SGL Identifier Descriptor Sub Type field is cleared to 0h, then this field specifies the starting 64-bit memory byte address of the data block. If the SGL Identifier Descriptor Sub Type field is set to 1h, then this field contains an offset from the beginning of the location where data may be transferred. If the controller requires dword alignment and granularity as indicated in the SGL Support (SGLS) field of the Identify Controller data structure (refer to Figure 312), then the two least significant bits shall be cleared to 00b.</p> <p>If dword alignment and granularity is required, the controller may report an error of Invalid Field in Command if bits 1:0 are not cleared to 00b. If the controller does not report an error of Invalid Field in Command, then the controller shall operate as if bits 1:0 are cleared to 00b.</p>						
11:08	<p><b>Length (LEN):</b> This field specifies the length in bytes of the data block. This field cleared to 0h specifies that no data is transferred. An SGL Data Block descriptor specifying that no data is transferred is a valid SGL Data Block descriptor. If the controller requires dword alignment and granularity as specified in the SGL Support (SGLS) field the of Identify Controller data structure, then the two least significant bits shall be cleared to 00b.</p> <p>If dword alignment and granularity is required, the controller may report an error of Invalid Field in Command if bits 1:0 are not cleared to 00b. If the controller does not report an error of Invalid Field in Command, then the controller shall operate as if bits 1:0 are cleared to 00b.</p> <p>If the value in the Address field plus the value in this field is greater than 1_00000000_00000000h, then the SGL Data Block descriptor shall be processed as having a Data SGL Length Invalid or Metadata SGL Length Invalid error.</p>						
14:12	Reserved						
15	<p><b>SGL Identifier (SGLID):</b> The definition of this field is described in the table below.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>07:04</td> <td><b>SGL Descriptor Type (SGLDT):</b> 0h as specified in Figure 115.</td> </tr> <tr> <td>03:00</td> <td><b>SGL Descriptor Sub Type (SGLDST):</b> Valid values are specified in Figure 116.</td> </tr> </tbody> </table>	Bits	Description	07:04	<b>SGL Descriptor Type (SGLDT):</b> 0h as specified in Figure 115.	03:00	<b>SGL Descriptor Sub Type (SGLDST):</b> Valid values are specified in Figure 116.
Bits	Description						
07:04	<b>SGL Descriptor Type (SGLDT):</b> 0h as specified in Figure 115.						
03:00	<b>SGL Descriptor Sub Type (SGLDST):</b> Valid values are specified in Figure 116.						

The SGL Bit Bucket descriptor, defined in Figure 118, is used to ignore parts of source data.

**Figure 118: SGL Bit Bucket descriptor**

Bytes	Description
07:00	Reserved

**Figure 118: SGL Bit Bucket descriptor**

Bytes	Description						
11:08	<p><b>Length (LEN):</b> This field specifies the amount of source data that is discarded. An SGL Bit Bucket descriptor specifying that no source data be discarded (i.e., this field cleared to 0h) is a valid SGL Bit Bucket descriptor.</p> <p>If the SGL Bit Bucket descriptor describes a destination data buffer (e.g., a read from the controller to host memory), then this field specifies the number of bytes of the source data which the controller shall discard (i.e., not transfer to the destination data buffer).</p> <p>If the SGL Bit Bucket descriptor describes a source data buffer (e.g., a write from host memory to the controller), then the Bit Bucket Descriptor shall be treated as if this field were cleared to 0h (i.e., the Bit Bucket Descriptor has no effect).</p> <p>If SGL Bit Bucket descriptors are supported, their length in a destination data buffer shall be included in the specified length of data to be transferred (e.g., their length in a source data buffer is not included in the transfer length specified by the NLB parameter).</p>						
14:12	Reserved						
15	<p><b>SGL Identifier (SGLID):</b> The definition of this field is described in the table below.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>07:04</td> <td><b>SGL Descriptor Type (SGLDT):</b> 1h as specified in Figure 115.</td> </tr> <tr> <td>03:00</td> <td><b>SGL Descriptor Sub Type (SGLDST):</b> Valid values are specified in Figure 116.</td> </tr> </tbody> </table>	Bits	Description	07:04	<b>SGL Descriptor Type (SGLDT):</b> 1h as specified in Figure 115.	03:00	<b>SGL Descriptor Sub Type (SGLDST):</b> Valid values are specified in Figure 116.
Bits	Description						
07:04	<b>SGL Descriptor Type (SGLDT):</b> 1h as specified in Figure 115.						
03:00	<b>SGL Descriptor Sub Type (SGLDST):</b> Valid values are specified in Figure 116.						

The SGL Segment descriptor, defined in Figure 119, describes the next SGL segment, which is not the last SGL segment.

**Figure 119: SGL Segment descriptor**

Bytes	Description						
07:00	<p><b>Address (ADDR):</b> If the SGL Descriptor Sub Type field is cleared to 0h in the SGL Identifier field, then this field specifies the starting 64-bit memory byte address of the next SGL segment, which is an SGL segment. If the SGL Descriptor Sub Type field is set to 1h in the SGL Identifier field, then this field contains an offset from the beginning of the location where data may be transferred.</p>						
11:08	<p><b>Length (LEN):</b> This field specifies the length in bytes of the next SGL segment. This field shall be a non-zero value and a multiple of 16.</p> <p>If the value in the Address field plus the value in this field is greater than 1_00000000_00000000h, then the SGL Segment descriptor shall be processed as having a Data SGL Length Invalid or Metadata SGL Length Invalid error.</p>						
14:12	Reserved						
15	<p><b>SGL Identifier (SGLID):</b> The definition of this field is described in the table below.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>07:04</td> <td><b>SGL Descriptor Type (SGLDT):</b> 2h as specified in Figure 115.</td> </tr> <tr> <td>03:00</td> <td><b>SGL Descriptor Sub Type (SGDST):</b> Valid values are specified in Figure 116.</td> </tr> </tbody> </table>	Bits	Description	07:04	<b>SGL Descriptor Type (SGLDT):</b> 2h as specified in Figure 115.	03:00	<b>SGL Descriptor Sub Type (SGDST):</b> Valid values are specified in Figure 116.
Bits	Description						
07:04	<b>SGL Descriptor Type (SGLDT):</b> 2h as specified in Figure 115.						
03:00	<b>SGL Descriptor Sub Type (SGDST):</b> Valid values are specified in Figure 116.						

The SGL Last Segment descriptor, defined in Figure 120, describes the next and last SGL segment. A last SGL segment that contains an SGL Segment descriptor or an SGL Last Segment descriptor is processed as an error.

**Figure 120: SGL Last Segment descriptor**

Bytes	Description
07:00	<p><b>Address (ADDR):</b> If the SGL Identifier Descriptor Sub Type field is cleared to 0h, then this field specifies the starting 64-bit memory byte address of the next and last SGL segment, which is an SGL segment. If the SGL Identifier Descriptor Sub Type field is set to 1h, then this field contains an offset from the beginning of the location where data may be transferred.</p>

**Figure 120: SGL Last Segment descriptor**

Bytes	Description						
11:08	<b>Length (LEN):</b> This field specifies the length in bytes of the next and last SGL segment. This field shall be a non-zero value and a multiple of 16.  If the value in the Address field plus the value in this field is greater than 1_00000000_00000000h, then the SGL Last Segment descriptor shall be processed as having a Data SGL Length Invalid or Metadata SGL Length Invalid error.						
14:12	Reserved						
15	<b>SGL Identifier (SGLID):</b> The definition of this field is described in the table below. <table border="1" data-bbox="402 531 1377 619"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>07:04</td> <td><b>SGL Descriptor Type (SGLDT):</b> 3h as specified in Figure 115.</td> </tr> <tr> <td>03:00</td> <td><b>SGL Descriptor Sub Type (SGLDST):</b> Valid values are specified in Figure 116.</td> </tr> </tbody> </table>	Bits	Description	07:04	<b>SGL Descriptor Type (SGLDT):</b> 3h as specified in Figure 115.	03:00	<b>SGL Descriptor Sub Type (SGLDST):</b> Valid values are specified in Figure 116.
Bits	Description						
07:04	<b>SGL Descriptor Type (SGLDT):</b> 3h as specified in Figure 115.						
03:00	<b>SGL Descriptor Sub Type (SGLDST):</b> Valid values are specified in Figure 116.						

The Keyed SGL Data Block descriptor, defined in Figure 121, describes a keyed data block.

**Figure 121: Keyed SGL Data Block descriptor**

Bytes	Description						
07:00	<b>Address (ADDR):</b> This field specifies the starting 64-bit memory byte address of the data block.						
10:08	<b>Length (LEN):</b> This field specifies the length in bytes of the data block. This field cleared to 0h specifies that no data is transferred. An SGL Data Block descriptor specifying that no data is transferred is a valid SGL Data Block descriptor.  If the value in the Address field plus the value in this field is greater than 1_00000000_00000000h, then the SGL Data Block descriptor shall be processed as having a Data SGL Length Invalid or Metadata SGL Length Invalid error.						
14:11	<b>Key (KEY):</b> Specifies a 32-bit key that is associated with the data block.						
15	<b>SGL Identifier (SGLID):</b> The definition of this field is described in the table below. <table border="1" data-bbox="402 1085 1377 1173"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>07:04</td> <td><b>SGL Descriptor Type (SGLDT):</b> 4h as specified in Figure 115.</td> </tr> <tr> <td>03:00</td> <td><b>SGL Descriptor Sub Type (SGLDST):</b> Valid values are specified in Figure 116.</td> </tr> </tbody> </table>	Bits	Description	07:04	<b>SGL Descriptor Type (SGLDT):</b> 4h as specified in Figure 115.	03:00	<b>SGL Descriptor Sub Type (SGLDST):</b> Valid values are specified in Figure 116.
Bits	Description						
07:04	<b>SGL Descriptor Type (SGLDT):</b> 4h as specified in Figure 115.						
03:00	<b>SGL Descriptor Sub Type (SGLDST):</b> Valid values are specified in Figure 116.						

**Figure 122: Transport SGL Data Block descriptor**

Bytes	Description						
07:00	Reserved						
11:08	<b>Length (LEN):</b> This field specifies the length in bytes of the data block. This field cleared to 0h specifies that no data is transferred. A Transport SGL Data Block descriptor specifying that no data is transferred is a valid Transport SGL Data Block descriptor. If the controller requires dword alignment and granularity as specified in the SGL Support field of Identify Controller (refer to Figure 312), then the two least significant bits shall be cleared to 00b.  The data transfer mechanism and data buffers for data specified by a Transport SGL Data Block descriptor are defined by the binding section for the associated NVMe Transport.						
14:12	Reserved						
15	<b>SGL Identifier (SGLID):</b> The definition of this field is described in the table below. <table border="1" data-bbox="402 1621 1377 1709"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>07:04</td> <td><b>SGL Descriptor Type (SGLDT):</b> 5h as specified in Figure 115.</td> </tr> <tr> <td>03:00</td> <td><b>SGL Descriptor Sub Type (SGLDST):</b> Valid values are specified in Figure 116.</td> </tr> </tbody> </table>	Bits	Description	07:04	<b>SGL Descriptor Type (SGLDT):</b> 5h as specified in Figure 115.	03:00	<b>SGL Descriptor Sub Type (SGLDST):</b> Valid values are specified in Figure 116.
Bits	Description						
07:04	<b>SGL Descriptor Type (SGLDT):</b> 5h as specified in Figure 115.						
03:00	<b>SGL Descriptor Sub Type (SGLDST):</b> Valid values are specified in Figure 116.						

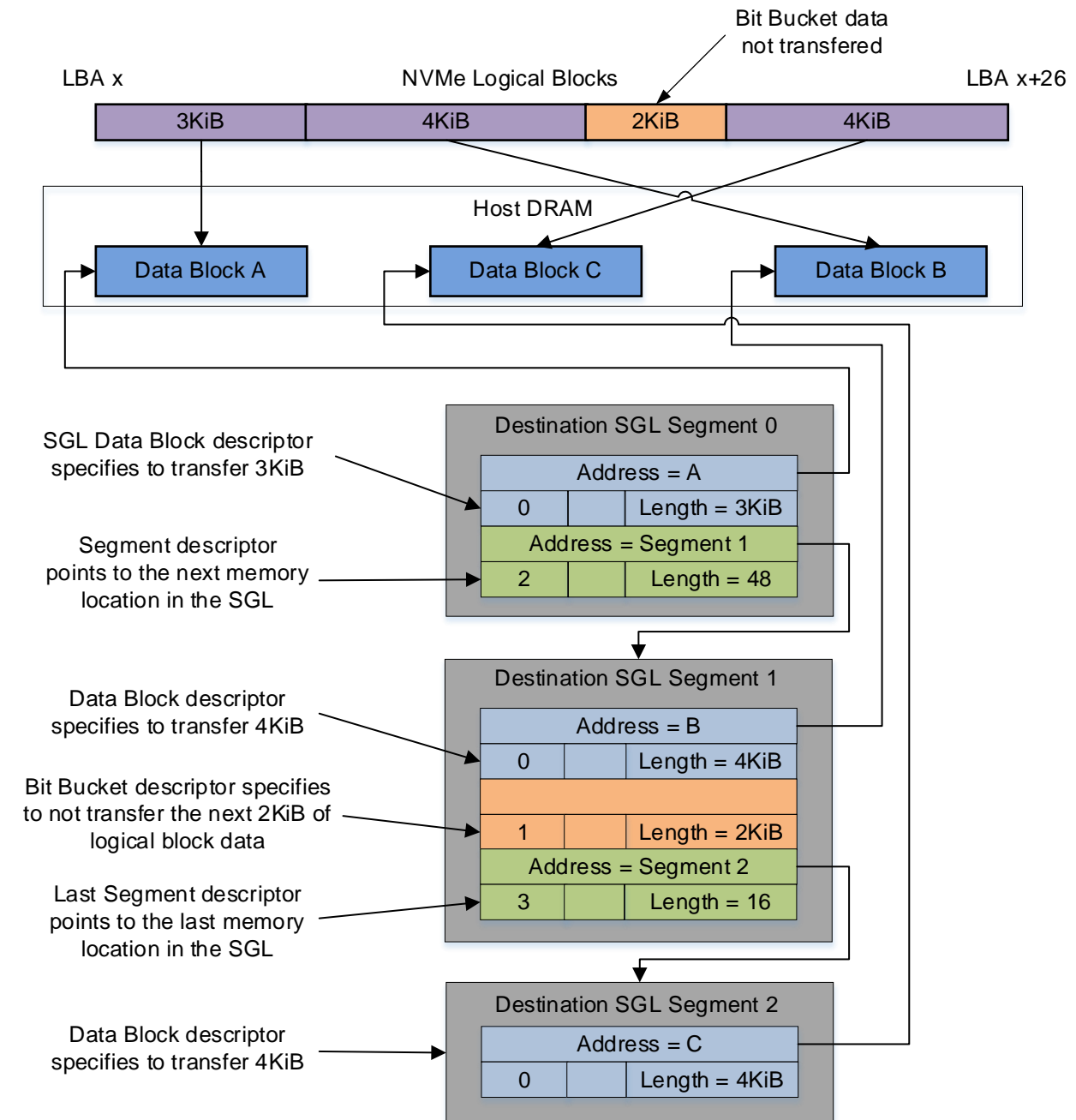
#### 4.3.2.1 SGL Example

Figure 123 shows an example of a data read request using SGLs. In the example, the logical block size is 512B. The total length of the logical blocks accessed is 13 KiB, of which only 11 KiB is transferred to the host. The Number of Logical Blocks (NLB) field in the command shall specify 26, indicating the total length

of the logical blocks accessed on the controller is 13 KiB. There are three SGL segments describing the locations in memory where the logical block data is transferred.

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

**Figure 123: SGL Read Example**





### 4.3.3 Metadata Region (MR)

The definition for the Metadata Region is command set specific. Refer to each I/O Command Set specification for applicability and additional details, if any.

## 4.4 Feature Values

The Get Features command (refer to section 5.1.11) and Set Features command (refer to section 5.1.25) 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 the Save and Select Feature Support (SSFS) bit is set to '1' in the Optional NVM Command Support (ONCS) field of the Identify Controller data structure in Figure 312, then for each Feature, there are three settings: default, saved, and current. If the SSFS bit is cleared to '0', 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 385.

If the SSFS bit is set to '1', then each Feature has supported capabilities (refer to Figure 195), which are discovered using the Supported Capabilities value in the Select field in Get Features (refer to Figure 192).

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

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 385, 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 124 for an NVM subsystem that supports only a single controller (i.e., the Multiple Ports bit of the CMIC field (refer to Figure 311) is cleared to '0').

The current value for a Feature, as a result of an NVM Subsystem Reset, is indicated in Figure 124, 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.

**Figure 124: 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"> <li>the saved value if a saved value is set; or</li> <li>the default value if a saved value is not set,</li> </ul> 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 125 for an NVM subsystem that is able to support two or more controllers (i.e., the Multiple Ports bit is set to '1' in the CMIC field).

The current value for a Feature, as a result of an NVM Subsystem Reset, is indicated in Figure 125 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 125: 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"> <li>the saved value if a saved value is set; or</li> <li>the default value if a saved value is not set,</li> </ul> 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 apply based on the feature scope (e.g., a feature is namespace specific if that feature has a namespace scope) as defined in Figure 385.

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

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

For feature values that apply to 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 5.1.25, 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 is aborted by the controller as described in Figure 92.

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 92.

There are mandatory and optional Feature Identifiers defined in section 3.1.3.6. 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.

#### **4.5 Identifier Format and Layout (Informative)**

This section provides guidance for proper implementation of various identifiers defined in the Identify Controller, Identify Namespace, and Namespace Identification Descriptor data structures.

##### **4.5.1 PCI Vendor ID (VID) and PCI Subsystem Vendor ID (SSVID)**

The PCI Vendor ID (VID, bytes 01:00) and PCI Subsystem Vendor ID (SSVID, bytes 03:02) are defined in the Identify Controller data structure. The values are assigned by the PCI SIG. Each identifier is a 16-bit number in little endian format.

Example:

- VID = ABCDh; and
- SSVID = 1234h.

**Figure 126: PCI Vendor ID (VID) and PCI Subsystem Vendor ID (SSVID)**

Bytes	00	01	02	03
Value	CDh	ABh	34h	12h

#### 4.5.2 Serial Number (SN) and Model Number (MN)

The Serial Number (SN, bytes 23:04) and Model Number (MN, bytes 63:24) are defined in the Identify Controller data structure. The values are ASCII strings assigned by the vendor. Each identifier is in big endian format.

Example (Value shown as ASCII characters):

- SN = "SN1"; and
- MN = "M2".

**Figure 127: Serial Number (SN) and Model Number (MN)**

Bytes	04	05	06	23 to 07	24	25	63 to 26
Value	53h ('S')	4Eh ('N')	31h ('1')	20h ('')	4Dh ('M')	32h ('2')	20h ('')

#### 4.5.3 IEEE OUI Identifier (IEEE)

The IEEE OUI Identifier (OUI, bytes 75:73) is defined in the Identify Controller data structure. The value is assigned by the IEEE Registration Authority. The identifier is in little endian format.

Example:

- OUI = ABCDEFh.

**Figure 128: IEEE OUI Identifier (IEEE)**

Bytes	73	74	75
Value	EFh	CDh	ABh

#### 4.5.4 IEEE Extended Unique Identifier (EUI64)

The IEEE Extended Unique Identifier (EUI64, bytes 127:120) is defined in the Identify Namespace data structure. Tutorials are available at <https://standards.ieee.org/develop/regauth/tut/index.html>. IEEE defines three formats that may be used in this field: MA-L, MA-M, and MA-S. The examples in this section use the MA-L format.

The MA-L format is defined as a string of eight octets:

**Figure 129: IEEE Extended Unique Identifier (EUI64), MA-L Format**

EUI[0]	EUI[1]	EUI[2]	EUI[3]	EUI[4]	EUI[5]	EUI[6]	EUI[7]
OUI			Extension Identifier				

EUI64 is defined in big endian format. The OUI field differs from the OUI Identifier which is in little endian format as described in section 4.5.3.

Example:

- OUI Identifier = ABCDEFh; and
- Extension Identifier = 0123456789h.

**Figure 130: IEEE Extended Unique Identifier (EUI64), OUI Identifier**

<b>Bytes</b>	<b>120</b>	<b>121</b>	<b>122</b>	<b>123</b>	<b>124</b>	<b>125</b>
<b>Value</b>	ABh	CDh	EFh	01h	23h	45h
<b>Field</b>	OUI			Extension Identifier		

**Figure 131: IEEE Extended Unique Identifier (EUI64), Ext. ID (cont)**

<b>Bytes</b>	<b>126</b>	<b>127</b>
<b>Value</b>	67h	89h
<b>Field</b>	Ext ID (cont)	

The MA-L format is similar to the World Wide Name (WWN) format defined as IEEE Registered designator (NAA = 5) as shown below.

**Figure 132: MA-L similarity to WWN**

<b>Bytes</b>	<b>0</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>	<b>6</b>	<b>7</b>
<b>EUI64</b>	OUI			Extension Identifier				
<b>WWN (NAA = 5)</b>	5h	OUI			Vendor Specific Identifier			

#### 4.5.5 Namespace Globally Unique Identifier (NGUID)

The Namespace Globally Unique Identifier (NGUID, bytes 119:104) is defined in the Identify Namespace data structure. The NGUID is composed of an IEEE OUI, an extension identifier, and a vendor specific extension identifier. The extension identifier and vendor specific extension identifier are both assigned by the vendor and may be considered as a single field. NGUID is defined in big endian format. The OUI field differs from the OUI Identifier which is in little endian format as described in section 4.5.3.

Example:

- OUI Identifier = ABCDEFh;
- Extension Identifier = 0123456789h; and
- Vendor Specific Extension Identifier = FEDCBA9876543210h.

**Figure 133: Namespace Globally Unique Identifier (NGUID)**

<b>Bytes</b>	<b>104</b>	<b>105</b>	<b>106</b>	<b>107</b>	<b>108</b>	<b>109</b>
<b>Value</b>	FEh	DCh	BAh	98h	76h	54h
<b>Field</b>	Vendor Specific Extension Identifier					

**Figure 134: Namespace Globally Unique Identifier (NGUID), OUI**

<b>Bytes</b>	<b>110</b>	<b>111</b>	<b>112</b>	<b>113</b>	<b>114</b>	<b>115</b>
<b>Value</b>	32h	10h	ABh	CDh	EFh	01h
<b>Field</b>	VSP Ex ID (cont)		OUI			Ex ID

**Figure 135: Namespace Globally Unique Identifier (NGUID), Extension Identifier (continued)**

<b>Bytes</b>	<b>116</b>	<b>117</b>	<b>118</b>	<b>119</b>
<b>Value</b>	23h	45h	67h	89h
<b>Field</b>	Extension Identifier (continued)			

The NGUID format is similar to the World Wide Name (WWN) format as IEEE Registered Extended designator (NAA = 6) as shown below.

**Figure 136: Namespace Globally Unique Identifier (NGUID), NGUID similarity to WWN**

Bytes	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
<b>NGUID</b>	Vendor Specific Extension Identifier							OUI		Extension Identifier						
<b>WWN (NAA = 6)</b>	6h	OUI			Vendor Specific Identifier			Vendor Specific Identifier Extension								

**4.5.6 Universally Unique Identifier (UUID)**

The Universally Unique Identifier is defined in RFC 9562 and contained in the Namespace Identification Descriptor (refer to Figure 315). Byte ordering requirements for a UUID are described in RFC 9562.

For historical reasons, UUID subtypes are called UUID versions. Multiple UUID versions are able to be used in the same implementation.

RFC 9562 defines UUID version 8 (i.e., UUIDv8) for experimental or vendor-specific use cases. Uniqueness of UUIDv8 values is implementation specific. NVM Express UUID use cases assume uniqueness within the set of hosts, NVM subsystems, and fabrics that are connected or accessible by a common instance of an administrative tool. For UUIDv8 values, that uniqueness is the responsibility of all of the implementations in that set. RFC 9562 Appendix B provides examples of UUIDv8 generation algorithms that produce unique UUIDs if all implementations in that set generate UUIDv8 values with the same algorithm.

**4.6 List Data Structures**

This section describes list data structures used in this specification.

**4.6.1 Controller List**

A Controller List, defined in Figure 137, is an ordered list of ascending controller identifiers. The controller identifier of a controller is the value indicated in the CNTLID field of the Identify Controller data structure in Figure 312. Unused entries are zero filled.

**Figure 137: Controller List Format**

Bytes	Description
<b>Header</b>	
01:00	<b>Number of Controller Identifiers (NUMCIDS):</b> This field contains the number of controller entries in the list. There may be up to 2,047 identifiers in the list. A value of 0h indicates there are no controllers in the list.
<b>Controller Identifier List</b>	
03:02	<b>Controller Identifier 0:</b> This field contains the NVM subsystem unique controller identifier for the first controller in the list, if present.
05:04	<b>Controller Identifier 1:</b> This field contains the NVM subsystem unique controller identifier for the second controller in the list, if present.
...	...
(NUMCIDS*2+1): (NUMCIDS*2)	<b>Controller Identifier NUMCIDS-1:</b> This field contains the NVM subsystem unique controller identifier for the last controller in the list, if present.

**4.6.2 Namespace List**

A Namespace List, defined in Figure 138, is an ordered list of namespace IDs. Unused entries are zero filled.

**Figure 138: Namespace List Format**

Bytes	Description
<b>Namespace Identifier List</b>	
03:00	<b>Namespace Identifier 0:</b> This field contains the lowest namespace ID in the list or 0h if the list is empty.

**Figure 138: Namespace List Format**

Bytes	Description
<b>Namespace Identifier List</b>	
07:04	<b>Namespace Identifier 1:</b> This field contains the second lowest namespace ID in the list or 0h if the list contains less than two entries.
...	...
(N*4+3):(N*4)	<b>Namespace Identifier N:</b> This field contains the largest namespace ID in the list or 0h if the list contains less than N+1 entries.

#### 4.7 NVMe Qualified Names

NVMe Qualified Names (NQNs) are used to uniquely describe a host or NVM subsystem for the purposes of identification and authentication. The NVMe Qualified Name for the NVM subsystem is specified in the Identify Controller data structure. An NQN is permanent for the lifetime of the host or NVM subsystem.

An NVMe Qualified Name is encoded as a UTF-8 string of Unicode characters (refer to section 1.4.2) with the following properties:

- The encoding is UTF-8 (refer to RFC 3629);
- The following characters are used in formatting:
  - dash ('-'=U+002d);
  - dot ('.'=U+002e); and
  - colon (':'=U+003a);
- The maximum name is 223 bytes in length; and
- The string is null terminated.

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

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

The reverse domain name in an NQN that uses this format shall not be “org.nvmexpress”.

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

- The string “nqn.2014-08.com.example:nvme:nvm-subsystem-sn-d78432”; and
- The string “nqn.2014-08.com.example:nvme.host.sys.xyz”.

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

- The string “nqn”;
- The string “.” (i.e., the ASCII period character);
- The string “2014-08.org.nvmexpress:uuid:”; and
- A 128-bit UUID based on the definition in RFC 9562 represented as a string formatted as “11111111-2222-3333-4444-555555555555”.

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

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

NVMe hosts, controllers and NVM subsystems process (e.g., compare for equality) UTF-8 NVMe Qualified Names without any text processing or text comparison logic that is specific to the Unicode character set or locale (refer to section 4.8).

#### 4.7.1 Unique Identifier

##### 4.7.1.1 Unique Identifier Overview

Unique identifiers include the following:

- NVM subsystem unique identifiers (refer to section 4.7.1.2);
- controller unique identifiers (refer to section 4.7.1.3); and
- namespace unique identifiers (refer to section 4.7.1.4).

Unique identifiers are either globally unique or unique only within the NVM subsystem. NVM subsystem unique identifiers are globally unique. Controller unique identifiers (i.e., the CNTLID) are unique only within the NVM subsystem. Namespace unique identifiers are globally unique.

##### 4.7.1.2 NVM Subsystem Unique Identifier

The NVM Subsystem NVMe Qualified Name specified in the Identify Controller data structure (refer to Figure 312) should be used (e.g., by host software) as the unique identifier for the NVM subsystem. If the controller is compliant with an NVM Express Specification prior to revision 1.2.1 (i.e., revisions where the NVM Subsystem NQN was not defined), then the PCI Vendor ID, Serial Number, and Model Number fields in the Identify Controller data structure and the NQN Starting String “nqn.2014.08.org.nvmexpress:” may be combined by the host to form a globally unique value that identifies the NVM subsystem (e.g., for host software that uses NQNs). The method shown in Figure 139 should be used by the host to construct an NVM Subsystem NQN for older NVM subsystems that do not provide an NQN in the Identify Controller data structure. The mechanism used by the vendor to assign Serial Number and Model Number values to ensure uniqueness is outside the scope of this specification.

**Figure 139: NQN Construction for Older NVM Subsystems**

Bytes	Description
26:00	<b>NQN Starting String (NSS):</b> Contains the 27 letter ASCII string “nqn.2014-08.org.nvmexpress:”.
30:27	<b>PCI Vendor ID (VID):</b> Contains the company vendor identifier that is assigned by the PCI SIG as a hexadecimal ASCII string.
34:31	<b>PCI Subsystem Vendor ID (SSVID):</b> Contains the company vendor identifier that is assigned by the PCI SIG for the subsystem as a hexadecimal ASCII string.
54:35	<b>Serial Number (SN):</b> Contains the serial number for the NVM subsystem that is assigned by the vendor as an ASCII string.
94:55	<b>Model Number (MN):</b> Contains the model number for the NVM subsystem that is assigned by the vendor as an ASCII string.
222:95	<b>Padding (PAD):</b> Contains spaces (ASCII character 20h).

##### 4.7.1.3 Controller Unique Identifier

An NVM subsystem may contain multiple controllers. All controllers contained in the NVM subsystem share the same NVM subsystem unique identifier. The Controller ID (CNTLID) value returned in the Identify Controller data structure may be used to uniquely identify a controller within an NVM subsystem. The Controller ID value when combined with the NVM subsystem identifier forms a globally unique value that identifies the controller. The mechanism used by the vendor to assign Controller ID values is outside the scope of this specification.

##### 4.7.1.4 Namespace Unique Identifier

The Identify Namespace data structure (refer to the applicable I/O Command Set specification) contains the IEEE Extended Unique Identifier (EUI64) and the Namespace Globally Unique Identifier (NGUID) fields. The Namespace Identification Descriptor data structure (refer to Figure 315) contains the Namespace



UUID. EUI64 is an 8-byte EUI-64 identifier (refer to section 4.5.4), NGUID is a 16-byte identifier based on EUI-64 (refer to section 4.5.5), and Namespace UUID is a 16-byte identifier described in RFC 9562 (refer to section 4.5.6).

When creating a namespace, the controller shall indicate a globally unique namespace identifier in one or more of the following:

- a) the EUI64 field;
- b) the NGUID field; or
- c) a Namespace Identification Descriptor with the Namespace Identifier Type field set to 3h (i.e., a UUID).

Refer to section 8.1.9.2 for additional globally unique namespace identifier requirements related to dispersed namespaces.

If the EUI64 field is cleared to 0h and the NGUID field is cleared to 0h, then the namespace shall support a valid Namespace UUID in the Namespace Identification Descriptor data structure.

If the UIDREUSE bit in the NSFEAT field is cleared to '0', then a controller may reuse a non-zero NGUID/EUI64 value for a new namespace after the original namespace using the value has been deleted. If the UIDREUSE bit is set to '1', then a controller shall not reuse a non-zero NGUID/EUI64 for a new namespace after the original namespace using the value has been deleted.

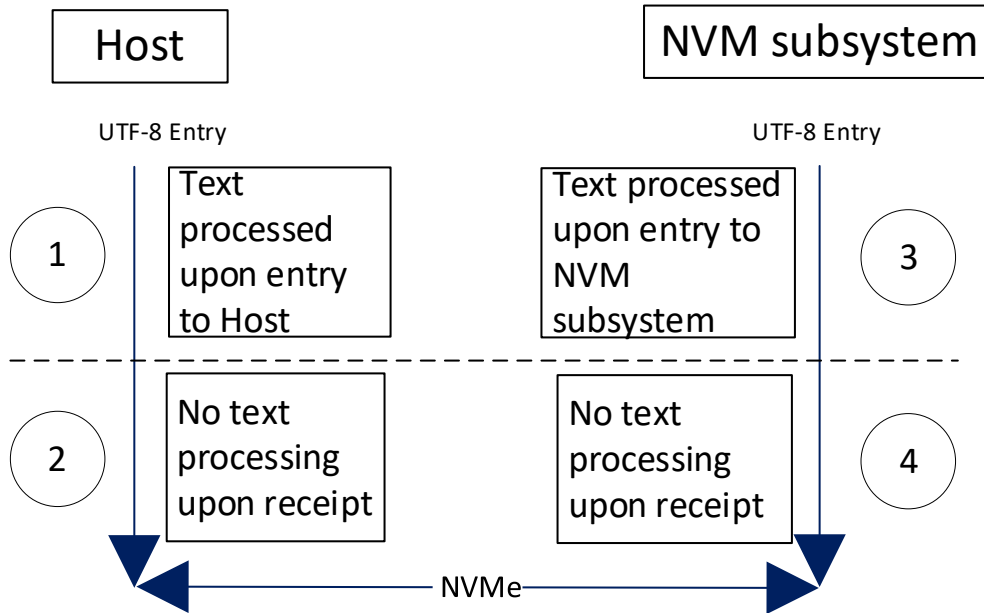
#### **4.8 UTF-8 String Processing**

NVMe hosts, controllers and NVM subsystems process (e.g., store, compare) UTF-8 strings used by NVMe as binary strings without any text processing or text comparison logic that is specific to the Unicode character set or locale (e.g., check for byte values not used by UTF-8, Unicode normalization, etc.). Any such text processing:

- a) may occur as part of entry of UTF-8 strings into NVMe hosts and NVM subsystems as shown at points 1 and 3 in Figure 140; and
- b) should not occur as part of receiving UTF-8 strings via NVMe, as shown at points 2 and 4 in Figure 140.

Upon entry into the NVMe host (e.g., via a configuration interface at point 1 in Figure 140, described as "input" in RFC 9562), NVMe host software may process a UTF-8 string (e.g., perform Unicode normalization). Upon entry into the NVM subsystem (e.g., via a configuration interface at point 3 in Figure 140, described as "input" in RFC 9562), a controller may process a UTF-8 string (e.g., perform Unicode normalization). Upon receipt by the host (e.g., at point 2 in Figure 140) of a UTF-8 string from the controller, text processing (e.g., Unicode normalization) should not occur. Upon receipt by the controller (e.g., at point 4 in Figure 140) of a UTF-8 string from the host, text processing (e.g., Unicode normalization) should not occur.

Figure 140: UTF-8 Input Processing



## 5 Admin Command Set

The Admin Command Set defines the commands that may be submitted to the Admin Submission Queue. Section 5.1 describes Admin commands that are common to all transport models. Section 5.2 describes Admin commands that are specific to the Memory-based transport model. Section 5.3 describes Admin commands that are specific to the Message-based transport model.

The submission queue entry (SQE) structure and the fields that are common to all Admin commands are defined in section 4.1. The completion queue entry (CQE) structure and the fields that are common to all Admin commands are defined in section 4.2. The command specific fields in the SQE and CQE structures (i.e., SQE Command Dwords 10 to 15, CQE Dword 0, and CQE Dword 1) for the Admin Command Set are defined in this section.

Admin commands should not be impacted by the state of I/O queues (e.g., a full I/O Completion Queue should not delay or stall the Delete I/O Submission Queue command).

Figure 141 defines all Admin commands. Refer to Figure 28 for mandatory, optional, and prohibited commands for the various controller types.

**Figure 141: Opcodes for Admin Commands**

Opcode by Field		Combined Opcode <sup>1</sup>	Namespace Identifier Used <sup>2</sup>	Command	Reference <sup>8</sup>
(07:02)	(01:00)				
Function	Data Transfer <sup>3</sup>				
0000 00b	00b	00h	No	Delete I/O Submission Queue	5.2.4
0000 00b	01b	01h	No	Create I/O Submission Queue	5.2.2
0000 00b	10b	02h	Yes <sup>11</sup>	Get Log Page	5.1.12
0000 01b	00b	04h	No	Delete I/O Completion Queue	5.2.3
0000 01b	01b	05h	No	Create I/O Completion Queue	5.2.1
0000 01b	10b	06h	NOTE 6	Identify	5.1.13
0000 10b	00b	08h	No	Abort	5.1.1
0000 10b	01b	09h	Yes	Set Features	5.1.25
0000 10b	10b	0Ah	Yes	Get Features	5.1.11
0000 11b	00b	0Ch	No	Asynchronous Event Request	5.1.2
0000 11b	01b	0Dh	Yes	Namespace Management	5.1.21
0001 00b	00b	10h	No	Firmware Commit	5.1.8
0001 00b	01b	11h	No	Firmware Image Download	5.1.9
0001 01b	00b	14h	Yes	Device Self-test	5.1.4
0001 01b	01b	15h	Yes <sup>4</sup>	Namespace Attachment	5.1.20
0001 10b	00b	18h	No	Keep Alive	5.1.14
0001 10b	01b	19h	Yes <sup>5</sup>	Directive Send	5.1.7
0001 10b	10b	1Ah	Yes <sup>5</sup>	Directive Receive	5.1.6
0001 11b	00b	1Ch	No	Virtualization Management	5.2.6
0001 11b	01b	1Dh	No	NVMe-MI Send	5.1.19
0001 11b	10b	1Eh	No	NVMe-MI Receive	5.1.18
0010 00b	00b	20h	No	Capacity Management	5.1.3
0010 00b	01b	21h	No	Discovery Information Management	5.3.3
0010 00b	10b	22h	No	Fabric Zoning Receive	5.3.5
0010 01b	00b	24h	No	Lockdown	5.1.15
0010 01b	01b	25h	No	Fabric Zoning Lookup	5.3.4
0010 10b	00b	28h	No	Clear Exported NVM Resource Configuration <sup>10</sup>	5.3.1
0010 10b	01b	29h	No	Fabric Zoning Send	5.3.6
0010 10b	10b	2Ah	No	Create Exported NVM Subsystem <sup>10</sup>	5.3.2

Figure 141: Opcodes for Admin Commands

Opcode by Field		Combined Opcode <sup>1</sup>	Namespace Identifier Used <sup>2</sup>	Command	Reference <sup>8</sup>
(07:02)	(01:00)				
Function	Data Transfer <sup>3</sup>				
0010 11b	01b	2Dh	No	Manage Exported NVM Subsystem <sup>10</sup>	5.3.8
0011 00b	01b	31h	Yes	Manage Exported Namespace <sup>10</sup>	5.3.7
0011 01b	01b	35h	No	Manage Exported Port <sup>10</sup>	5.3.9
0011 10b	01b	39h	No	Send Discovery Log Page	5.3.10
0011 11b	01b	3Dh	No	Track Send	5.1.27
0011 11b	10b	3Eh	No	Track Receive	5.1.26
0100 00b	01b	41h	No	Migration Send	5.1.17
0100 00b	10b	42h	No	Migration Receive	5.1.16
0100 01b	01b	45h	No	Controller Data Queue	5.1.4
0111 11b	00b	7Ch	No	Doorbell Buffer Config	5.2.5
0111 11b	11b <sup>9</sup>	7Fh	No	Fabrics Commands <sup>9</sup>	6
1000 00b	00b	80h	Yes	Format NVM	5.1.10
1000 00b	01b	81h	NOTE 7	Security Send	5.1.24
1000 00b	10b	82h	NOTE 7	Security Receive	5.1.23
1000 01b	00b	84h	No	Sanitize	5.1.22
1000 01b	01b	85h	Yes	Load Program	CP
1000 01b	10b	86h	Yes <sup>4</sup>	Get LBA Status	NVM, ZNS
1000 10b	00b	88h	Yes	Program Activation Management	CP
1000 10b	01b	89h	Yes	Memory Range Set Management	CP
Vendor Specific					
11xx xxb	NOTE 3	C0h to FFh		Vendor specific	
<b>Notes:</b> 1. Opcodes not listed are reserved. 2. A subset of commands use the Namespace Identifier (NSID) field. If the Namespace Identifier field is used, then the value FFFFFFFFh is supported in this field unless otherwise indicated in footnotes in this figure that a specific command does not support that value or supports that value only under specific conditions. When this field is not used, the field is cleared to 0h as described in Figure 92. 3. 00b = no data transfer; 01b = host to controller; 10b = controller to host; 11b = bidirectional. Refer to the Data Transfer Direction field in Figure 91. 4. This command does not support the use of the Namespace Identifier (NSID) field set to FFFFFFFFh. 5. Support for the Namespace Identifier field set to FFFFFFFFh depends on the Directive Operation (refer to section 8.1.8). 6. Use of the Namespace Identifier field depends on the CNS value in the Identify Command as described in Figure 310. 7. The use of the Namespace Identifier is Security Protocol specific. 8. Section 5.1 contains commands common to all transport models, section 5.2 contains commands specific to the Memory-based transport model, and section 5.3 contains commands specific to the Message-based transport model. NVM = NVM Command Set specific, ZNS = Zoned Namespace Command Set specific, CP = Computational Programs Command Set specific. 9. All Fabrics commands use the opcode 7Fh with the data transfer direction specified as shown in Figure 540. Refer to section 6 for details. 10. Support for this command is prohibited in NVM subsystems that use a Memory-Based Transport Model (e.g., the PCIe transport) for any controller. 11. Use of the Namespace Identifier field is specified further in section 5.1.12.1 and Figure 202.					

Figure 142 lists the Admin commands that are allowed while a sanitize operation is in progress and the Admin commands that should be allowed during the processing of a Format NVM command.

If a Format NVM command is in progress, then an Admin command not listed in Figure 142 that is submitted for any namespace affected by that Format NVM command may be aborted. If aborted for that reason, then a status code of Format in Progress should be returned.

If there are Admin commands not listed in Figure 142 being processed for a namespace, then a Format NVM command which is submitted that affects that namespace may be aborted. If aborted for that reason, then a status code of Command Sequence Error should be returned.

**Figure 142: Sanitize Operations and Format NVM Command – Admin Commands Allowed**

Admin Command	Additional Restrictions for a Format NVM Command	Additional Restrictions for a Sanitize Operation
Abort		
Asynchronous Event Request		
Create I/O Completion Queue		
Create I/O Submission Queue		
Device Self-test	Only Controller DST should be allowed	Prohibited
Delete I/O Completion Queue		
Delete I/O Submission Queue		
Get Features		
Get Log Page	The log pages allowed for both Format NVM command and sanitize operations are listed below.	
	<b>Log Pages</b>	<b>Additional Restrictions for both Format NVM command and sanitize operations</b>
	Error Information	Return 0h in the LBA field.
	SMART / Health Information	
	Changed Attached Namespace List	
	Reservation Notification	
	Asymmetric Namespace Access	
	Sanitize Status	
	Vendor specific	Prohibited for Sanitize
Persistent Event Log	Prohibited for Sanitize	
Boot Partition		
Identify		
Keep Alive		
NVMe-MI Receive	Allowed unless explicitly prohibited in the NVM Express Management Interface Specification.	
NVMe-MI Send		
Sanitize		Prohibited during sanitize operations if the SANACT field is set to a value other than 101b (i.e., Exit Media Verification State). Sanitize operations are described in section 8.1.24.
Set Features	Namespace Write Protection Config Feature is not allowed.	
Opcode 7Fh	The Fabrics commands allowed are listed below. Refer to section 6.	
	<b>Fabrics command</b>	<b>Additional Restrictions for a Format NVM Command and a Sanitize Operation</b>
	Property Set	
	Connect	
	Disconnect	
	Property Get	
	Authentication Send	
Authentication Receive		
Vendor Specific	Commands are allowed that do not affect or retrieve user data.	
Vendor Specific	Commands are allowed that do not affect or retrieve user data.	

## 5.1 Common Admin Commands

This section describes Admin commands that are common to all transport models.

### 5.1.1 Abort command

The Abort command is used to request an abort of a specific command previously submitted to the Admin Submission Queue or an I/O Submission Queue. An Abort command may or may not abort the command that was requested to be aborted. For example, the command to abort may have already completed, currently be in execution, or may be deeply queued.

The controller should process the Abort command as soon as the command is fetched.

To abort a large number of commands (e.g., a larger number of commands than the limit listed in the ACL field), the host should:

- use the Cancel command (refer to section 7.1), if supported, to abort all commands submitted to an I/O Submission Queue or to abort all commands submitted to an I/O Submission Queue for a specific namespace; or
- follow the procedures described in section 3.3.1.3 to delete the I/O Submission Queue and recreate the I/O Submission Queue.

If an abort action is performed on the command to abort, that action may be:

- an immediate abort (i.e., the abort action occurs prior to posting the completion queue entry for the Abort command); or
- a deferred abort (i.e., the abort action occurs after posting the completion queue entry for the Abort command).

If an immediate abort is performed, then the controller shall either:

- post the completion queue entry for the command to abort to the appropriate Admin Completion Queue or I/O Completion Queue with a status code of Command Abort Requested before the completion queue entry for the Abort command is posted to the Admin Completion Queue; or
- ensure that there are no subsequent effects of the command to abort, as described in section 5.1.1.1, prior to posting the completion queue entry for the Abort command and post the completion queue entry for the command to abort to the appropriate Admin Completion Queue or I/O Completion Queue with a status code of Command Abort Requested. The completion queue entry for the Abort command and the completion queue entry for the command to abort are allowed to be posted in any order.

If an immediate abort is not performed, then the controller may or may not perform a deferred abort.

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

The Abort Command Limit field in the Identify Controller data structure (refer to Figure 312) indicates the controller limit on concurrent execution of Abort commands. A host should not allow the number of outstanding Abort commands to exceed this value. The controller may complete any excess Abort commands with the status code set to Abort Command Limit Exceeded.

**Figure 143: Abort – Command Dword 10**

Bits	Description
31:16	<b>Command Identifier (CID):</b> This field specifies the command identifier of the command to be aborted, that was specified in the CDW0.CID field within the command itself.
15:00	<b>Submission Queue Identifier (SQID):</b> This field specifies the identifier of the Submission Queue that the command to be aborted is associated with.

#### 5.1.1.1 Immediate Abort requirements

If the controller performs an immediate abort, then the controller shall ensure that, other than the posting of the completion queue entry for the command to abort, there are no effects of that command (e.g., no

host memory access, command processing does not modify: NVM media, NVM Set state, Endurance Group state, namespace state, controller state, domain state, or NVM subsystem state) subsequent to the posting of the CQE for the command that requested the abort. If the controller is not able to ensure there are no effects of the command to abort subsequent to the posting of the CQE for the command that requested the abort, then the controller shall not perform an immediate abort on that command.

For example, if a command:

- requests a data transfer;
- the data transfer has been initiated and not completed; and
- the CQE for that command was not posted prior to the posting of the CQE for the command that requested the abort,

then the controller is prohibited from performing an immediate abort.

Performing an immediate abort effects the information returned in the CQE for the command that requested the abort as described in section 5.1.1.2 for an Abort command and in section 7.1.1 for a Cancel command.

### 5.1.1.2 Command Completion

Upon completion of the Abort command, the controller posts a completion queue entry to the Admin Completion Queue indicating the status for the Abort command. Dword 0 of the completion queue entry indicates information about the command to abort as shown in Figure 144.

**Figure 144: Abort Command – Completion Queue Entry Dword 0**

Bits	Description
31:01	Reserved
00	<b>Immediate Abort Not Performed (IANP):</b> Indicates whether or not an immediate abort was performed. If this bit is set to '1', then an immediate abort was not performed (i.e., the command to abort was not aborted prior to posting the CQE for the Abort command) for any reason (e.g., the immediate abort requirements were not met, the requested command to abort was not found). If this bit is cleared to '0', then an immediate abort was performed (i.e., the command to abort was aborted prior to posting the CQE for the Abort command).

If the Immediate Abort Not Performed bit in Dword 0 (refer to Figure 144) is set to '1' in the completion queue entry for the Abort command and a deferred abort is performed, then the controller shall set the status code for the command to abort to Command Abort Requested. The host should examine the status in the completion queue entry of the command to abort to determine if the command was aborted.

Command specific status code values associated with the Abort command are defined in Figure 145.

**Figure 145: Abort – Command Specific Status Values**

Value	Description
3h	<b>Abort Command Limit Exceeded:</b> The number of concurrently outstanding Abort commands has exceeded the limit indicated in the Identify Controller data structure.

### 5.1.2 Asynchronous Event Request command

Asynchronous events are used to notify a host of status, error, and health information as these events occur. To enable asynchronous events to be reported by the controller, a host needs to submit one or more Asynchronous Event Request commands to the controller. The controller specifies an event to the host by completing an Asynchronous Event Request command. A host should expect that the controller may not execute the command immediately; the command should be completed when there is an event to be reported.

The Asynchronous Event Request command is submitted by a host to enable the reporting of asynchronous events from the controller. This command has no timeout. The controller posts a completion queue entry for this command when there is an asynchronous event to report to the host. If Asynchronous Event

Request commands are outstanding when the controller is reset, then each of those commands is aborted and shall not return a CQE.

All command specific fields are reserved.

A host may submit multiple Asynchronous Event Request commands to reduce event reporting latency. The total number of simultaneously outstanding Asynchronous Event Request commands is limited by the value indicated in the Asynchronous Event Request Limit field in the Identify Controller data structure in Figure 312.

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 5.1.12). 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.

The following event types are defined:

- a) **Error event:** Indicates a general error that is not associated with a specific command (refer to Figure 149). The controller shall set the Log Page Identifier field to the identifier of the Error Information log page (i.e., 01h). To clear this event, a host reads that log page (i.e., the Error Information log page (refer to section 5.1.12.1.2)) using the Get Log Page command with the Retain Asynchronous Event bit cleared to '0';
- b) **SMART / Health Status event:** Indicates a SMART or health status event (refer to Figure 150). The controller shall set the Log Page Identifier field to the identifier of the SMART/Health Information log page (i.e., 02h). To clear this event, a host reads the SMART / Health Information log (refer to section 5.1.12.1.3) using the Get Log Page command with the Retain Asynchronous Event bit cleared to '0'. The SMART / Health conditions that trigger asynchronous events may be configured in the Asynchronous Event Configuration feature using the Set Features command (refer to section 5.1.25.1.5);
- c) **Notice event:** Indicates a general event (refer to Figure 151). The controller shall set the Log Page Identifier field to the log page identifier of the appropriate log page as described in Figure 151. To clear this event, a host reads that log page (i.e., the appropriate log page as described in Figure 151). The conditions that trigger asynchronous events may be configured in the Asynchronous Event Configuration feature using the Set Features command (refer to section 5.1.25.1.5);
- d) **I/O Command Specific Status events:** Events that are specific to an I/O command (refer to Figure 152);
- e) **Immediate events:** Events that are only reported when an outstanding Asynchronous Event Request command exists at the time the event occurs (refer to Figure 153). If the event occurs and there is no outstanding Asynchronous Event Request command, then the event shall not be reported;
- f) **One-Shot events:** Events that are only reported once and the event is cleared by the controller when that event is reported in the completion of an Asynchronous Event Request command. Refer to the description of each event to determine how to clear any pending events (e.g., refer to the Controller Data Queue feature in section 5.1.25.1.23); and
- g) **Vendor Specific event:** Indicates a vendor specific event. To clear this event, a host reads the indicated vendor specific log page using the Get Log Page command with the Retain Asynchronous Event bit cleared to '0'.

The Sanitize Operation Completed With Unexpected Deallocation asynchronous event shall be supported if the controller supports the Sanitize Config feature (refer to section 5.1.25.1.15).

Asynchronous events are reported due to a new entry being added to a log page (e.g., Error Information log page), a status change (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). A host should modify the event threshold or mask the event for transient and permanent status changes before 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.

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

### 5.1.2.1 Command Completion

A completion queue entry is posted to the Admin Completion Queue if there is an asynchronous event to report to the host. Command specific status values associated with Asynchronous Event Request are defined in Figure 146.

**Figure 146: Status Code – Command Specific Status Values**

Value	Definition
05h	<b>Asynchronous Event Request Limit Exceeded:</b> The number of concurrently outstanding Asynchronous Event Request commands has been exceeded.

Dword 0 and Dword 1 of the completion queue entry contains information about the asynchronous event. The definition of Dword 0 of the completion queue entry is in Figure 147. The definition of Dword 1 of the completion queue entry is in Figure 148.

**Figure 147: Asynchronous Event Request – Completion Queue Entry Dword 0**

Bits	Description																											
31:24	Reserved																											
23:16	<b>Log Page Identifier (LID):</b> 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 5.1.2), this field should be ignored by the host.																											
15:08	<b>Asynchronous Event Information (AEI):</b> This field indicates additional information about the asynchronous event for the event indicated in the Asynchronous Event Type field.																											
07:03	Reserved																											
02:00	<b>Asynchronous Event Type (AET):</b> Indicates the event type of the asynchronous event. <table border="1" data-bbox="586 1551 1156 1812"> <thead> <tr> <th>Value</th> <th>Definition</th> <th>Reference</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>Error status</td> <td>Figure 149</td> </tr> <tr> <td>001b</td> <td>SMART / Health status</td> <td>Figure 150</td> </tr> <tr> <td>010b</td> <td>Notice</td> <td>Figure 151</td> </tr> <tr> <td>011b</td> <td>Immediate</td> <td>Figure 153</td> </tr> <tr> <td>100b</td> <td>One-Shot</td> <td>Figure 154</td> </tr> <tr> <td>101b</td> <td>Reserved</td> <td></td> </tr> <tr> <td>110b</td> <td>I/O Command specific status</td> <td>Figure 152</td> </tr> <tr> <td>111b</td> <td>Vendor specific</td> <td></td> </tr> </tbody> </table>	Value	Definition	Reference	000b	Error status	Figure 149	001b	SMART / Health status	Figure 150	010b	Notice	Figure 151	011b	Immediate	Figure 153	100b	One-Shot	Figure 154	101b	Reserved		110b	I/O Command specific status	Figure 152	111b	Vendor specific	
Value	Definition	Reference																										
000b	Error status	Figure 149																										
001b	SMART / Health status	Figure 150																										
010b	Notice	Figure 151																										
011b	Immediate	Figure 153																										
100b	One-Shot	Figure 154																										
101b	Reserved																											
110b	I/O Command specific status	Figure 152																										
111b	Vendor specific																											

**Figure 148: Asynchronous Event Request – Completion Queue Entry Dword 1**

Bits	Description
31:00	<b>Event Specific Parameter (EVNTSP):</b> This field specifies information for a specific event. If this field is not defined by a specific event, then this field is reserved for that specific event.

**Figure 149: Asynchronous Event Information – Error Status**

Value	Definition
00h	<b>Write to Invalid Doorbell Register:</b> Host software wrote the doorbell of a queue that was not created.
01h	<b>Invalid Doorbell Write Value:</b> Host software attempted to write an invalid doorbell value. Some possible causes of this error are: <ul style="list-style-type: none"> <li>the value written was out of range of the corresponding queue's base address and size;</li> <li>the value written is the same as the previously written doorbell value;</li> <li>the number of commands that would be added as part of a doorbell write would exceed the number of available entries;</li> <li>host software attempts to add a command to a full Submission Queue; and</li> <li>host software attempts to remove a completion queue entry from an empty Completion Queue.</li> </ul>
02h	<b>Diagnostic Failure:</b> A diagnostic failure was detected. This may include a self-test operation.
03h	<b>Persistent Internal Error:</b> A failure occurred that is persistent and the controller is unable to isolate to a specific set of commands. If this error is indicated, then the CSTS.CFS bit may be set to '1' and the host should perform a reset as described in section 3.7.
04h	<b>Transient Internal Error:</b> A transient error occurred that is specific to a particular set of commands; controller operation may continue without a reset.
05h	<b>Firmware Image Load Error:</b> The firmware image could not be loaded. The controller reverted to the previously active firmware image or a baseline read-only firmware image.
06h to FFh	Reserved

**Figure 150: Asynchronous Event Information – SMART / Health Status**

Value	Definition
00h	<b>NVM subsystem Reliability:</b> NVM subsystem reliability has been compromised. This may be due to significant media errors, an internal error, the media being placed in read only mode, or a volatile memory backup device failing. This status value shall not be used if the read-only condition on the media is due to a change in the write protection state of a namespace (refer to section 8.1.16.1).
01h	<b>Temperature Threshold:</b> A temperature is greater than or equal to an over temperature threshold or less than or equal to an under temperature threshold (refer to section 5.1.25.1.3).
02h	<b>Spare Below Threshold:</b> Available spare capacity has fallen below the threshold.
03h to FFh	Reserved

**Figure 151: Asynchronous Event Information – Notice**

Value	Definition
00h	<p><b>Attached Namespace Attribute Changed:</b> Indicates a change to one or more of the following:</p> <ul style="list-style-type: none"> <li>• Identify Namespace data structures (refer to section 1.5.49) that are associated with an attached namespace;</li> <li>• the Active Namespace ID list (i.e., CNS 02h);</li> <li>• the I/O Command Set specific Active Namespace ID list (i.e., CNS 07h); or</li> <li>• other data structures as specified in applicable NVMe I/O Command Set specifications.</li> </ul> <p>This event shall be reported for any of the changes in the preceding list to namespaces that are attached to the controller, unless otherwise specified. This event may be reported by controllers compliant to previous versions of this specification for changes to namespaces that are not attached to the controller.</p> <p>To clear this event, a host issues a Get Log Page command for the Changed Attached Namespace List log page (refer to section 5.1.12.1.5) with the Retain Asynchronous Event bit cleared to '0'.</p> <p>A controller shall not send this event if:</p> <ol style="list-style-type: none"> <li>a) Namespace Status (refer to Figure 319) has changed and shutdown processing is either reported as in progress (i.e., CSTS.SHST is set to 01b) or is reported as complete (i.e., CSTS.SHST is set to 10b);</li> <li>b) that controller receives a command (e.g., a Namespace Management command or Capacity Management command) on the Admin Submission Queue (e.g., not via a Management Endpoint, refer to the NVM Express Management Interface Specification) that requests a delete operation for a namespace;</li> <li>c) the ANAGRPID field (refer to Figure 319) has changed;</li> <li>d) the RGRPID field (refer to Figure 319) has changed; or</li> <li>e) an I/O Command Set specific change occurs (refer to the applicable I/O Command Set specification).</li> </ol> <p>A controller shall only send this event for changes to the Remaining Format NVM (RFNVM) field in the Format Progress Indicator field transition from a non-zero value to 0h, or from 0h to a non-zero value.</p>
01h	<p><b>Firmware Activation Starting:</b> The controller is starting a firmware activation process during which command processing is paused. The host may use CSTS.PP to determine when command processing has resumed. To clear this event, the host reads the Firmware Slot Information log page with the Retain Asynchronous Event bit cleared to '0'.</p>
02h	<p><b>Telemetry Log Changed:</b> The controller has saved the controller internal state in the Telemetry Controller-Initiated log page and set the Telemetry Controller-Initiated Data Available field to 1h in that log page. To clear this event, the host issues a Get Log Page command with Retain Asynchronous Event bit cleared to '0' for the Telemetry Controller-Initiated log.</p>
03h	<p><b>Asymmetric Namespace Access Change:</b> The Asymmetric Namespace Access information (refer to section 5.1.12.1.13) related to an ANA Group that contains namespaces attached to this controller has changed (e.g., an ANA state has changed, an ANAGRPID has changed). The current Asymmetric Namespace Access information for attached namespaces is indicated in the Asymmetric Namespace Access log page (refer to section 5.1.12.1.13). To clear this event, the host issues a Get Log Page command (refer to section 5.1.12) with the Retain Asynchronous Event bit cleared to '0' for the Asymmetric Namespace Access log.</p> <p>A controller shall not send this event if an Attached Namespace Attribute Changed asynchronous event or an Allocated Namespace Attribute Changed asynchronous event is sent for the same event, such as a change due to:</p> <ol style="list-style-type: none"> <li>a) the attachment of a namespace (refer to section 5.1.20);</li> <li>b) the deletion of a namespace (refer to section 8.1.15.2); or</li> <li>c) the detachment of a namespace (refer to section 5.1.20).</li> </ol>
04h	<p><b>Predictable Latency Event Aggregate Log Change:</b> Indicates that event pending entries for one or more NVM Sets (refer to section 5.1.12.1.12) have been added to the Predictable Latency Event Aggregate log. To clear this event, the host reads the Predictable Latency Event Aggregate log page with the Retain Asynchronous Event bit cleared to '0'.</p>

**Figure 151: Asynchronous Event Information – Notice**

Value	Definition
05h <sup>1</sup>	<b>LBA Status Information Alert:</b> I/O Command Set specific definition.
06h	<b>Endurance Group Event Aggregate Log Page Change:</b> Indicates that event entries for one or more Endurance Groups (refer to section 5.1.12.1.10) have been added to the Endurance Group Event Aggregate log. To clear this event, the host issues a Get Log Page command with the Retain Asynchronous Event bit cleared to '0' for the Endurance Group Event Aggregate log.
07h	<p><b>Reachability Group Change:</b> The Reachability Group information (refer to section 5.1.12.1.25) related to a Reachability Group that contains namespaces attached to this controller has changed (e.g., a member was added to or deleted from a Reachability Group). The current Reachability Group information for attached namespaces is indicated in the Reachability Groups log page. To clear this event, the host issues a Get Log Page command (refer to section 5.1.12) with the Retain Asynchronous Event bit cleared to '0' for the Reachability Groups log.</p> <p>A controller shall not send this event if an Attached Namespace Attribute Changed asynchronous event or an Allocated Namespace Attribute Changed asynchronous event is sent for the same event, such as a change due to:</p> <ul style="list-style-type: none"> <li>a) the attachment of a namespace (refer to section 5.1.20);</li> <li>b) the deletion of a namespace (refer to section 5.1.21); or</li> <li>c) the detachment of a namespace (refer to section 5.1.20).</li> </ul>
08h	<p><b>Reachability Association Change:</b> The Reachability Association information (refer to section 5.1.12.1.26) related to a Reachability Association that contains namespaces attached to this controller has changed (e.g., a member was added to or deleted from a Reachability Association). The current Reachability Association information for attached namespaces is indicated in the Reachability Associations log page. To clear this event, the host issues a Get Log Page command (refer to section 5.1.12) with the Retain Asynchronous Event bit cleared to '0' for the Reachability Associations log.</p> <p>A controller shall not send this event if an Attached Namespace Attribute Changed asynchronous event or an Allocated Namespace Attribute Changed asynchronous event is sent for the same event, such as a change due to:</p> <ul style="list-style-type: none"> <li>a) the attachment of a namespace (refer to section 5.1.20);</li> <li>b) the deletion of a namespace (refer to section 5.1.21); or</li> <li>c) the detachment of a namespace (refer to section 5.1.20).</li> </ul>

**Figure 151: Asynchronous Event Information – Notice**

Value	Definition
09	<p><b>Allocated Namespace Attribute Changed:</b> Indicates a change to one or more of the following:</p> <ul style="list-style-type: none"> <li>Identify Namespace data structures (refer to section 1.5.49) that are associated with an allocated namespace (refer to section 1.5.6);</li> <li>the Active Namespace ID list (i.e., CNS 02h);</li> <li>the I/O Command Set specific Active Namespace ID list (i.e., CNS 07h);</li> <li>the Allocated Namespace ID list (i.e., CNS 10h);</li> <li>the I/O Command Set specific Allocated Namespace ID list (i.e., CNS 1Ah); or</li> <li>other data structures as specified in the applicable NVMe I/O Command Set specifications.</li> </ul> <p>This event shall be reported for any of the changes in the preceding list to namespaces that are either attached to the controller or not attached to the controller, unless otherwise specified.</p> <p>To clear this event, a host issues a Get Log Page command for the Changed Allocated Namespace List log page (refer to section 5.1.12.1.27) with the Retain Asynchronous Event bit cleared to '0'.</p> <p>A controller shall not send this event if:</p> <ol style="list-style-type: none"> <li>Namespace Status (refer to Figure 319) has changed and shutdown processing is either occurring (i.e., CSTS.SHST is set to 01b) or complete (i.e., CSTS.SHST is set to 10b);</li> <li>that controller receives a command (e.g., a Namespace Management command or Capacity Management command) on the Admin Submission Queue (e.g., not via a Management Endpoint, refer to the NVMe Express Management Interface Specification) that requests a delete operation for a namespace;</li> <li>the ANAGRPID field (refer to Figure 319) has changed;</li> <li>the RGRPID field (refer to Figure 319) has changed; or</li> <li>an I/O Command Set specific change occurs (refer to the applicable I/O Command Set specification).</li> </ol> <p>A controller shall only send this event for changes to the Format Progress Indicator (FPI) field when the Remaining Format NVMe field of that FPI field transition from a non-zero value to 0h, or from 0h to a non-zero value.</p>
0Ah to EEh	Reserved
EFh <sup>2</sup>	<b>Zone Descriptor Changed:</b> I/O Command Set specific definition.
F0h	<b>Discovery Log Page Change:</b> A change has occurred to one or more of the Discovery log pages. To clear this event, the host or Discovery controller reads the Discovery log page with the Retain Asynchronous Event bit cleared to '0'.
F1h	<b>Host Discovery Log Page Change:</b> A change has occurred to one or more of the Host Discovery log pages. The host or Discovery controller should submit a Get Log Page command to receive updated Host Discovery log pages.
F2h	<b>AVE Discovery Log Page Change:</b> A change has occurred to the AVE Discovery log page. The host or controller should submit a Get Log Page command to receive an updated AVE Discovery log page.
F3h	<b>Pull Model DDC Request:</b> A pull model DDC is requesting a CDC to perform an operation.
F4h to FFh	Reserved for future NVMe over Fabrics Asynchronous Event Notifications
<p>Notes:</p> <ol style="list-style-type: none"> <li>Refer to the NVMe Command Set Specification.</li> <li>Refer to the Zoned Namespace Command Set Specification.</li> </ol>	

**Figure 152: Asynchronous Event Information – I/O Command Specific Status**

Value	Definition
00h	<p><b>Reservation Log Page Available:</b> Indicates that one or more Reservation Notification log pages (refer to section 5.1.12.1.32) have been added to the Reservation Notification log. To clear this event, the host reads the Reservation log page with the Retain Asynchronous Event bit cleared to '0'.</p>

**Figure 152: Asynchronous Event Information – I/O Command Specific Status**

Value	Definition
01h	<b>Sanitize Operation Completed:</b> Indicates that a sanitize operation has completed (including any associated additional media modification, refer to the No-Deallocate Modifies Media After Sanitize field in Figure 312) without unexpected deallocation of all media allocated for user data (refer to section 5.1.25.1.15) and status is available in the Sanitize Status log page (refer to section 5.1.12.1.33). To clear this event, the host reads the Sanitize Status log page with the Retain Asynchronous Event bit cleared to '0'.
02h	<b>Sanitize Operation Completed With Unexpected Deallocation:</b> Indicates that a sanitize operation for which No-Deallocate After Sanitize (refer to Figure 372) was requested has completed with the unexpected deallocation of all media allocated for user data (refer to section 5.1.25.1.15) and status is available in the Sanitize Status log page (refer to section 5.1.12.1.33). To clear this event, the host reads the Sanitize Status log page with the Retain Asynchronous Event bit cleared to '0'.
03h	<b>Sanitize Operation Entered Media Verification State:</b> Indicates that sanitize processing was successful, the sanitize operation entered the Media Verification state (refer to section 8.1.24), and status is available in the Sanitize Status log page (refer to section 5.1.12.1.33).
04h to FFh	Reserved

**Figure 153: Asynchronous Event Information – Immediate**

Value	Description
00h	<b>NVM Subsystem Normal Shutdown:</b> This controller has started performing a normal NVM Subsystem Shutdown that is due to: <ul style="list-style-type: none"> <li>the value 4E726D6Ch ("Nrml") has been written to the NSSD.NSSC field within the NVM subsystem or Domain; or</li> <li>an NVMe-MI Shutdown command (refer to the NVM Express Management Interface Specification) being processed.</li> </ul> Refer to section 3.6.3.
01h	<b>Temperature Threshold Hysteresis Recovery:</b> Indicates the end of a temperature threshold hysteresis event (Refer to section 5.1.25.1.3.1).
02h to FFh	Reserved

**Figure 154: Asynchronous Event Information – One Shot**

Value	Definition
00h	<b>Controller Data Queue Tail Pointer:</b> Indicates that an entry in a Controller Data Queue was posted at the tail pointer location specified by the Controller Data Queue feature (refer to section 5.1.25.1.23). Figure 155 defines the Event Specific Parameter field in completion queue entry Dword 1 field for this event.
01h	<b>Controller Data Queue Full Error:</b> The controller detected that a Controller Data Queue became full. Figure 155 defines the Event Specific Parameter field in Completion Queue Entry Dword 1 for this event.
02h to FFh	Reserved

**Figure 155: Controller Data Queue Tail Pointer – Event Specific Parameter**

Bits	Description
31:16	Reserved
15:00	<b>Controller Data Queue Identifier (CDQID):</b> This field indicates the identifier for the Controller Data Queue associated with this event (refer to Figure 168).

### 5.1.3 Capacity Management command

Host software uses the Capacity Management command to configure Endurance Groups and NVM Sets in an NVM subsystem by either selecting one of a set of supported configurations (i.e., Fixed Capacity Management; refer to section 8.1.4.2) or by specifying the capacity of the Endurance Group or NVM Set to be created (i.e., Variable Capacity Management; refer to section 8.1.4.3). The Capacity Management command specifies the operations defined in Figure 156.

The Capacity Management command uses the Command Dword 10, Command Dword 11, and Command Dword 12 fields. All other command specific fields are reserved.

For requirements to support the operations in Figure 156, refer to section 8.1.4.

**Figure 156: Capacity Management – Command Dword 10**

Bits	Description																					
31:16	<b>Element Identifier (ELID):</b> This field contains a value specific to the value of the Operation field. <sup>1</sup>																					
15:04	Reserved																					
03:00	<b>Operation (OPER):</b> Specifies the operation to be performed by the controller:																					
	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> <th>Element Identifier Field</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td><b>Select Capacity Configuration:</b> Endurance Groups and NVM Sets are configured as indicated by the Capacity Configuration Descriptor specified by this command (refer to section 5.1.3.1).</td> <td>Capacity Configuration Identifier (refer to Figure 257).</td> </tr> <tr> <td>1h</td> <td><b>Create Endurance Group:</b> An Endurance Group is created (refer to section 8.1.4.3) with the capacity specified by the Capacity Lower field (refer to Figure 157) and the Capacity Upper field (refer to Figure 158).</td> <td>Domain Identifier (refer to section 3.2.5.3) of the domain in which the Endurance Group is to be created. A value of 0h specifies that the controller selects the domain in which the Endurance Group is created.</td> </tr> <tr> <td>2h</td> <td><b>Delete Endurance Group:</b> The Endurance Group specified by this command is deleted (refer to section 8.1.4.3). All namespaces and NVM Sets contained by the Endurance Group are deleted.</td> <td>Endurance Group Identifier of the Endurance Group to be deleted.</td> </tr> <tr> <td>3h</td> <td><b>Create NVM Set:</b> An NVM Set is created (refer to section 8.1.4.3) in the specified Endurance Group with the capacity specified by the Capacity Lower field and the Capacity Upper field. If the controller does not support NVM Sets, then this operation is not supported.</td> <td>Endurance Group Identifier of the Endurance Group in which the NVM Set is to be created. A value of 0h specifies that the controller selects the Endurance Group in which the NVM Set is created.</td> </tr> <tr> <td>4h</td> <td><b>Delete NVM Set:</b> The NVM Set specified by this command is deleted (refer to section 8.1.4.3). All namespaces in the NVM Set are deleted. If the controller does not support NVM Sets, then this operation is not supported.</td> <td>NVM Set identifier of the NVM Set to be deleted.</td> </tr> <tr> <td>5h to Fh</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Value	Description	Element Identifier Field	0h	<b>Select Capacity Configuration:</b> Endurance Groups and NVM Sets are configured as indicated by the Capacity Configuration Descriptor specified by this command (refer to section 5.1.3.1).	Capacity Configuration Identifier (refer to Figure 257).	1h	<b>Create Endurance Group:</b> An Endurance Group is created (refer to section 8.1.4.3) with the capacity specified by the Capacity Lower field (refer to Figure 157) and the Capacity Upper field (refer to Figure 158).	Domain Identifier (refer to section 3.2.5.3) of the domain in which the Endurance Group is to be created. A value of 0h specifies that the controller selects the domain in which the Endurance Group is created.	2h	<b>Delete Endurance Group:</b> The Endurance Group specified by this command is deleted (refer to section 8.1.4.3). All namespaces and NVM Sets contained by the Endurance Group are deleted.	Endurance Group Identifier of the Endurance Group to be deleted.	3h	<b>Create NVM Set:</b> An NVM Set is created (refer to section 8.1.4.3) in the specified Endurance Group with the capacity specified by the Capacity Lower field and the Capacity Upper field. If the controller does not support NVM Sets, then this operation is not supported.	Endurance Group Identifier of the Endurance Group in which the NVM Set is to be created. A value of 0h specifies that the controller selects the Endurance Group in which the NVM Set is created.	4h	<b>Delete NVM Set:</b> The NVM Set specified by this command is deleted (refer to section 8.1.4.3). All namespaces in the NVM Set are deleted. If the controller does not support NVM Sets, then this operation is not supported.	NVM Set identifier of the NVM Set to be deleted.	5h to Fh	Reserved	
	Value	Description	Element Identifier Field																			
	0h	<b>Select Capacity Configuration:</b> Endurance Groups and NVM Sets are configured as indicated by the Capacity Configuration Descriptor specified by this command (refer to section 5.1.3.1).	Capacity Configuration Identifier (refer to Figure 257).																			
	1h	<b>Create Endurance Group:</b> An Endurance Group is created (refer to section 8.1.4.3) with the capacity specified by the Capacity Lower field (refer to Figure 157) and the Capacity Upper field (refer to Figure 158).	Domain Identifier (refer to section 3.2.5.3) of the domain in which the Endurance Group is to be created. A value of 0h specifies that the controller selects the domain in which the Endurance Group is created.																			
	2h	<b>Delete Endurance Group:</b> The Endurance Group specified by this command is deleted (refer to section 8.1.4.3). All namespaces and NVM Sets contained by the Endurance Group are deleted.	Endurance Group Identifier of the Endurance Group to be deleted.																			
	3h	<b>Create NVM Set:</b> An NVM Set is created (refer to section 8.1.4.3) in the specified Endurance Group with the capacity specified by the Capacity Lower field and the Capacity Upper field. If the controller does not support NVM Sets, then this operation is not supported.	Endurance Group Identifier of the Endurance Group in which the NVM Set is to be created. A value of 0h specifies that the controller selects the Endurance Group in which the NVM Set is created.																			
4h	<b>Delete NVM Set:</b> The NVM Set specified by this command is deleted (refer to section 8.1.4.3). All namespaces in the NVM Set are deleted. If the controller does not support NVM Sets, then this operation is not supported.	NVM Set identifier of the NVM Set to be deleted.																				
5h to Fh	Reserved																					
Notes:																						
1. If the Element Identifier field specifies a non-zero value which does not correspond to an existing capacity entity, then the controller shall abort the command with a status code of Invalid Field in Command.																						

**Figure 157: Capacity Management – Command Dword 11**

Bits	Description
31:00	<b>Capacity Lower (CAPL):</b> This field specifies the least significant 32 bits of the capacity in bytes of the Endurance Group or NVM Set to be created.

**Figure 158: Capacity Management – Command Dword 12**

Bits	Description
31:00	<b>Capacity Upper (CAPU):</b> This field specifies the most significant 32 bits of the capacity in bytes of the Endurance Group or NVM Set to be created.

If the Operation field specifies the Create Endurance Group operation or the Create NVM Set operation, then the Capacity Upper and Capacity Lower fields specify the capacity of the Endurance Group or NVM Set to be created. If the Operation field specifies any other operation, then the Capacity Upper field and the Capacity Lower field are reserved.

### 5.1.3.1 Media Unit Configuration Selection

If:

- a) the Operation field specifies the Select Capacity Configuration operation;
- b) the Element Identifier field specifies a Capacity Configuration Descriptor in the Supported Capacity Configuration List; and
- c) the Selected Configuration field of the Media Unit Status log page (refer to Figure 254) is cleared to 0h,

then the controller shall perform all of the following actions in sequence:

- 1) Create an Endurance Group for each Endurance Group Configuration Descriptor in the selected Capacity Configuration Descriptor.
- 2) Create an NVM Set for each NVM Set Identifier specified in each Endurance Group Configuration Descriptor, if any.

If the Operation field specifies the Select Capacity Configuration operation and the Element Identifier field is cleared to '0', then the controller shall clear the configuration by performing all of the following actions in sequence:

- 1) Delete all namespaces in the domain that contains the controller processing the command, as described in section 8.1.15.
- 2) Delete all NVM Sets in the domain that contains the controller processing the command, if any.
- 3) Delete all Endurance Groups in the domain that contains the controller processing the command.
- 4) Clear the Selected Configuration field to 0h in the Media Unit Status log page.

If the Operation field specifies the Select Capacity Configuration operation and the Element Identifier field specifies the value reported in the Selected Configuration field of the Media Unit Status log page (i.e., the currently-selected configuration), then the controller shall complete the command without error and shall make no change to the capacity configuration.

If the Operation field specifies the Select Capacity Configuration operation and:

- a) the Element Identifier field does not specify either a value of 0h or the Capacity Configuration Identifier of a Capacity Configuration Descriptor in the Capacity Configuration List; or
- b) the Selected Configuration field of the Media Unit Status log page (refer to Figure 254) is not cleared to 0h,

then the controller shall abort the command with a status code of Invalid Field in Command and no changes shall be made to the configuration of any Media Unit.

### 5.1.3.2 Endurance Group Operations

If the Operation field specifies the Create Endurance Group operation, the controller shall select a non-zero Endurance Group Identifier not assigned to an existing Endurance Group in the specified domain (refer to Figure 156) and indicate that value in Dword 0 of the completion queue entry (refer to Figure 160). If a non-zero unassigned Endurance Group Identifier is unavailable, then the controller shall abort the command with a status code of Identifier Unavailable.



If the Operation field specifies the Create Endurance Group operation and Media Units are supported, then the controller selects Media Units from the specified domain to be allocated to the Endurance Group.

If the Operation field specifies the Create Endurance Group operation and Media Units are not supported, then the controller selects NVM capacity from the specified domain to be allocated to the Endurance Group.

If the Operation field specifies the Create Endurance Group operation and the Capacity Lower and Capacity Upper fields specify a value that requires allocation of NVM capacity that is greater than the value in:

- a) the Max Endurance Group Capacity (MEGCAP) field in the Identify Controller data structure;
- b) the Unallocated NVM Capacity (UNVMCAP) field in the Identify Controller data structure; or
- c) the Max Endurance Group Capacity (MEGCAP) field in the Domain Attributes Entry for the domain in which the Endurance Group is being created,

then the controller:

- a) shall abort the command with Insufficient Capacity status; and
- b) if the Error Information log page is supported, shall indicate in the Command Specific Information field in the Error Information Log Entry data structure (refer to Figure 205) the total amount of NVM capacity in bytes of the largest Endurance Group that is able to be created.

If the Operation field specifies the Delete Endurance Group operation and the Element Identifier field specifies a value of 0h or the identifier of an Endurance Group that does not exist, then the controller shall abort the command with a status code of Invalid Field in Command.

### 5.1.3.3 NVM Set Operations

If the Operation field specifies the Create NVM Set operation, the controller shall select a non-zero NVM Set Identifier not assigned to an existing NVM Set in the specified Endurance Group (refer to Figure 156) and indicate that value in the completion queue entry. If a non-zero unassigned NVM Set Identifier is unavailable, then the controller shall abort the command with a status code of Identifier Unavailable.

If the Operation field specifies the Create NVM Set operation and the Capacity Lower and Capacity Upper fields specify a value that requires allocation of NVM capacity that is greater than the value in the Unallocated Endurance Group Capacity (UEGCAP) field in the Endurance Group Information log page (refer to Figure 218) for the specified Endurance Group, then the controller:

- a) if the Error Information log page is supported, shall indicate in the Command Specific Information field in the Error Information Log Entry data structure (refer to Figure 205) the total amount of NVM capacity in bytes of the largest NVM Set that is able to be created; and
- b) shall abort the command with Insufficient Capacity status.

If the Operation field specifies the Create NVM Set operation and the Element Identifier field is cleared to 0h, then the controller shall choose an existing Endurance Group in an existing domain in which to create the NVM Set.

If the Operation field specifies the Create NVM Set operation and Media Units are supported, then the controller selects Media Units from the Endurance Group to be allocated to the NVM Set.

If the Operation field specifies the Create NVM Set operation and Media Units are not supported, then the controller selects NVM capacity from the Endurance Group to be allocated to the NVM Set.

If the Operation field specifies the Delete NVM Set operation and the Element Identifier field specifies a value of 0h or the identifier of an NVM Set that does not exist, then the controller shall abort the command with a status code of Invalid Field in Command.

If the controller does not support NVM Sets and the Operation field specifies either the Create NVM Set operation or the Delete NVM Set operation, then the controller shall abort the command with a status code of Invalid Field in Command.

### 5.1.3.4 Command Completion

Upon completion of the Capacity Management command, the controller posts a completion queue entry to the Admin Completion Queue. Capacity Management command specific status values are defined in Figure 159.

**Figure 159: Capacity Management – Command Specific Status Values**

Value	Description
26h	<b>Insufficient Capacity:</b> The requested operation requires more free space than is currently available. The Command Specific Information field in the Error Information Log Entry data structure (refer to Figure 205) specifies the total amount of NVM capacity in bytes required to create the Endurance Group or NVM Set.
2Dh	<b>Identifier Unavailable:</b> The number of Endurance Groups or NVM Sets supported has been exceeded.

Dword 0 of the completion queue entry contains the identifier of the Endurance Group or NVM Set created, if any. Completion queue entry Dword 0 is defined in Figure 160.

**Figure 160: Capacity Management – Completion Queue Entry Dword 0**

Bits	Description
31:16	Reserved
15:00	<p><b>Created Element Identifier (CELID):</b> This field indicates the identifier of the NVM Set that was created if the Create NVM Set operation was used.</p> <p>This field indicates the identifier of the Endurance Group Identifier if the Create Endurance Group operation was used.</p> <p>This field is reserved for all other operations.</p>

### 5.1.4 Controller Data Queue command

The Controller Data Queue command is used to manage queues in host memory that a controller posts a specific type of data (refer to section 8.1.6).

The Controller Data Queue command uses the Command Dword 10 field. The use of the PRP1 field, Command Dword 11 field, and Command Dword 12 field is specific to the management operation specified by the Select field. All other command specific fields are reserved.

The Select field defined in Figure 161 determines which management operation is to be performed by this command and refer to section 5.1.4.1 for a description of each management operation.

**Figure 161: Controller Data Queue – Command Dword 10**

Bits	Description																
31:16	<b>Management Operation Specific (MOS):</b> The definition of this field is specific to a management operation (refer to the Select field). If a management operation does not define the use of this field, then this field is reserved.																
15:08	Reserved																
07:00	<b>Select (SEL):</b> This field specifies the type of management operation to perform.																
		<table border="1"> <thead> <tr> <th>Value</th> <th>Management Operation</th> <th>Reference Section</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Create Controller Data Queue</td> <td>5.1.4.1.1</td> </tr> <tr> <td>1h</td> <td>Delete Controller Data Queue</td> <td>5.1.4.1.2</td> </tr> <tr> <td>2h to BFh</td> <td>Reserved</td> <td></td> </tr> <tr> <td>C0h to FFh</td> <td>Vendor Specific</td> <td></td> </tr> </tbody> </table>	Value	Management Operation	Reference Section	0h	Create Controller Data Queue	5.1.4.1.1	1h	Delete Controller Data Queue	5.1.4.1.2	2h to BFh	Reserved		C0h to FFh	Vendor Specific	
		Value	Management Operation	Reference Section													
		0h	Create Controller Data Queue	5.1.4.1.1													
		1h	Delete Controller Data Queue	5.1.4.1.2													
2h to BFh	Reserved																
C0h to FFh	Vendor Specific																

### 5.1.4.1 Controller Data Queue Management Operations

#### 5.1.4.1.1 Create Controller Data Queue (Management Operation 0h)

The Create Controller Data Queue management operation of the Controller Data Queue command is used to create a queue in host memory to which a controller posts the information specified by the Queue Type field (refer to Figure 165).

The Create Controller Data Queue management operation uses the PRP Entry 1 field, Command Dword 11 field, and Command Dword 12 field.

If a PRP List is provided to describe the Controller Data Queue, then the PRP List shall be maintained by the host at the same location in host physical memory and the values in the PRP List shall not be modified until the Controller Data Queue is deleted (refer to section 5.1.4.1.2) or the controller is reset. If the PRP List values are modified, then the behavior is undefined.

If the number of memory ranges specified by the PRP Entry 1 field is greater than the value defined by the Maximum CDQ Memory Ranges (MCMR) field in the Identify Controller data structure (refer to Figure 312), then the controller shall abort the command with a status code of Invalid Field in Command.

If the number of memory ranges specified by the PRP Entry 1 field plus the number of memory ranges that exists for all of the Controller Data Queues in the NVM subsystem is greater than the value defined by the NVM Subsystem Maximum Controller CDQ Memory Ranges (NMCMR) field in the Identify Controller data structure (refer to Figure 312), then the controller shall abort the command with a status code of Invalid Field in Command.

If the current number of User Data Migration Queues that exist in the controller is equal to the value in the Maximum Controller User Data Migration Queues (MCUDMQ) field in the Identify Controller data structure, then the controller shall abort the command with a status code of Invalid Field in Command.

If the current number of User Data Migration Queues that exist in the NVM subsystem is equal to the value in the Maximum NVM Subsystem User Data Migration Queues (MNSUDMQ) field in the Identify Controller data structure, then the controller shall abort the command with a status code of Invalid Field in Command.

**Figure 162: Create Controller Data Queue – PRP Entry 1**

Bits	Description
63:00	<p><b>PRP Entry 1 (PRP1):</b> If the PC bit is set to '1', then this field specifies a 64-bit base host memory (refer to section 1.5.46) address pointer of the Controller Data Queue that is physically contiguous. The address pointer is memory page aligned (based on the value in CC.MPS field) unless otherwise specified. If the PC bit (refer to Figure 163) is cleared to '0', then this field specifies a PRP List pointer that describes the list of pages that constitute the Controller Data Queue. The list of pages is host memory (refer to section 1.5.46) that is page aligned (based on the value in CC.MPS field) unless otherwise specified. In both cases the PRP Entry shall have an offset of 0h. In a non-contiguous Controller Data Queue, each PRP Entry in the PRP List shall have an offset of 0h. If there is a PRP Entry with a non-zero offset, then the controller shall return an error of PRP Offset Invalid.</p> <p>If the memory specified by this field is not host memory (e.g., CMB or PMR), then the controller shall abort the command with a status code of Invalid Field in Command.</p>

**Figure 163: Create Controller Data Queue – Command Dword 11**

Bits	Description
31:16	<b>Create Queue Specific (CQS):</b> The definition of this field is specific to the type of queue being created (refer to Queue Type (QT) field in Figure 165). If the type of queue being created does not define the use of this field, then this field is reserved.
15:01	Reserved
00	<b>Physically Contiguous (PC):</b> If set to '1', then the Controller Data Queue is physically contiguous and PRP Entry 1 (PRP1) is the address of a contiguous physical buffer. If cleared to '0', then the Controller Data Queue is not physically contiguous and PRP Entry 1 (PRP1) is a PRP List pointer.

**Figure 164: Create Queue – Command Dword 12**

Bits	Description
31:00	<b>Controller Data Queue Size (CDQSIZE):</b> This field specifies the size of the queue to be created in dwords.

If the Controller Data Queue Size field value is not a multiple of the size of the entries for the type of Controller Data Queue as specified by the Queue Type field (refer to Figure 165), then the controller shall abort the command with a status code of Invalid Field in Command.

The Management Operation Specific field (refer to Figure 161) for the Create Controller Data Queue management operation is specified in Figure 165.

**Figure 165: Create Controller Data Queue – Management Operation Specific**

Bits	Description												
15:08	Reserved												
07:00	<p><b>Queue Type (QT):</b> This field specifies the type of queue to be created which defines the information that the controller posts into the entries of the queue.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> <th>Reference Section</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>User Data Migration Queue</td> <td>5.1.4.1.1.1</td> </tr> <tr> <td>1h to BFh</td> <td>Reserved</td> <td></td> </tr> <tr> <td>C0h to FFh</td> <td>Vendor Specific</td> <td></td> </tr> </tbody> </table>	Value	Definition	Reference Section	0h	User Data Migration Queue	5.1.4.1.1.1	1h to BFh	Reserved		C0h to FFh	Vendor Specific	
Value	Definition	Reference Section											
0h	User Data Migration Queue	5.1.4.1.1.1											
1h to BFh	Reserved												
C0h to FFh	Vendor Specific												

If the PRP Entry 1 field has a non-zero PRP offset, then the controller shall abort the command with a status code of PRP Offset Invalid.

#### 5.1.4.1.1.1 User Data Migration Queue (Queue Type 0h)

The User Data Migration Queue of the Controller Data Queue command is used to create a queue for logging of user data modified during the migration of the controller specified by the Controller Identifier field as defined by Figure 166.

If a User Data Migration Queue is associated with the same controller specified by the Controller Identifier field (refer to Figure 166), then the controller shall abort the command with a status code of Invalid Field in Command.

If the number of User Data Migration Queues that exist in the controller is equal to the value defined in the Maximum Controller User Data Migration Queues (MCUDMQ) field in the Identify Controller data structure (refer to Figure 312), then the controller shall abort this command with a status code of Not Enough Resources.

If the number of User Data Migration Queues that exist in the NVM subsystem is equal to the value defined in the Maximum NVM Subsystem User Data Migration Queues (MNSUDMQ) field in the Identify Controller data structure (refer to Figure 312), then the controller shall abort this command with a status code of Not Enough Resources.

**Figure 166: User Data Migration Queue – Create Queue Specific**

Bits	Description
15:00	<b>Controller Identifier (CNTLID):</b> This field specifies the controller associated with the information contained in the User Data Migration Queue.

Refer to the applicable I/O Command Set specification for the requirements and formats of the entries for the User Data Migration Queue.

#### 5.1.4.1.2 Delete Controller Data Queue (Management Operation 1h)

The Delete Controller Data Queue management operation of the Controller Data Queue command is used to delete a CDQ.

The Delete Controller Data Queue management operation uses the Command Dword 11 field.

**Figure 167: Delete Controller Data Queue – Command Dword 11**

Bits	Description
31:16	Reserved
15:00	<b>Controller Data Queue Identifier (CDQID):</b> This field specifies the identifier (refer to Figure 168) of the existing User Data Migration Queue to delete.

If the value in the Controller Data Queue Identifier field specifies a CDQ that does not exist in the controller processing the command, then the controller shall abort the command with a status code of Invalid Controller Data Queue.

#### 5.1.4.2 Command Completion

Upon completion of the Controller Data Queue command, the controller posts a completion queue entry to the Admin Completion Queue indicating the status for the command. Section 5.1.4.1 describes completion details for each management operation.

If a status code of Successful Completion is returned for the Create Controller Data Queue management operation, then Dword 0 of the completion queue entry contains the identifier of the Controller Data Queue created as defined in Figure 168. The indicated identifier is unique to the controller that processed the command.

**Figure 168: Controller Data Queue – Completion Queue Entry Dword 0**

Bits	Description
31:16	Reserved
15:00	<b>Controller Data Queue Identifier (CDQID):</b> This field indicates the identifier for the Controller Data Queue created (refer to section 8.1.6).

Controller Data Queue command specific status values (i.e., SCT field set to 1h) are shown in Figure 169.

**Figure 169: Controller Data Queue – Command Specific Status Values**

Value	Definition
1Fh	<b>Invalid Controller Identifier:</b> The Controller Identifier field contains an invalid value.
37h	<b>Invalid Controller Data Queue:</b> This error indicates that the specified Controller Data Queue Identifier is invalid for the controller processing the command.

### 5.1.5 Device Self-test command

The Device Self-test command is used to start a device self-test operation or abort a device self-test operation (refer to section 8.1.7). The Device Self-test command is used specifically to:

- a) start a short device self-test operation;
- b) start an extended device self-test operation;
- c) start a Host-Initiated Refresh operation;
- d) start a vendor specific device self-test operation; or
- e) abort a device self-test operation already in process.

A Host-Initiated Refresh operation is a device self-test operation.

The device self-test operation is performed by the controller that the Device Self-test command was submitted to. The Namespace Identifier field controls which namespaces are included in the device self-test operation as specified in Figure 170. For a Host-Initiated Refresh operation, the Namespace Identifier field shall be ignored by the controller.

**Figure 170: Device Self-test Namespace Test Action**

NSID Value	Description
00000000h	Specifies that the device self-test operation shall not include any namespaces, and only the controller is included as part of the device self-test operation.
00000001h to FFFFFFFEh	Specifies that the device self-test operation shall include the specified namespace. If this field specifies an invalid namespace ID, then the controller shall abort the command with a status code of Invalid Namespace or Format. If this field specifies an inactive namespace ID, then the controller shall abort the command with a status code of Invalid Field in Command.
FFFFFFFFh	Specifies that the device self-test operation shall include all attached namespaces accessible through the controller at the time the device self-test operation is started.

The Device Self-test command uses the Command Dword 10 field and the Command Dword 15 field. All other command specific fields are reserved.

**Figure 171: Device Self-test – Command Dword 10**

Bits	Description
31:04	Reserved

**Figure 171: Device Self-test – Command Dword 10**

Bits	Description																
03:00	<b>Self-test Code (STC):</b> This field specifies the action taken by the Device Self-test command.																
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Reserved</td> </tr> <tr> <td>1h</td> <td>Start a short device self-test operation</td> </tr> <tr> <td>2h</td> <td>Start an extended device self-test operation</td> </tr> <tr> <td>3h</td> <td>Start a Host-Initiated Refresh operation</td> </tr> <tr> <td>4h to Dh</td> <td>Reserved</td> </tr> <tr> <td>Eh</td> <td>Vendor specific. Additional information may be provided in Command Dword 15 (refer to Figure 172).</td> </tr> <tr> <td>Fh</td> <td>Abort device self-test operation</td> </tr> </tbody> </table>	Value	Definition	0h	Reserved	1h	Start a short device self-test operation	2h	Start an extended device self-test operation	3h	Start a Host-Initiated Refresh operation	4h to Dh	Reserved	Eh	Vendor specific. Additional information may be provided in Command Dword 15 (refer to Figure 172).	Fh	Abort device self-test operation
	Value	Definition															
	0h	Reserved															
	1h	Start a short device self-test operation															
	2h	Start an extended device self-test operation															
	3h	Start a Host-Initiated Refresh operation															
	4h to Dh	Reserved															
Eh	Vendor specific. Additional information may be provided in Command Dword 15 (refer to Figure 172).																
Fh	Abort device self-test operation																

**Figure 172: Device Self-test – Command Dword 15**

Bits	Description
31:00	<b>Device Self-test Parameter (DSTP):</b> If the Self-test Code field is set to Eh (refer to Figure 171), then this field is vendor specific; otherwise this field is reserved.

The processing of a Device Self-test command and interactions with a device self-test operation already in progress is defined in Figure 173.

**Figure 173: Device Self-test – Command Processing**

Self-test in Progress <sup>1</sup>	Self-test Code value in new Drive Self-test command	Controller Action
Yes	1h – Short device self-test	Abort the new Device Self-test command with a status code of Device Self-test in Progress.
	2h – Extended device self-test	
	3h – Host-Initiated Refresh	
	Eh – Vendor specific	Vendor specific
Yes	Fh – Abort device self-test	The controller takes the following actions in order: <ol style="list-style-type: none"> <li>1. Abort device self-test operation in progress;</li> <li>2. Create log entry in the Newest Self-test Result Data Structure in the Device Self-test log page;</li> <li>3. Set the Device Self-test Result field of the current Device Self-test Status field in the Device Self-test log page to 0h; and</li> <li>4. Completes command successfully.</li> </ol>
	No	1h – Short device self-test

**Figure 173: Device Self-test – Command Processing**

Self-test in Progress <sup>1</sup>	Self-test Code value in new Drive Self-test command	Controller Action
	2h – Extended device self-test	The controller takes the following actions in order: <ol style="list-style-type: none"> <li>1. Validate the command parameters;</li> <li>2. Set the Device Self-test Result field of the current Device Self-test Status field in the Device Self-test log page to 2h;</li> <li>3. Start a device self-test operation; and</li> <li>4. Completes command successfully.</li> </ol>
	3h – Host-Initiated Refresh	The controller takes the following actions in order: <ol style="list-style-type: none"> <li>1. Validate the command parameters;</li> <li>2. Set the Device Self-test Result field of the current Device Self-test Status field in the Device Self-test log page to 3h;</li> <li>3. Start a Host-Initiated Refresh operation; and</li> </ol> Complete the command successfully.
	Eh – Vendor specific	Vendor specific
	Fh – Abort device self-test	Completes command successfully. The Device Self-test log page is not modified.
Notes:		
1. If the Single Device Self-test Operation (SDSO) bit is cleared to '0' in the Device Self-test Options (DSTO) of the Identify Controller data structure (refer to Figure 312), then the Self-test in Progress column represents that a device self-test operation is in progress on the controller that the new Device Self-test command was received on. If the SDSO bit is set to '1', then the Self-test in Progress column represents that a device self-test operation is in progress on the NVM subsystem.		

### 5.1.5.1 Command Completion

A completion queue entry is posted to the Admin Completion Queue after the appropriate actions are taken as specified in Figure 173. Device Self-test command specific status values are defined in Figure 174.

**Figure 174: Device Self-test – Command Specific Status Values**

Value	Description
1Dh	<b>Device Self-test in Progress:</b> The controller or NVM subsystem already has a device self-test operation in process.

### 5.1.6 Directive Receive command

The Directive Receive command returns a data buffer that is dependent on the Directive Type. Refer to section 8.1.8.

The Directive Receive command uses the Data Pointer, Command Dword 10, and Command Dword 11 fields. Command Dword 12 and Dword 13 may be used based on the Directive Type field and the Directive Operation field. All other command specific fields are reserved.

If the Number of Dwords (NUMD) field corresponds to a length that is less than the size of the data structure to be returned, then only that specified portion of the data structure is transferred. If the NUMD field corresponds to a length that is greater than the size of the associated data structure, then the entire contents of the data structure are transferred and no additional data is transferred.

**Figure 175: Directive Receive – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field.



**Figure 176: Directive Receive – Command Dword 10**

Bits	Description
31:00	<b>Number of Dwords (NUMD):</b> This field specifies the number of dwords to transfer. This is a 0's based value.

**Figure 177: Directive Receive – Command Dword 11**

Bits	Description
31:16	<b>Directive Specific (DSPEC):</b> The interpretation of this field is Directive Type dependent. Refer to section 8.1.8.
15:08	<b>Directive Type (DTYPE):</b> This field specifies the Directive Type. Refer to Figure 597 for the list of Directive Types.
07:00	<b>Directive Operation (DOPER):</b> This field specifies the Directive Operation to perform. The interpretation of this field is Directive Type dependent. Refer to section 8.1.8.

### 5.1.6.1 Command Completion

When the command is completed, the controller posts a completion queue entry to the Admin Completion Queue indicating the status for the command. Command specific status values that may be returned are dependent on the Directive Type, refer to section 8.1.8.

### 5.1.7 Directive Send command

The Directive Send command transfers a data buffer that is dependent on the Directive Type to the controller. Refer to section 8.1.8.

The Directive Send command uses the Data Pointer, Command Dword 10, and Command Dword 11 fields. Command Dword 12 and Command Dword 13 may be used based on the Directive Type field and the Directive Operation field. All other command specific fields are reserved.

**Figure 178: Directive Send – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field.

**Figure 179: Directive Send – Command Dword 10**

Bits	Description
31:00	<b>Number of Dwords (NUMD):</b> This field specifies the number of dwords to transfer. This is a 0's based value.

**Figure 180: Directive Send – Command Dword 11**

Bits	Description
31:16	<b>Directive Specific (DSPEC):</b> The interpretation of this field is Directive Type dependent. Refer to section 8.1.8.
15:08	<b>Directive Type (DTYPE):</b> This field specifies the Directive Type. Refer to Figure 597 for the list of Directive Types.
07:00	<b>Directive Operation (DOPER):</b> This field specifies the Directive Operation to perform. The interpretation of this field is Directive Type dependent. Refer to section 8.1.8.

### 5.1.7.1 Command Completion

When the command is completed, the controller posts a completion queue entry to the Admin Completion Queue indicating the status for the command. Command specific status values that may be returned are dependent on the Directive Type, refer to section 8.1.8.

### 5.1.8 Firmware Commit command

Note: This command was known in NVM Express Base Specification revisions prior to revision 1.2 as “Firmware Activate.”

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

When modifying a firmware image, the Firmware Commit command verifies that a valid firmware image has been downloaded and commits that revision to a specific firmware slot. The host may select the firmware image to activate on the next Controller Level Reset as part of this command. The host may determine the currently executing firmware revision by examining the Firmware Revision field in the Identify Controller data structure in Figure 312. The host may determine the firmware revision to be executed on the next Controller Level Reset by examining the Firmware Slot Information log page. All controllers in a domain share firmware slots and the same firmware image is applied to all controllers in that domain (i.e., all the controllers in the NVM subsystem if multiple domains are not supported or all the controllers in that domain if multiple domains are supported).

Activation of a firmware image may result in a change in controller behavior that is not expected by the host (e.g., an incompatible change in the UUID List (refer to section 8.1.28.2)). In this case, if the Commit Action field is set to 011b, then the controller shall abort the command with a status code of Firmware Activation Requires Conventional Reset.

When modifying Boot Partitions, the host may select the Boot Partition to mark as active or to replace. A Boot Partition is only able to be written when write unlocked (refer to section 8.1.3).

The Firmware Commit command uses the Command Dword 10 field. All other command specific fields are reserved.

**Figure 181: Firmware Commit – Command Dword 10**

Bits	Description																
31	<b>Boot Partition ID (BPID):</b> Specifies the Boot Partition that shall be used for the Commit Action, if applicable.																
30:06	Reserved																
05:03	<b>Commit Action (CA):</b> This field specifies the action that is taken (refer to section 3.11) on the image downloaded with the Firmware Image Download command or on a previously downloaded and placed image. The actions are indicated in the following table.																
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>Downloaded image replaces the existing image, if any, in the specified Firmware Slot. The newly placed image is not activated.</td> </tr> <tr> <td>001b</td> <td>Downloaded image replaces the existing image, if any, in the specified Firmware Slot. The newly placed image is activated at the next Controller Level Reset.</td> </tr> <tr> <td>010b</td> <td>The existing image in the specified Firmware Slot is activated at the next Controller Level Reset.</td> </tr> <tr> <td>011b</td> <td>Downloaded image replaces the existing image, if any, in the specified Firmware Slot and is then activated immediately. If there is not a newly downloaded image, then the existing image in the specified firmware slot is activated immediately. The Firmware Commit command remains in progress until image activation has completed successfully or unsuccessfully (i.e., the Firmware Commit command is not a background operation).</td> </tr> <tr> <td>100b to 101b</td> <td>Reserved</td> </tr> <tr> <td>110b</td> <td>Downloaded image replaces the Boot Partition specified by the Boot Partition ID field.</td> </tr> <tr> <td>111b</td> <td>Mark the Boot Partition specified in the BPID field as active and update BPINFO.ABPID.</td> </tr> </tbody> </table>	Value	Definition	000b	Downloaded image replaces the existing image, if any, in the specified Firmware Slot. The newly placed image is not activated.	001b	Downloaded image replaces the existing image, if any, in the specified Firmware Slot. The newly placed image is activated at the next Controller Level Reset.	010b	The existing image in the specified Firmware Slot is activated at the next Controller Level Reset.	011b	Downloaded image replaces the existing image, if any, in the specified Firmware Slot and is then activated immediately. If there is not a newly downloaded image, then the existing image in the specified firmware slot is activated immediately. The Firmware Commit command remains in progress until image activation has completed successfully or unsuccessfully (i.e., the Firmware Commit command is not a background operation).	100b to 101b	Reserved	110b	Downloaded image replaces the Boot Partition specified by the Boot Partition ID field.	111b	Mark the Boot Partition specified in the BPID field as active and update BPINFO.ABPID.
	Value	Definition															
	000b	Downloaded image replaces the existing image, if any, in the specified Firmware Slot. The newly placed image is not activated.															
	001b	Downloaded image replaces the existing image, if any, in the specified Firmware Slot. The newly placed image is activated at the next Controller Level Reset.															
	010b	The existing image in the specified Firmware Slot is activated at the next Controller Level Reset.															
	011b	Downloaded image replaces the existing image, if any, in the specified Firmware Slot and is then activated immediately. If there is not a newly downloaded image, then the existing image in the specified firmware slot is activated immediately. The Firmware Commit command remains in progress until image activation has completed successfully or unsuccessfully (i.e., the Firmware Commit command is not a background operation).															
	100b to 101b	Reserved															
110b	Downloaded image replaces the Boot Partition specified by the Boot Partition ID field.																
111b	Mark the Boot Partition specified in the BPID field as active and update BPINFO.ABPID.																

**Figure 181: Firmware Commit – Command Dword 10**

Bits	Description
02:00	<b>Firmware Slot (FS):</b> Specifies the firmware slot that shall be used for the Commit Action, if applicable. If the value specified is 0h, then the controller shall choose the firmware slot (i.e., slot 1 to slot 7) to use for the operation.

**5.1.8.1 Command Completion**

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

For Firmware Commit commands that specify activation of a new firmware image at the next Controller Level Reset (i.e., the CA field was set to 001b or 010b) and complete with a status code value of 0h (i.e., Success Completion), a Controller Level Reset initiated by any of the methods defined in section 3.7.2 activates the specified firmware.

If the controller detects overlapping firmware/boot partition image update command sequences (refer to section 1.5.41) of more than one firmware image and/or Boot Partition or the use of more than one controller and/or Management Endpoint to update a single firmware image, then the results of that detection are reported in Dword 0 of the completion queue entry as defined in Figure 182. Refer to section 3.11 and section 8.1.3.2.

If:

- a) the Commit Action field is cleared to 000b (i.e., update the firmware image in the specified firmware slot but do not activate);
- b) the Firmware Slot field is set to the active firmware slot (refer to Figure 208); and
- c) the controller supports greater than one firmware slot for a host to store firmware images,

then the controller may abort the command with a status code of Command Sequence Error. A host may avoid this result by using a different firmware slot.

**Figure 182: Firmware Commit – Completion Queue Entry Dword 0**

Bits	Description						
31:02	Reserved						
01:00	<b>Multiple Update Detected (MUD):</b> This field indicates if a controller detected overlapping firmware/boot partition image update command sequences of Boot Partitions and/or firmware images (refer to section 3.11 and section 8.1.3.2). If the SMUD bit in the Firmware Update field of the Identify Controller data structure is cleared to '0', then this field shall be cleared to 00b.  This field is valid if the command is successful or aborted.						
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td><b>Management Endpoint FW Overlap (MEFWO):</b> If this bit is set to '1', then the controller detected an overlapping firmware/boot partition image update command sequence due to processing a command from a Management Endpoint. If this bit is cleared to '0', then the controller did not detect an overlapping firmware/boot partition image update command sequence due to processing a command from a Management Endpoint.</td> </tr> <tr> <td>0</td> <td><b>Admin Submission Queue FW Overlap (ASQFWO):</b> If this bit is set to '1', then the controller detected an overlapping firmware/boot partition image update command sequence due to processing a command from an Admin Submission Queue on a controller. If this bit is cleared to '0', then the controller did not detect an overlapping firmware/boot partition image update command sequence due to processing a command from an Admin Submission Queue on a controller.</td> </tr> </tbody> </table>	Bits	Description	1	<b>Management Endpoint FW Overlap (MEFWO):</b> If this bit is set to '1', then the controller detected an overlapping firmware/boot partition image update command sequence due to processing a command from a Management Endpoint. If this bit is cleared to '0', then the controller did not detect an overlapping firmware/boot partition image update command sequence due to processing a command from a Management Endpoint.	0	<b>Admin Submission Queue FW Overlap (ASQFWO):</b> If this bit is set to '1', then the controller detected an overlapping firmware/boot partition image update command sequence due to processing a command from an Admin Submission Queue on a controller. If this bit is cleared to '0', then the controller did not detect an overlapping firmware/boot partition image update command sequence due to processing a command from an Admin Submission Queue on a controller.
	Bits	Description					
1	<b>Management Endpoint FW Overlap (MEFWO):</b> If this bit is set to '1', then the controller detected an overlapping firmware/boot partition image update command sequence due to processing a command from a Management Endpoint. If this bit is cleared to '0', then the controller did not detect an overlapping firmware/boot partition image update command sequence due to processing a command from a Management Endpoint.						
0	<b>Admin Submission Queue FW Overlap (ASQFWO):</b> If this bit is set to '1', then the controller detected an overlapping firmware/boot partition image update command sequence due to processing a command from an Admin Submission Queue on a controller. If this bit is cleared to '0', then the controller did not detect an overlapping firmware/boot partition image update command sequence due to processing a command from an Admin Submission Queue on a controller.						

Firmware Commit command specific status values (i.e., SCT field set to 1h) are shown in Figure 183.

**Figure 183: Firmware Commit – Command Specific Status Values**

Value	Description
06h	<b>Invalid Firmware Slot:</b> The firmware slot indicated is invalid or read only. This error is indicated if the firmware slot exceeds the number supported.
07h	<b>Invalid Firmware Image:</b> The firmware image specified for activation is: <ul style="list-style-type: none"> <li>invalid and not loaded by the controller; or</li> <li>the specified firmware slot does not contain a firmware image.</li> </ul>
0Bh	<b>Firmware Activation Requires Conventional Reset:</b> 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 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.
10h	<b>Firmware Activation Requires NVM Subsystem Reset:</b> The firmware commit was successful; however, activation of the firmware image requires an NVM Subsystem Reset. If any other type of Controller Level Reset occurs prior to an NVM Subsystem Reset, the controller shall continue operation with the currently executing firmware image.
11h	<b>Firmware Activation Requires Controller Level Reset:</b> The firmware commit was successful; however, the firmware image specified does not support being activated without a Controller Level Reset. The firmware image shall be activated at the next Controller Level Reset. This status code should be returned only if the Commit Action field in the Firmware Commit command is set to 011b (i.e., activate immediately).
12h	<b>Firmware Activation Requires Maximum Time Violation:</b> The firmware commit was successful; however, the firmware image was not activated. Activation of the specified firmware image requires more time than the amount indicated in the Maximum Time for Firmware Activation (MTFA) field in the Identify Controller data structure (refer to Figure 312). To activate the firmware image committed to the specified firmware slot, a Firmware Commit command specifying a Commit Action set to 010b may be issued as described in section 3.11.
13h	<b>Firmware Activation Prohibited:</b> The image specified is being prohibited from activation by the controller for vendor specific reasons (e.g., controller does not support down revision firmware).
14h	<b>Overlapping Range:</b> This error is indicated if the controller detects that the firmware image has overlapping ranges.
1Eh	<b>Boot Partition Write Prohibited:</b> This error is indicated if a command attempts to modify a Boot Partition while write locked (refer to section 8.1.3.3).

### 5.1.9 Firmware Image Download command

The Firmware Image Download command is used to download all or a portion of an image for a future update to the controller. The Firmware Image Download command may be submitted while other commands on the Admin Submission Queue or I/O Submission Queues are outstanding. The Firmware Image Download command downloads a new image (in whole or in part) to the controller.

The image may be constructed of multiple pieces that are individually downloaded with separate Firmware Image Download commands. Each Firmware Image Download command includes a Dword Offset and Number of Dwords that specify a dword range. The host software should ensure that image pieces do not have dword ranges that overlap and that the NUMD field and OFST field meet the alignment and granularity requirements indicated in the FWUG field (refer to Figure 312). Firmware portions may be submitted out of order to the controller. Host software shall submit image portions in order when updating a Boot Partition. If ranges overlap, the controller may return an error of Overlapping Range.

The new firmware image is not activated as part of the Firmware Image Download command. Refer to section 3.11 for details on the firmware update process. The firmware update process does not modify the contents of Boot Partitions. Refer to section 8.1.3.2 for details on the Boot Partition update process.

Host software should not overlap command sequences that update Boot Partitions and/or firmware images (refer to section 3.11 and section 8.1.3.2).

After downloading an image, host software issues a Firmware Commit command before downloading another image. Processing of the first Firmware Image Download command after completion of a Firmware Commit command shall cause the controller to discard all remaining portion(s), if any, of downloaded

images. If a Controller Level Reset occurs between a firmware download and completion of the Firmware Commit command, then the controller shall discard all portion(s), if any, of downloaded images.

The Firmware Image Download command uses the Data Pointer, Command Dword 10, and Command Dword 11 fields. All other command specific fields are reserved.

**Figure 184: Firmware Image Download – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the location where data should be transferred from. Refer to Figure 92 for the definition of this field.

**Figure 185: Firmware Image Download – Command Dword 10**

Bits	Description
31:00	<b>Number of Dwords (NUMD):</b> This field specifies the number of dwords to transfer for this portion of the firmware. This is a 0's based value. If the value specified in this field does not meet the requirement indicated by the FWUG field (refer to Figure 312), then the controller may abort the command with a status code of Invalid Field in Command.

**Figure 186: Firmware Image Download – Command Dword 11**

Bits	Description
31:00	<b>Offset (OFST):</b> This field specifies the number of dwords offset from the start of the firmware image being downloaded to the controller. The offset is used to construct the complete firmware image when the firmware is downloaded in multiple pieces. The piece corresponding to the start of the firmware image shall have an Offset of 0h. If the value specified in this field does not meet the requirement indicated by the FWUG field (refer to Figure 312), then the controller may abort the command with a status code of Invalid Field in Command.

### 5.1.9.1 Command Completion

Upon completion of the Firmware Image Download command, the controller posts a completion queue entry to the Admin Completion Queue. Firmware Image Download command specific status values are defined in Figure 187.

**Figure 187: Firmware Image Download – Command Specific Status Values**

Value	Description
14h	<b>Overlapping Range:</b> This error is indicated if the controller detects that the firmware image has overlapping ranges. This error may indicate that the granularity or alignment of the firmware image downloaded does not conform to the Firmware Update Granularity field indicated in the Identify Controller data structure.

If the controller detects overlapping firmware/boot partition image update command sequences (refer to section 1.5.41) of more than one firmware image and/or Boot Partition or the use of more than one controller and/or Management Endpoint to update a single firmware image, then the results of that detection are reported in Dword 0 of the completion queue entry as defined in Figure 182. Refer to section 3.11 and section 8.1.3.2.

### 5.1.10 Format NVM command

The Format NVM command is used to low level format the NVM media. This command is used by the host to change the attributes of the NVM media (e.g., the LBA data size and/or metadata size for the NVM Command Set). A low level format may destroy all data and metadata associated with all namespaces that contain formatted storage or only the specific namespace associated with the command (refer to the Format NVM Attributes field in the Identify Controller data structure, Figure 312). After the Format NVM command successfully completes, the controller shall not return any user data that was previously contained in an affected namespace.

As part of the Format NVM command, the host requests a format operation and may request a secure erase of the contents of the NVM (refer to the SES field in Figure 189). There are two types of secure erase. The User Data Erase erases all user content present in the NVM subsystem. The Cryptographic Erase erases all user content present in the NVM subsystem by deleting the encryption key with which the user data was previously encrypted.

The scope of the format operation and the scope of the format with secure erase depend on the attributes that the controller supports for the Format NVM command and the Namespace Identifier specified in the command as described in Figure 188. The type of secure erase, if applicable, is based on the setting of the Secure Erase Settings field in Command Dword 10 as defined in Figure 189.

The scope of the entire Format NVM command is determined by the value of the SES field (refer to Figure 188) and the setting of either the Format Namespace Scope (FNS) bit or the Secure Erase Namespace Scope (SENS) bit (refer to Figure 312) as follows:

- a) If the Format NVM command does not specify Secure Erase (i.e., the SES field is cleared to 000b), then the scope of the Format NVM command is indicated by the value of FNS bit and the value of SENS bit is not applicable to this command.
- b) If the Format NVM command specifies Secure Erase (i.e., the SES field is set to a non-zero value), then the scope of the Format NVM command is indicated by the value of SENS bit and the value of FNS bit is not applicable to this command.

**Figure 188: Format NVM – Operation Scope**

SES	SENS Bit	FNS Bit	NSID	Format Operation
000b (i.e., not a secure erase)	0	N/A	FFFFFFFFh <sup>1</sup>	All namespaces attached to the controller. Other namespaces are not affected.
	0		Any allocated value (refer to section 3.2.1.3)	The specified namespace. Other namespaces are not affected.
	1		Any allocated value (refer to section 3.2.1.3) or FFFFFFFFFh	All namespaces that exist in the NVM subsystem.
001b or 010b (i.e., secure erase)	N/A	0	FFFFFFFFh <sup>1</sup>	All namespaces attached to the controller. Other namespaces are not affected.
		0	Any allocated value (refer to section 3.2.1.3)	The specified namespace. Other namespaces are not affected.
		1	Any allocated value (refer to section 3.2.1.3) or FFFFFFFFFh	All namespaces that exist in the NVM subsystem.
All others				The controller shall abort the command with a status code of Invalid Field in Command
Notes:				
1. If the Format NVM Broadcast Support (FNVMBBS) bit in the FNA field is set to '1', then this value is not supported, and the command is aborted as described in this section.				

If the NVM subsystem supports multiple domains and the Format NVM command is not able to format the specified namespaces as a result of the NVM subsystem being divided (refer to section 3.2.5), then the Format NVM command shall be aborted with a status code of Asymmetric Access Inaccessible or Asymmetric Access Persistent Loss.

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 shall 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 shall be returned. Refer to section 5 for further information about restrictions on Admin commands during Format NVM.

For a Format NVM command with the NSID field set to FFFFFFFFh that specifies secure erase:

- a) if the Secure Erase Namespace Scope (SENS) bit is set to '1' in the FNA field (refer to Figure 312) and there are no namespaces in the NVM subsystem, then that Format NVM command shall complete without error; and
- b) if the SENS bit is cleared to '0' in the FNA field and there are no attached namespaces, then that Format NVM command shall complete without error.

For a Format NVM command with an NSID field set to FFFFFFFFh that does not specify a secure erase:

- a) if the Format Namespace Scope (FNS) bit is set to '1' in the FNA field and there are no namespaces in the NVM subsystem, then that Format NVM command shall complete without error; and
- b) if the FNS bit is cleared to '0' in the FNA field and there are no attached namespaces, then that Format NVM command shall complete without error.

If a host does not set the LBA Format Extension Enable (LBAFEE) field to 1h in the Host Behavior Support feature (refer to section 5.1.25.1.14), then the 0h value of the LBAFEE field disables any I/O Command Set specific format that requires the LBAFEE field to be set to 1h (refer to the applicable I/O Command Set specification). If a Format NVM command specifies a format that is disabled (e.g., the LBAFEE field is cleared to 0h), then the controller shall abort that Format NVM command with a status code of Invalid Namespace or Format.

If the format operation scope (refer to Figure 188) for a Format NVM command includes any namespace that is write protected (refer to section 8.1.16), then the controller aborts that Format NVM command with a status code of Namespace is Write Protected.

If the Format NVM Broadcast Support (FNVMB) bit in the FNA field is set to '1' and a Format NVM command has the NSID field set to FFFFFFFFh, then the controller shall abort the command with a status code of Invalid Field in Command.

After successful completion of a Format NVM command, the settings specified in the Format NVM command (e.g., PI, MSET, LBAF) are reported as part of the Identify Namespace data structures (refer to section 1.5.49).

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

**Figure 189: Format NVM – Command Dword 10**

Bits	Description
31:14	Reserved
13:12	<p><b>LBA Format Upper (LBAFU):</b> This field specifies the most significant 2 bits of the Format Index of the User Data Format to apply to the NVM media. This corresponds to the User Data Formats indicated in the Identify command, refer to the Identify Namespace data structure and the LBA Format data structure in the applicable I/O Command Set specification. If an unsupported User Data Format is selected, the controller shall abort the command with a status code of Invalid Format.</p> <p>This field shall be ignored by the controller if the LBA Format Extension Enable (LBAFEE) field is cleared to 0h in the Host Behavior Support feature (refer to section 5.1.25.1.14).</p> <p>NOTE: This field applies to all User Data Formats. The original name has been retained for historical continuity.</p>

**Figure 189: Format NVM – Command Dword 10**

Bits	Description										
11:09	<b>Secure Erase Settings (SES):</b> This field specifies whether a secure erase should be performed as part of the format and the type of the secure erase operation. The erase applies to all user data, regardless of location (e.g., within an exposed LBA (refer to the NVM Express NVM Command Set Specification), within a cache, within deallocated logical blocks, etc.).										
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>No secure erase operation requested</td> </tr> <tr> <td>001b</td> <td><b>User Data Erase:</b> All user data shall be erased, contents of the media allocated for user data after the erase is indeterminate (e.g., the user data may be zero filled, one filled, etc.). If a User Data Erase is requested and all affected user data is encrypted, then the controller is allowed to use a cryptographic erase to perform the requested User Data Erase.</td> </tr> <tr> <td>010b</td> <td><b>Cryptographic Erase:</b> All user data shall be erased cryptographically. This is accomplished by deleting the encryption key.</td> </tr> <tr> <td>011b to 111b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	000b	No secure erase operation requested	001b	<b>User Data Erase:</b> All user data shall be erased, contents of the media allocated for user data after the erase is indeterminate (e.g., the user data may be zero filled, one filled, etc.). If a User Data Erase is requested and all affected user data is encrypted, then the controller is allowed to use a cryptographic erase to perform the requested User Data Erase.	010b	<b>Cryptographic Erase:</b> All user data shall be erased cryptographically. This is accomplished by deleting the encryption key.	011b to 111b	Reserved
	Value	Definition									
	000b	No secure erase operation requested									
001b	<b>User Data Erase:</b> All user data shall be erased, contents of the media allocated for user data after the erase is indeterminate (e.g., the user data may be zero filled, one filled, etc.). If a User Data Erase is requested and all affected user data is encrypted, then the controller is allowed to use a cryptographic erase to perform the requested User Data Erase.										
010b	<b>Cryptographic Erase:</b> All user data shall be erased cryptographically. This is accomplished by deleting the encryption key.										
011b to 111b	Reserved										
08	<b>Protection Information Location (PIL):</b> I/O Command Set specific definition. <sup>1</sup>										
07:05	<b>Protection Information (PI):</b> I/O Command Set specific definition. <sup>1</sup>										
04	<b>Metadata Settings (MSET):</b> I/O Command Set specific definition. <sup>1</sup>										
03:00	<b>LBA Format Lower (LBAFL):</b> This field specifies the least significant 4 bits of the Format Index of the User Data Format to apply to the NVM media. This corresponds to the User Data Formats indicated in the Identify Namespace data structure, refer to the Identify Namespace data structure and the I/O Command Set specific Format data structure in the applicable I/O Command Set specification. If an unsupported User Data Format is selected, the controller shall abort the command with a status code of Invalid Format.  Note: This field applies to all User Data Formats. The original name has been retained for historical continuity.										
Note: 1. I/O Command Set specific fields are described in the applicable I/O Command Set specification.											

### 5.1.10.1 Command Completion

A completion queue entry is posted to the Admin Completion Queue when the NVM media format is complete. Format NVM command specific status values (i.e., SCT field set to 1h) are shown in Figure 190.

**Figure 190: Format NVM – Command Specific Status Values**

Value	Description
0Ah	<b>Invalid Format:</b> The format specified is invalid. This may be due to various conditions, including: <ul style="list-style-type: none"> <li>specifying an invalid User Data Format number;</li> <li>enabling protection information when there are not sufficient metadata resources; or</li> <li>the specified format is not available in the current configuration.</li> </ul>

### 5.1.11 Get Features command

The Get Features command retrieves the attributes of the Feature specified.

The Get Features command uses the Data Pointer, Command Dword 10 and Command Dword 14 fields. The use of the Command Dword 11 field is Feature specific. If not used by a Feature, then Command Dword 11 is reserved unless otherwise stated. All other command specific fields are reserved.

The mandatory, optional, and prohibited Feature Identifiers for each type of controller are defined in section 3.1.3.6.



**Figure 191: Get Features – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field. If no data structure is used as part of the specified feature (refer to the FID field), then this field shall be ignored by the controller.

**Figure 192: Get Features – Command Dword 10**

Bits	Description												
31:11	Reserved												
10:08	<p><b>Select (SEL):</b> This field specifies the attribute of the value requested in the returned data:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Select</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>Current</td> </tr> <tr> <td>001b</td> <td>Default</td> </tr> <tr> <td>010b</td> <td>Saved</td> </tr> <tr> <td>011b</td> <td>Supported Capabilities</td> </tr> <tr> <td>100b to 111b</td> <td>Reserved</td> </tr> </tbody> </table> <p>Refer to section 5.1.11.1 and section 4.4 for details on the data returned in each case.</p> <p>The controller indicates in the Save and Select Feature Support (SSFS) bit of the Optional NVM Command Support field of the Identify Controller data structure in Figure 312 whether this field is supported.</p> <p>If a Get Features command is received with the Select field set to 010b (i.e., saved) and the controller does not support the Feature Identifier being saved or does not currently have any saved values, then the controller shall operate as if the Select field is set to 001b (i.e., default).</p>	Select	Description	000b	Current	001b	Default	010b	Saved	011b	Supported Capabilities	100b to 111b	Reserved
Select	Description												
000b	Current												
001b	Default												
010b	Saved												
011b	Supported Capabilities												
100b to 111b	Reserved												
07:00	<b>Feature Identifier (FID):</b> This field specifies the identifier of the Feature for which to provide data.												

If the controller supports selection of a UUID by the Get Features command (refer to Figure 385 and section 8.1.28) and the controller supports selection of a UUID for the specified vendor specific Feature Identifier (refer to Figure 385), then Command Dword 14 is used to specify a UUID Index value (refer to Figure 193). If the controller does not support selection of a UUID by the Get Features command or the controller does not support selection of a UUID for the specified vendor specific Feature Identifier, then Command Dword 14 does not specify a UUID Index value.

**Figure 193: Get Features – Command Dword 14**

Bits	Description
31:07	Reserved
06:00	<b>UUID Index (UIDX):</b> Refer to Figure 658.

Figure 194 describes the Feature Identifiers whose attributes may be retrieved using the Get Features command. The definition of the attributes returned and the associated format is specified in the section indicated.

**Figure 194: Get Features – Feature Identifiers**

Description	Feature Identifier	Section Defining Format of Attributes Returned
Arbitration	01h	5.1.25.1.1
Power Management	02h	5.1.25.1.2
Temperature Threshold	04h	5.1.25.1.3
Volatile Write Cache	06h	5.1.25.1.4
Number of Queues	07h	5.1.25.2.1

Figure 194: Get Features – Feature Identifiers

Description	Feature Identifier	Section Defining Format of Attributes Returned
Interrupt Coalescing	08h	5.1.25.2.2
Interrupt Vector Configuration	09h	5.1.25.2.3
Asynchronous Event Configuration	0Bh	5.1.25.1.5
Autonomous Power State Transition	0Ch	5.1.25.1.6
Host Memory Buffer	0Dh	5.1.25.2.4
Timestamp	0Eh	5.1.25.1.7
Keep Alive Timer	0Fh	5.1.25.1.8
Host Controlled Thermal Management	10h	5.1.25.1.9
Non-Operational Power State Config	11h	5.1.25.1.10
Read Recovery Level Config	12h	5.1.25.1.11
Predictable Latency Mode Config	13h	5.1.25.1.12
Predictable Latency Mode Window	14h	5.1.25.1.13
Host Behavior Support	16h	5.1.25.1.14
Sanitize Config	17h	5.1.25.1.15
Endurance Group Event Configuration	18h	5.1.25.1.16
I/O Command Set Profile	19h	5.1.25.1.17
Spinup Control	1Ah	5.1.25.1.18
Power Loss Signaling Config	1Bh	5.1.25.1.19
Flexible Data Placement	1Dh	5.1.25.1.20
Flexible Data Placement Events	1Eh	5.1.25.1.21
Namespace Admin Label	1Fh	5.1.25.1.22
Controller Data Queue	21h	5.1.25.1.23
Embedded Management Controller Address	78h	5.1.25.1.24
Host Management Agent Address	79h	5.1.25.1.25
Enhanced Controller Metadata <sup>1</sup>	7Dh	5.1.25.1.26.1
Controller Metadata <sup>1</sup>	7Eh	5.1.25.1.26.2
Namespace Metadata <sup>1</sup>	7Fh	5.1.25.1.26.3
Software Progress Marker	80h	5.1.25.1.27
Host Identifier	81h	5.1.25.1.28
Reservation Notification Mask	82h	5.1.25.1.29
Reservation Persistence	83h	5.1.25.1.30
Namespace Write Protection Config	84h	5.1.25.1.31
Boot Partition Write Protection Config	85h	5.1.25.1.32
I/O Command Set specific features		I/O Command Set specification
Notes:		
1. Information that is common to all Host Metadata features is described in section 5.1.25.1.26.		

#### 5.1.11.1 Select field

A Select field cleared to 000b (i.e., current) returns the current operating attribute value for the Feature Identifier specified.

A Select field set to 001b (i.e., default) returns the default attribute value for the Feature Identifier specified.

A Select field set to 010b (i.e., saved) returns the last saved attribute value for the Feature Identifier specified (i.e., the last Set Features command completed without error, with the Save bit set to '1' for the Feature Identifier specified).

A Select field set to 011b (i.e., supported capabilities) returns the capabilities supported for this Feature Identifier. The capabilities supported are returned in Dword 0 of the completion queue entry of the Get Features command (refer to Figure 195).

### 5.1.11.2 Command Completion

Upon completion of the Get Features command, the controller posts a completion queue entry to the Admin Completion Queue. If the Select field is not set to 011b, then Dword 0 of the completion queue entry may contain feature-dependent information (refer to section 5.1.25).

If the Select field is set to 011b, then Figure 195 describes the contents of Dword 0 of the completion queue entry.

**Figure 195: Completion Queue Entry Dword 0 when Select is set to 11b**

Bits	Description
31:3	Reserved
2	<b>Changeable (CHANG):</b> If this bit is set to '1', then the feature values are changeable. If this bit is cleared to '0', then the feature values are not changeable. Changing a feature value is described in section 5.1.25.
1	<b>NS Specific (NSSPEC):</b> If this bit is set to '1', then the Feature Identifier has a namespace scope. If this bit is cleared to '0', then the Feature Identifier has a scope as defined in Figure 385.
0	<b>Saveable (SVBL):</b> If this bit is set to '1', then the feature values are saveable. If this bit is cleared to '0', then the feature values are not saveable.

If the controller supports any changeable value of any attribute of a feature, then the controller reports that feature as changeable (i.e., the controller sets the Changeable bit to '1' in Completion Queue Entry Dword 0 for a Get Features command that specifies that feature and has the Select field set to 011b), even if the feature has been set to a value or values that are not changeable.

For some features (e.g., Namespace Write Protection Config (refer to section 5.1.25.1.31), Boot Partition Write Protection Config (refer to section 5.1.25.1.32)), the changeability of feature values is value-dependent (e.g., the Permanent Write Protect (i.e., 011b) value of the Write Protection State in the Namespace Write Protection Config feature is not changeable).

### 5.1.12 Get Log Page command

The Get Log Page command returns a data buffer containing the log page requested. The Get Log Page command may be impacted by the ANA state (refer to section 8.1.1.10).

The Get Log Page command uses the Data Pointer, Command Dword 10, Command Dword 11, Command Dword 12, Command Dword 13, and Command Dword 14 fields. All other command specific fields are reserved.

There are mandatory and optional Log Page Identifiers defined in section 3.1.3.5. If a Get Log Page command is processed that specifies a Log Page Identifier that is not supported, then the controller shall abort the command with a status code of Invalid Log Page with the exception defined in Figure 305.

The controller indicates support for the Log Page Offset and extended Number of Dwords (32 bits rather than 12 bits) in the Log Page Attributes field of the Identify Controller data structure. If extended data is not supported, then bits 27:16 of the Number of Dwords Lower field specify the Number of Dwords to transfer.

If the Log Page Offset is supported, then

- a byte offset shall be supported (i.e., Offset Type field cleared to '0') for all log pages; and
- for each log page that has the IOS bit set to '1' in the LID Supported and Effects data structure (refer to Figure 204) for the specified LID in the Supported Log Pages log page an index offset shall be supported (i.e., Offset Type field set to '1').

If the IOS bit is cleared to '0' in the LID Supported and Effects data structure for the specified LID in the Supported Log Pages log page and a Get Log Page command specifies Offset Type field set to '1', then that command shall be aborted with a status code of Invalid Field in Command.

**Figure 196: Get Log Page – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field.

**Figure 197: Get Log Page – Command Dword 10**

Bits	Description
31:16	<b>Number of Dwords Lower (NUMDL):</b> This field specifies the least significant 16 bits of the number of dwords to return unless otherwise specified. If host software specifies a size larger than the log page requested, the controller returns the complete log page with undefined results for dwords beyond the end of the log page. The combined NUMDL and NUMDU fields form a 0's based value.
15	<p><b>Retain Asynchronous Event (RAE):</b> This bit specifies whether to retain or clear an Asynchronous Event. If this bit is cleared to '0', then the corresponding Asynchronous Event is cleared by the controller upon successful command completion. If this bit is set to '1', then the corresponding Asynchronous Event is retained (i.e., not cleared) by the controller upon command completion.</p> <p>If the command does not complete successfully, the Asynchronous Event shall be retained by the controller.</p> <p>Host software should clear this bit to '0' for log pages that are not used with Asynchronous Events. Refer to section 5.1.2.</p> <p>Refer to the NVM Express Admin Command Set section of the NVM Express Management Interface Specification for the behavior associated with this bit when the Get Log Page command is received on a Management Endpoint.</p>
14:08	<b>Log Specific Parameter (LSP):</b> If not defined for the log specified by the Log Page Identifier field, this field is reserved.
07:00	<b>Log Page Identifier (LID):</b> This field specifies the identifier of the log page to retrieve (refer to Figure 202).

**Figure 198: Get Log Page – Command Dword 11**

Bits	Description
31:16	<b>Log Specific Identifier (LSI):</b> This field specifies an identifier that is required for a particular log page. If not defined for the log page specified by the Log Page Identifier field, this field is reserved.
15:00	<b>Number of Dwords (NUMDU):</b> This field specifies the most significant 16 bits of the number of dwords to return unless otherwise specified.

**Figure 199: Get Log Page – Command Dword 12**

Bits	Description
31:00	<p><b>Log Page Offset Lower (LPOL):</b> The log page offset specifies the location within a log page to start returning data from unless otherwise specified.</p> <p>If the OT bit is cleared to '0', then:</p> <ul style="list-style-type: none"> <li>a) This field specifies the least significant 32 bits of the log page offset. The offset shall be dword aligned, indicated by bits 1:0 being cleared to 00b;</li> <li>b) The controller is not required to check that bits 1:0 are cleared to 00b. The controller may report an error of Invalid Field in Command if bits 1:0 are not cleared to 00b. If the controller does not report an error of Invalid Field in Command, then the controller shall operate as if bits 1:0 are cleared to 00b; and</li> <li>c) If the host specifies an offset (i.e., LPOL and LPOU) that is greater than the size of the log page requested (e.g., a log page containing 100 bytes is requested starting at offset 200), then the controller shall abort the command with a status of Invalid Field in Command.</li> </ul> <p>If the OT bit is set to '1', then:</p> <ul style="list-style-type: none"> <li>a) This field specifies the least significant 32 bits of the index into the list of data structures in the log page;</li> <li>b) If the host specifies an index (i.e., LPOL and LPOU) that is greater than the number of entries in the list of data structures in the log page requested (e.g., a log page containing 100 data structures is requested starting at index 200), then the controller shall abort the command with a status code of Invalid Field in Command;</li> <li>c) If the IOS bit is cleared to '0' in the LID Supported and Effects data structure for the specified LID in the Supported Log Pages log page, then the controller shall abort the command with a status code of Invalid Field in Command; and</li> <li>d) Each log page that supports the use of an index offset value defines the contents of an entry for the purposes of indexing into that log page.</li> </ul>

**Figure 200: Get Log Page – Command Dword 13**

Bits	Description
31:00	<p><b>Log Page Offset Upper (LPOU):</b> This field specifies the most significant 32 bits of either the log page offset or the index into the list of data structures unless otherwise specified. Refer to the Log Page Offset Lower definition.</p>

If the controller supports selection of a UUID by the Get Log Page command (refer to Figure 202 and section 8.1.28), then Command Dword 14 is used to specify a UUID Index value (refer to Figure 201).

**Figure 201: Get Log Page – Command Dword 14**

Bits	Description
31:24	<p><b>Command Set Identifier (CSI):</b> This field specifies the I/O Command Set (refer to Figure 311) to be used by the command. This field is log page specific; refer to Figure 202 for log pages that use this field. This field shall be ignored by the controller if the specified log page does not support the use of this field (refer to Figure 202) or if the CC.CSS field is not set to 110b.</p>
23	<p><b>Offset Type (OT):</b> If this bit is set to '1', then the Log Page Offset Lower field and the Log Page Offset Upper field specify the index into the list of data structures in the log page to be returned. If this bit is cleared to '0', then the Log Page Offset Lower field and the Log Page Offset Upper field specify the byte offset into the log page to be returned.</p>
22:07	Reserved
06:00	<p><b>UUID Index (UIDX):</b> Refer to Figure 658.</p>

Figure 202 defines the log pages that are able to be retrieved with the Get Log Page command and the scope of the information that is returned in those log pages. Section 5.1.12.1 describes log pages that are common to all transport models. Section 5.1.12.2 describes log pages that are specific to the Memory-based transport model. Section 5.1.12.3 describes log pages that are specific to the Message-based transport model.

Figure 202: Get Log Page – Log Page Identifiers

Log Page Identifier	CSI <sup>8</sup>	Scope	Log Page Name	Reference Section
00h	Y	Controller	Supported Log Pages	5.1.12.1.1
01h	N	Controller	Error Information	5.1.12.1.2
02h	N	Controller / Namespace <sup>1, 2</sup>	SMART / Health Information	5.1.12.1.3
03h	N	Domain / NVM subsystem <sup>6</sup>	Firmware Slot Information	5.1.12.1.4
04h	N	Controller	Changed Attached Namespace List	5.1.12.1.5
05h	Y	Controller	Commands Supported and Effects	5.1.12.1.6
06h	N	Controller / Domain / NVM subsystem <sup>3, 4, 6</sup>	Device Self-test <sup>5</sup>	5.1.12.1.7
07h	N	Controller / NVM subsystem <sup>7</sup>	Telemetry Host-Initiated <sup>5</sup>	5.1.12.1.8
08h	N	Controller / NVM subsystem <sup>7</sup>	Telemetry Controller-Initiated <sup>5</sup>	5.1.12.1.9
09h	N	Domain / NVM subsystem <sup>6</sup>	Endurance Group Information	5.1.12.1.10
0Ah	N	Domain / NVM subsystem <sup>6</sup>	Predictable Latency Per NVM Set	5.1.12.1.11
0Bh	N	Domain / NVM subsystem <sup>6</sup>	Predictable Latency Event Aggregate	5.1.12.1.12
0Ch	N	Controller	Asymmetric Namespace Access	5.1.12.1.13
0Dh	N	NVM subsystem	Persistent Event Log <sup>5</sup>	5.1.12.1.14
0Eh	Refer to the NVM Command Set Specification			
0Fh	N	Domain / NVM subsystem <sup>6</sup>	Endurance Group Event Aggregate	5.1.12.1.15
10h	N	Domain / NVM subsystem <sup>5, 6</sup>	Media Unit Status	5.1.12.1.16
11h	N	Domain / NVM subsystem <sup>6</sup>	Supported Capacity Configuration List	5.1.12.1.17
12h	Y	Controller	Feature Identifiers Supported and Effects	5.1.12.1.18
13h	N	Controller	NVMe-MI Commands Supported and Effects	5.1.12.1.19
14h	Y	NVM subsystem	Command and Feature Lockdown <sup>5</sup>	5.1.12.1.20
15h	N	Controller	Boot Partition	5.1.12.1.21
16h	N	Endurance Group	Rotational Media Information	5.1.12.1.22
17h	N	Namespace	Dispersed Namespace Participating NVM Subsystems	5.1.12.1.23
18h	N	NVM subsystem	Management Address List	5.1.12.1.24
19h	Refer to the applicable NVM Express transport specification			
1Ah	N	Controller	Reachability Groups	5.1.12.1.25
1Bh	N	Controller	Reachability Associations	5.1.12.1.26
1Ch	N	Controller	Changed Allocated Namespace List	5.1.12.1.27
1Dh to 1Fh	Reserved			
20h	N	Endurance Group	FDP Configurations	5.1.12.1.28
21h	N	Endurance Group	Reclaim Unit Handle Usage	5.1.12.1.29
22h	N	Endurance Group	FDP Statistics	5.1.12.1.30
23h	N	Endurance Group	FDP Events	5.1.12.1.31
24h to 6Fh	Reserved			
70h	N	Controller	Discovery	5.1.12.3.1
71h	N	Controller	Host Discovery	5.1.12.3.2
72h	N	Controller	AVE Discovery	5.1.12.3.3
73h	N	Controller	Pull Model DDC Request	5.1.12.3.4
74h to 7Fh	Reserved			
80h	N	Controller	Reservation Notification	5.1.12.1.32
81h	N	NVM subsystem	Sanitize Status	5.1.12.1.33

**Figure 202: Get Log Page – Log Page Identifiers**

Log Page Identifier	CSI <sup>8</sup>	Scope	Log Page Name	Reference Section
82h to BEh		I/O Command Set Specific		
BFh		Refer to the Zoned Namespace Command Set Specification		
C0h to FFh		Vendor specific <sup>5</sup>		
Notes: 1. For namespace identifiers of 0h or FFFFFFFFh. 2. For namespace identifiers other than 0h or FFFFFFFFh. 3. The Single Device Self-test Operation (SDSO) bit is cleared to '0' in the DSTO field in the Identify Controller data structure (refer to Figure 312). 4. The SDSO bit is set to '1'. 5. Selection of a UUID may be supported. Refer to section 8.1.28. 6. For NVM subsystems that support multiple domains (refer to the MDS bit in the Identify Controller data structure, Figure 312), Domain scope information is returned. 7. The scope is defined in the header of the log page (refer to the Telemetry Host-Initiated Scope field and the Telemetry Controller-Initiated Scope field). 8. If multiple I/O Command Sets are supported, then the CDW14.CSI field (refer to Figure 201) is used by the log page: Y = Yes, N = No. If Yes, then refer to the definition of the log page for details on usage.				

### 5.1.12.1 Common Log Specific Information

Refer to section 3.1.3.5 for mandatory, optional, and prohibited log pages for the various controller types.

Log pages that indicate a scope of NVM subsystem return information that is global to the NVM subsystem. Log pages that indicate a scope of Domain return information that is global to the Domain. Log pages that indicate a scope of controller return information that is specific to the controller that is processing the command. Log pages that indicate a scope of Namespace return information that is specific to the specified namespace. For log pages that indicate multiple scopes, support for multiple domains or the namespace identifier that is specified determines which information is returned. The definition of any individual field within a log page may indicate a different scope that is specific to that individual field.

For log pages with a scope of NVM subsystem or controller (as shown in Figure 202), the controller shall abort commands that specify namespace identifiers other than 0h or FFFFFFFFh with a status code of Invalid Field in Command. Otherwise, the rules for namespace identifier usage in Figure 92 apply.

#### 5.1.12.1.1 Supported Log Pages (Log Page Identifier 00h)

An NVM subsystem may support several interfaces for submitting a Get Log Page command such as an Admin Submission Queue, PCIe VDM Management Endpoint, or 2-Wire Management Endpoint (refer to the NVM Express Management Interface Specification for details on Management Endpoints) and may have zero or more instances of each of those interfaces. The log pages supported on each instance of each interface may be different. This log page is used to describe the log pages that are supported on the interface to which the Get Log Page command was submitted and attributes specific to each log page. The log page is defined in Figure 203. The attributes of each log page are described in a LID Supported and Effects data structure defined in Figure 204.

If the UUID Selection Supported bit is set to '1' for the Get Log Page command in the Commands Supported and Effects log page (refer to section 5.1.12.1.6), then the log page data reflects the log pages that are supported based on the value of the UUID Index field (refer to section 8.1.28).

For controllers that implement I/O Queues, the log pages that the controller supports are dependent on the I/O Command Set that is based on:

- the I/O Command Set selected in CC.CSS, if CC.CSS is not set to 110b; and
- the Command Set Identifier (CSI) field in CDW 14, if CC.CSS is set to 110b.

**Figure 203: Supported Log Pages Log Page**

Bytes	Description
3:0	<b>Log Page Identifier Supported 0 (LIDS0):</b> Contains the LID Supported and Effects data structure (refer to Figure 204) for the LID 0h.
7:4	<b>Log Page Identifier Supported 1 (LIDS1):</b> Contains the LID Supported and Effects data structure (refer to Figure 204) for the LID 1h.
...	...
1019:1016	<b>Log Page Identifier Supported 254 (LIDS254):</b> Contains the LID Supported and Effects data structure (refer to Figure 204) for the LID FEh.
1023:1020	<b>Log Page Identifier Supported 255 (LIDS255):</b> Contains the LID Supported and Effects data structure (refer to Figure 204) for the LID FFh.

**Figure 204: LID Supported and Effects Data Structure**

Bits	Description
31:16	<b>LID Specific Parameter (LIDSP):</b> This field is specific to the log page identifier and if not defined for the log specified by the Log Page Identifier field, this field is reserved.
15:2	Reserved
1	<p><b>Index Offset Supported (IOS):</b> If this bit is set to '1', then the controller supports an index offset for this LID in a Get Log Page command (i.e., the OT bit in the Get Log Page command is allowed to be set to '1'). If this bit is cleared to '0', then the controller does not support an index offset for this LID in a Get Log Page command (i.e., the OT bit in the Get Log Page command is only allowed to be cleared to '0').</p> <p>If the controller does not support extended data for the Get Log Page command (refer to the Log Page Extended Data Support (SPEDS) bit in the LPA field in Figure 312), then this bit shall be cleared to '0'.</p>
0	<p><b>LID Supported (LSUPP):</b> If this bit is set to '1', then the controller supports this LID for a Get Log Page command. If this bit is cleared to '0', then the controller does not support this LID for a Get Log Page command, and the host should ignore other fields in this data structure.</p> <p>Refer to section 3.1.3.5 for the LID support requirements for each controller type.</p>



### 5.1.12.1.2 Error Information (Log Page Identifier 01h)

This log page is used to describe extended error information for a command that completed with error or report an error that is not specific to a particular command. Extended error information is provided when the More (M) bit is set to '1' in the Status field for the completion queue entry associated with the command that completed with error or as part of an asynchronous event with an Error status type. This log page is global to the controller.

This error log may return the last  $n$  errors. If host software specifies a data transfer of the size of  $n$  error logs, then the error logs for the most recent  $n$  errors are returned. The ordering of the entries is based on the time when the error occurred, with the most recent error being returned as the first log entry.

Each entry in the log page returned is defined in Figure 205. The log page is a set of 64-byte entries; the maximum number of entries supported is indicated in the ELPE field in the Identify Controller data structure (refer to Figure 312). If the log page is full when a new entry is generated, the controller should insert the new entry into the log and discard the oldest entry.

The controller should clear this log page by removing all entries on power cycle and Controller Level Reset.

**Figure 205: Error Information Log Entry Data Structure**

Bytes	Description								
07:00	<p><b>Error Count (ECNT):</b> This is a 64-bit incrementing error count, indicating a unique identifier for this error. The error count starts at 1h, is incremented for each unique error log entry, and is retained across power off conditions. A value of 0h indicates an invalid entry; this value is used when there are lost entries or when there are fewer errors than the maximum number of entries the controller supports.</p> <p>If the value of this field is FFFFFFFF_FFFFFFFFh, then the field shall be set to 1h when incremented (i.e., rolls over to 1h). Prior to NVMe 1.4, processing of incrementing beyond FFFFFFFFh is unspecified.</p>								
09:08	<p><b>Submission Queue ID (SQID):</b> This field indicates the Submission Queue Identifier of the command that the error information is associated with. If the error is not specific to a particular command, then this field shall be set to FFFFh.</p>								
11:10	<p><b>Command ID (CID):</b> This field indicates the Command Identifier of the command that the error is associated with. If the error is not specific to a particular command, then this field shall be set to FFFFh.</p>								
13:12	<p><b>Status Info (STS):</b></p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:1</td> <td> <p><b>Status (STATUS):</b> This field contains the Status field from the completion queue entry (refer to Figure 100) of the command that completed. If the error is not specific to a particular command, then this field reports the most applicable status value.</p> </td> </tr> <tr> <td>0</td> <td> <p><b>Phase Tag (P):</b> This bit may indicate the Phase Tag posted for the command.</p> </td> </tr> </tbody> </table>	Bits	Description	15:1	<p><b>Status (STATUS):</b> This field contains the Status field from the completion queue entry (refer to Figure 100) of the command that completed. If the error is not specific to a particular command, then this field reports the most applicable status value.</p>	0	<p><b>Phase Tag (P):</b> This bit may indicate the Phase Tag posted for the command.</p>		
Bits	Description								
15:1	<p><b>Status (STATUS):</b> This field contains the Status field from the completion queue entry (refer to Figure 100) of the command that completed. If the error is not specific to a particular command, then this field reports the most applicable status value.</p>								
0	<p><b>Phase Tag (P):</b> This bit may indicate the Phase Tag posted for the command.</p>								
15:14	<p><b>Parameter Error Location (PEL):</b> This field indicates the byte and bit of the command parameter that the error is associated with, if applicable. If the parameter spans multiple bytes or bits, then the location indicates the least-significant byte and bit of the parameter.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:11</td> <td>Reserved</td> </tr> <tr> <td>10:08</td> <td> <p><b>Bit Location (BITLOC):</b> The offset in the byte specified by the Byte Location field to the bit in that byte that contained the error.</p> </td> </tr> <tr> <td>07:00</td> <td> <p><b>Byte Location (BYTLOC):</b> The offset in the submission queue entry to the byte in the command that contained the error. Valid values are based on the SQES field (refer to Figure 312) (e.g., a value of 6 in SQES indicates that the valid values for this field are 0 to 63).</p> </td> </tr> </tbody> </table> <p>If the error is not specific to a particular command, then this field shall be set to FFFFh.</p>	Bits	Description	15:11	Reserved	10:08	<p><b>Bit Location (BITLOC):</b> The offset in the byte specified by the Byte Location field to the bit in that byte that contained the error.</p>	07:00	<p><b>Byte Location (BYTLOC):</b> The offset in the submission queue entry to the byte in the command that contained the error. Valid values are based on the SQES field (refer to Figure 312) (e.g., a value of 6 in SQES indicates that the valid values for this field are 0 to 63).</p>
Bits	Description								
15:11	Reserved								
10:08	<p><b>Bit Location (BITLOC):</b> The offset in the byte specified by the Byte Location field to the bit in that byte that contained the error.</p>								
07:00	<p><b>Byte Location (BYTLOC):</b> The offset in the submission queue entry to the byte in the command that contained the error. Valid values are based on the SQES field (refer to Figure 312) (e.g., a value of 6 in SQES indicates that the valid values for this field are 0 to 63).</p>								

Figure 205: Error Information Log Entry Data Structure

Bytes	Description						
23:16	<b>Logical Block Address (LBA):</b> This field indicates I/O Command Set specific data about the error condition, if applicable. The description is described in the applicable I/O Command Set specification.  NOTE: The original field name has been retained for historical continuity.						
27:24	<b>Namespace Identifier (NSID):</b> This field indicates the NSID of the namespace that the error is associated with, if applicable.						
28	<b>Vendor Specific Information Available (VSIA):</b> If there is additional vendor specific error information available, this field provides the log page identifier associated with that page. A value of 0h indicates that no additional information is available. Valid values are in the range of 80h to FFh.						
29	<b>Transport Type (TRTYPE):</b> This field indicates the Transport Type of the transport associated with the error. The values in this field are the same as the TRTYPE values in the Discovery Log Page Entry (refer to section 5.1.12.3.1). If the error is not transport related, this field shall be cleared to 0h. If the error is transport related, this field shall be set to the type of the transport as defined in the TRTYPE field within Figure 294.						
30	<b>Command Set Indicator (CSI):</b> This field contains command set indicator for the command that the error is associated with. This field is valid if the Log Page Version field is greater than or equal to 1h.						
31	<b>Opcode (OPC):</b> This field contains opcode for the command that the error is associated with. This field is valid if the Log Page Version field is greater than or equal to 1h.  This field, the CSI field, and the Submission Queue ID field uniquely indicate the command and the command set.						
39:32	<b>Command Specific Information (CSINFO):</b> This field contains the command specific information. If used, the command definition specifies the information returned.						
41:40	<b>Transport Type Specific Information (TTSI):</b> This field indicates additional transport type specific error information. If multiple errors exist, then this field indicates additional information about the first error. This field is transport type dependent (refer to TRTYPE) as follows:						
	<table border="1"> <thead> <tr> <th>Transport Type</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>All other values</td> <td>Reserved</td> </tr> <tr> <td>3h</td> <td>This field indicates, the offset, in bytes, from the start of the Transport Header to the start of the field that is in error. If multiple errors exist, then this field indicates the lowest offset that is in error.</td> </tr> </tbody> </table>	Transport Type	Description	All other values	Reserved	3h	This field indicates, the offset, in bytes, from the start of the Transport Header to the start of the field that is in error. If multiple errors exist, then this field indicates the lowest offset that is in error.
	Transport Type	Description					
All other values	Reserved						
3h	This field indicates, the offset, in bytes, from the start of the Transport Header to the start of the field that is in error. If multiple errors exist, then this field indicates the lowest offset that is in error.						
Reserved							
62:42	Reserved						
63	<b>Log Page Version (LPVER):</b> This field shall be set to 1h.						

### 5.1.12.1.3 SMART / Health Information (Log Page Identifier 02h)

This log page is used to provide SMART and general health information. The information provided is over the life of the controller and is retained across power cycles unless otherwise specified. To request the controller log page, the namespace identifier specified is FFFFFFFFh or 0h. For compatibility with implementations compliant with NVM Express Base Specification, Revision 1.4 and earlier, hosts should use a namespace identifier of FFFFFFFFh to request the controller log page. The controller may also support requesting the log page on a per namespace basis, as indicated by the SMART Support bit of the LPA field in the Identify Controller data structure in Figure 312.

If the log page is not supported on a per namespace basis and:

- if a namespace identifier other than 0h or FFFFFFFFh is specified by the host, then the controller shall abort the command with a status code of Invalid Field in Command; and
- if a namespace identifier of 0h or FFFFFFFFh is specified by the host, then the controller returns the controller log page.

There is no namespace specific information defined in the SMART / Health Information log page in this revision of the specification, thus the controller log page and namespaces specific log page contain identical information.

Critical warnings regarding the health of the NVM subsystem may be indicated via an asynchronous event notification to the host. The warnings that results in an asynchronous event notification to the host are configured using the Set Features command; refer to section 5.1.25.1.5.

Performance may be calculated using parameters returned as part of the SMART / Health Information log. Specifically, the number of Read or Write commands, the amount of data read or written, and the amount of controller busy time enables both I/Os per second and bandwidth to be calculated.

The log page returned is defined in Figure 206.

**Figure 206: SMART / Health Information Log Page**

Bytes	Description																
00	<p><b>Critical Warning (CW):</b> This field indicates critical warnings for the state of the controller. Each bit corresponds to a critical warning type; multiple bits may be set to '1'. If a bit is cleared to '0', then that critical warning does not apply. Critical warnings may result in an asynchronous event notification to the host. Bits in this field represent the state at the time the Get Log Page command is processed and may not reflect the state at the time a related asynchronous event notification, if any, occurs or occurred.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:6</td> <td>Reserved</td> </tr> <tr> <td>5</td> <td><b>Persistent Memory Region Read-Only (PMRRO):</b> If this bit is set to '1', then the Persistent Memory Region has become read-only or unreliable (refer to section 8.2.4).</td> </tr> <tr> <td>4</td> <td><b>Volatile Memory Backup Failed (VMBF):</b> If this bit is set to '1', then the volatile memory backup device has failed. This field is only valid if the controller has a volatile memory backup solution.</td> </tr> <tr> <td>3</td> <td><b>All Media Read-Only (AMRO):</b> If this bit is set to '1', then all of the media has been placed in read only mode. The controller shall not set this bit to '1' if the read-only condition on the media is a result of a change in the write protection state of a namespace (refer to section 8.1.16.1).</td> </tr> <tr> <td>2</td> <td><b>NVM Subsystem Degraded Reliability (NDR):</b> If this bit is set to '1', then the NVM subsystem reliability has been degraded due to significant media related errors or any internal error that degrades NVM subsystem reliability.</td> </tr> <tr> <td>1</td> <td><b>Temperature Threshold Condition (TTC):</b> If this bit is set to '1', then a temperature is:                             <ul style="list-style-type: none"> <li>a) greater than or equal to an over temperature threshold; or</li> <li>b) less than or equal to an under temperature threshold,</li> </ul>                             (refer to section 5.1.25.1.3).                         </td> </tr> <tr> <td>0</td> <td><b>Available Spare Capacity Below Threshold (ASCBT):</b> If this bit is set to '1', then the available spare capacity has fallen below the threshold.</td> </tr> </tbody> </table>	Bits	Description	7:6	Reserved	5	<b>Persistent Memory Region Read-Only (PMRRO):</b> If this bit is set to '1', then the Persistent Memory Region has become read-only or unreliable (refer to section 8.2.4).	4	<b>Volatile Memory Backup Failed (VMBF):</b> If this bit is set to '1', then the volatile memory backup device has failed. This field is only valid if the controller has a volatile memory backup solution.	3	<b>All Media Read-Only (AMRO):</b> If this bit is set to '1', then all of the media has been placed in read only mode. The controller shall not set this bit to '1' if the read-only condition on the media is a result of a change in the write protection state of a namespace (refer to section 8.1.16.1).	2	<b>NVM Subsystem Degraded Reliability (NDR):</b> If this bit is set to '1', then the NVM subsystem reliability has been degraded due to significant media related errors or any internal error that degrades NVM subsystem reliability.	1	<b>Temperature Threshold Condition (TTC):</b> If this bit is set to '1', then a temperature is: <ul style="list-style-type: none"> <li>a) greater than or equal to an over temperature threshold; or</li> <li>b) less than or equal to an under temperature threshold,</li> </ul> (refer to section 5.1.25.1.3).	0	<b>Available Spare Capacity Below Threshold (ASCBT):</b> If this bit is set to '1', then the available spare capacity has fallen below the threshold.
Bits	Description																
7:6	Reserved																
5	<b>Persistent Memory Region Read-Only (PMRRO):</b> If this bit is set to '1', then the Persistent Memory Region has become read-only or unreliable (refer to section 8.2.4).																
4	<b>Volatile Memory Backup Failed (VMBF):</b> If this bit is set to '1', then the volatile memory backup device has failed. This field is only valid if the controller has a volatile memory backup solution.																
3	<b>All Media Read-Only (AMRO):</b> If this bit is set to '1', then all of the media has been placed in read only mode. The controller shall not set this bit to '1' if the read-only condition on the media is a result of a change in the write protection state of a namespace (refer to section 8.1.16.1).																
2	<b>NVM Subsystem Degraded Reliability (NDR):</b> If this bit is set to '1', then the NVM subsystem reliability has been degraded due to significant media related errors or any internal error that degrades NVM subsystem reliability.																
1	<b>Temperature Threshold Condition (TTC):</b> If this bit is set to '1', then a temperature is: <ul style="list-style-type: none"> <li>a) greater than or equal to an over temperature threshold; or</li> <li>b) less than or equal to an under temperature threshold,</li> </ul> (refer to section 5.1.25.1.3).																
0	<b>Available Spare Capacity Below Threshold (ASCBT):</b> If this bit is set to '1', then the available spare capacity has fallen below the threshold.																
02:01	<p><b>Composite Temperature (CTEMP):</b> Contains a value corresponding to a temperature in Kelvins that represents the current composite temperature of the controller and namespace(s) associated with that controller. The manner in which this value is computed is implementation specific and may not represent the actual temperature of any physical point in the NVM subsystem. The value of this field may be used to trigger an asynchronous event (refer to section 5.1.25.1.3).</p> <p>Warning and critical overheating composite temperature threshold values are reported by the WCTEMP and CTEMP fields in the Identify Controller data structure in Figure 312.</p>																
03	<b>Available Spare (AVSP):</b> Contains a normalized percentage (0% to 100%) of the remaining spare capacity available.																
04	<b>Available Spare Threshold (AVSPT):</b> When the Available Spare falls below the threshold indicated in this field, an asynchronous event completion may occur. The value is indicated as a normalized percentage (0% to 100%). The values 101 to 255 are reserved.																
05	<p><b>Percentage Used (PUSED):</b> Contains a vendor specific estimate of the percentage of NVM subsystem life used based on the actual usage and the manufacturer's prediction of NVM life. A value of 100 indicates that the estimated endurance of the NVM in the NVM subsystem has been consumed, but may not indicate an NVM subsystem failure. The value is allowed to exceed 100. Percentages greater than 254 shall be represented as 255. This value shall be updated once per power-on hour (when the controller is not in a sleep state).</p> <p>Refer to the JEDEC JESD218B-02 standard for SSD device life and endurance measurement techniques.</p>																

**Figure 206: SMART / Health Information Log Page**

Bytes	Description												
06	<p><b>Endurance Group Critical Warning Summary (EGCWS):</b> This field indicates critical warnings for the state of Endurance Groups. Each bit corresponds to a critical warning type, multiple bits may be set to '1'. If a bit is cleared to '0', then that critical warning does not apply to any Endurance Group. Critical warnings may result in an asynchronous event notification to the host. Bits in this field represent the current associated state and are not persistent.</p> <p>If a bit is set to '1' in one or more Endurance Groups, then the corresponding bit shall be set to '1' in this field.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:4</td> <td>Reserved</td> </tr> <tr> <td>3</td> <td><b>Endurance Group Read-Only (EGRO):</b> If this bit is set to '1', then the namespaces in one or more Endurance Groups have been placed in read only mode not as a result of a change in the write protection state of a namespace (refer to section 8.1.16.1).</td> </tr> <tr> <td>2</td> <td><b>Endurance Group Degraded Reliability (EGDR):</b> If this bit is set to '1', then the reliability of one or more Endurance Groups has been degraded due to significant media related errors or any internal error that degrades NVM subsystem reliability.</td> </tr> <tr> <td>1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td><b>Endurance Group Available Spare Capacity Below Threshold (EGASCBT):</b> If this bit is set to '1', then the available spare capacity of one or more Endurance Groups has fallen below the threshold.</td> </tr> </tbody> </table>	Bits	Description	7:4	Reserved	3	<b>Endurance Group Read-Only (EGRO):</b> If this bit is set to '1', then the namespaces in one or more Endurance Groups have been placed in read only mode not as a result of a change in the write protection state of a namespace (refer to section 8.1.16.1).	2	<b>Endurance Group Degraded Reliability (EGDR):</b> If this bit is set to '1', then the reliability of one or more Endurance Groups has been degraded due to significant media related errors or any internal error that degrades NVM subsystem reliability.	1	Reserved	0	<b>Endurance Group Available Spare Capacity Below Threshold (EGASCBT):</b> If this bit is set to '1', then the available spare capacity of one or more Endurance Groups has fallen below the threshold.
Bits	Description												
7:4	Reserved												
3	<b>Endurance Group Read-Only (EGRO):</b> If this bit is set to '1', then the namespaces in one or more Endurance Groups have been placed in read only mode not as a result of a change in the write protection state of a namespace (refer to section 8.1.16.1).												
2	<b>Endurance Group Degraded Reliability (EGDR):</b> If this bit is set to '1', then the reliability of one or more Endurance Groups has been degraded due to significant media related errors or any internal error that degrades NVM subsystem reliability.												
1	Reserved												
0	<b>Endurance Group Available Spare Capacity Below Threshold (EGASCBT):</b> If this bit is set to '1', then the available spare capacity of one or more Endurance Groups has fallen below the threshold.												
31:07	Reserved												
47:32	<p><b>Data Units Read (DUR):</b> Contains the number of 512 byte data units the host has read from the controller as part of processing a SMART Data Units Read Command; this value does not include metadata. This value is reported in thousands (i.e., a value of 1 corresponds to 1,000 units of 512 bytes read) and is rounded up (e.g., one indicates that the number of 512 byte data units read is from 1 to 1,000, three indicates that the number of 512 byte data units read is from 2,001 to 3,000).</p> <p>Refer to the specific I/O Command Set specification for the list of SMART Data Units Read Commands that affect this field.</p> <p>A value of 0h in this field indicates that the number of SMART Data Units Read is not reported.</p>												
63:48	<p><b>Data Units Written (DUW):</b> Contains the number of 512 byte data units the host has written to the controller as part of processing a User Data Out Command; this value does not include metadata. This value is reported in thousands (i.e., a value of 1 corresponds to 1,000 units of 512 bytes written) and is rounded up (e.g., one indicates that the number of 512 byte data units written is from 1 to 1,000, three indicates that the number of 512 byte data units written is from 2,001 to 3,000).</p> <p>Refer to the specific I/O Command Set specification for the list of User Data Out Commands that affect this field.</p> <p>A value of 0h in this field indicates that the number of Data Units Written is not reported.</p>												
79:64	<p><b>Host Read Commands (HRC):</b> Contains the number of SMART Host Read Commands completed by the controller.</p> <p>Refer to the specific I/O Command Set specification for the list of SMART Host Read Commands that affect this field.</p>												
95:80	<p><b>Host Write Commands (HWC):</b> Contains the number of User Data Out Commands completed by the controller.</p> <p>Refer to the specific I/O Command Set specification for the list of User Data Out Commands that affect this field.</p>												
111:96	<p><b>Controller Busy Time (CBT):</b> Contains the amount of time the controller is busy with I/O commands. The controller is busy when there is a command outstanding to an I/O Queue (specifically, a command was issued via an I/O Submission Queue Tail doorbell write and the corresponding completion queue entry has not been posted yet to the associated I/O Completion Queue). This value is reported in minutes.</p>												
127:112	<b>Power Cycles (PWRC):</b> Contains the number of power cycles.												
143:128	<b>Power On Hours (POH):</b> Contains the number of power-on hours. This may not include time that the controller was powered and in a non-operational power state.												

**Figure 206: SMART / Health Information Log Page**

Bytes	Description
159:144	<p><b>Unexpected Power Losses (UPL):</b> Contains a count of the number of unexpected power losses. This count shall be incremented if, and only if, main power is lost when:</p> <ul style="list-style-type: none"> <li>a) the controller does not report it is ready to be powered off (i.e., the CSTS.SHST field is not set to 10b); or</li> <li>b) media is not in the shutdown state (refer to Figure 85) because an Admin command that accesses media as defined by Figure 84 was processed via the out-of-band mechanism with the Ignore Shutdown bit set to '1' (refer to the NVM Express Management Interface Specification) while shutdown processing was reported as in progress or was reported as complete (i.e., the CSTS.SHST field was set to 01b or set to 10b).</li> </ul> <p>Note that this field was previously named Unsafe Shutdowns.</p> <p>If power is lost when the CSTS.SHST field is not set to 10b, then subsequent controller initialization (i.e., the amount of time from when the CC.EN bit transitions from '0' to '1' until the CSTS.RDY bit transitions from '0' to '1') may take longer for any NVM subsystem, and data corruption may occur for any NVM subsystem that is not protected against power loss.</p> <p>If the Controller Power Scope (i.e., CAP.CPS) field is cleared to 00b (i.e., Not Reported) or set to 01b (i.e., Controller scope), then the controller reports that the controller is ready to be powered off when the controller is shutdown (i.e., CSTS.SHST field is set to 10b).</p> <p>If the CAP.CPS field is set to 10b (i.e., Domain scope), then the controller reports that the domain is ready to be powered off when all the controllers in that domain are shutdown (e.g., NVM Subsystem Shutdown processing is complete).</p> <p>If the CAP.CPS field is set to 11b (i.e., NVM subsystem scope), then the controller reports that the NVM subsystem is ready to be powered off when all controllers in the NVM subsystem are shutdown (e.g., NVM Subsystem Shutdown processing is complete).</p>
175:160	<p><b>Media and Data Integrity Errors (MDIE):</b> Contains the number of occurrences where the controller detected an unrecovered data integrity error. Errors such as uncorrectable ECC, CRC checksum failure, or LBA tag mismatch are included in this field. Errors introduced as a result of a Write Uncorrectable command (refer to the NVM Command Set Specification) may or may not be included in this field.</p>
191:176	<p><b>Number of Error Information Log Entries (NEILE):</b> Contains the number of Error Information Log Entries over the life of the controller.</p>
195:192	<p><b>Warning Composite Temperature Time (WCTT):</b> If the Temperature Threshold Hysteresis Attributes (TMPTTHA) field cleared to 0h (refer to Figure 312), then this field contains the amount of time in minutes that the controller is operational and the Composite Temperature is greater than or equal to the Warning Composite Temperature Threshold (WCTEMP) field and less than the Critical Composite Temperature Threshold (CCTEMP) field in the Identify Controller data structure in Figure 312.</p> <p>If the Temperature Threshold Maximum Hysteresis (TMPTMH) and Temperature Threshold Hysteresis (TMPTHH) fields are non-zero, then hysteresis time is included in the time specified in this field (refer to section 5.1.25.1.3.1).</p> <p>If the value of the WCTEMP or CCTEMP field is 0h, then this field is always cleared to 0h regardless of the Composite Temperature value.</p>
199:196	<p><b>Critical Composite Temperature Time (CCTT):</b> Contains the amount of time in minutes that the controller is operational and the Composite Temperature is greater than or equal to the Critical Composite Temperature Threshold (CCTEMP) field in the Identify Controller data structure in Figure 312.</p> <p>If the value of the CCTEMP field is 0h, then this field is always cleared to 0h regardless of the Composite Temperature value.</p>
201:200	<p><b>Temperature Sensor 1 (TSEN1):</b> Contains the current temperature reported by temperature sensor 1. This field is defined by Figure 207.</p>
203:202	<p><b>Temperature Sensor 2 (TSEN2):</b> Contains the current temperature reported by temperature sensor 2. This field is defined by Figure 207.</p>
205:204	<p><b>Temperature Sensor 3 (TSEN3):</b> Contains the current temperature reported by temperature sensor 3. This field is defined by Figure 207.</p>

**Figure 206: SMART / Health Information Log Page**

Bytes	Description
207:206	<b>Temperature Sensor 4 (TSEN4):</b> Contains the current temperature reported by temperature sensor 4. This field is defined by Figure 207.
209:208	<b>Temperature Sensor 5 (TSEN5):</b> Contains the current temperature reported by temperature sensor 5. This field is defined by Figure 207.
211:210	<b>Temperature Sensor 6 (TSEN6):</b> Contains the current temperature reported by temperature sensor 6. This field is defined by Figure 207.
213:212	<b>Temperature Sensor 7 (TSEN7):</b> Contains the current temperature reported by temperature sensor 7. This field is defined by Figure 207.
215:214	<b>Temperature Sensor 8 (TSEN8):</b> Contains the current temperature reported by temperature sensor 8. This field is defined by Figure 207.
219:216	<b>Thermal Management Temperature 1 Transition Count (TMT1TC):</b> Contains the number of times the controller transitioned to lower power active power states or performed vendor specific thermal management actions while minimizing the impact on performance in order to attempt to reduce the Composite Temperature because of the host controlled thermal management feature (refer to section 8.1.17.5) (i.e., the Composite Temperature rose above the Thermal Management Temperature 1). This counter shall not wrap once the value FFFFFFFFh is reached. A value of 0h, indicates that this transition has never occurred or this field is not implemented.
223:220	<b>Thermal Management Temperature 2 Transition Count (TMT2TC):</b> Contains the number of times the controller transitioned to lower power active power states or performed vendor specific thermal management actions regardless of the impact on performance (e.g., heavy throttling) in order to attempt to reduce the Composite Temperature because of the host controlled thermal management feature (refer to section 8.1.17.5) (i.e., the Composite Temperature rose above the Thermal Management Temperature 2). This counter shall not wrap once the value FFFFFFFFh is reached. A value of 0h, indicates that this transition has never occurred or this field is not implemented.
227:224	<b>Total Time For Thermal Management Temperature 1 (TTMT1):</b> Contains the number of seconds that the controller had transitioned to lower power active power states or performed vendor specific thermal management actions while minimizing the impact on performance in order to attempt to reduce the Composite Temperature because of the host controlled thermal management feature (refer to section 8.1.17.5). This counter shall not wrap once the value FFFFFFFFh is reached. A value of 0h, indicates that this transition has never occurred or this field is not implemented.
231:228	<b>Total Time For Thermal Management Temperature 2 (TTMT2):</b> Contains the number of seconds that the controller had transitioned to lower power active power states or performed vendor specific thermal management actions regardless of the impact on performance (e.g., heavy throttling) in order to attempt to reduce the Composite Temperature because of the host controlled thermal management feature (refer to section 8.1.17.5). This counter shall not wrap once the value FFFFFFFFh is reached. A value of 0h, indicates that this transition has never occurred or this field is not implemented.
511:232	Reserved

**Figure 207: Temperature Sensor Data Structure**

Bits	Description
15:00	<b>Temperature Sensor Temperature (TST):</b> Contains the current temperature in Kelvins reported by the temperature sensor.  The physical point in the NVM subsystem whose temperature is reported by the temperature sensor and the temperature accuracy is implementation specific. An implementation that does not implement the temperature sensor reports a value of 0h. The temperature reported by a temperature sensor may be used to trigger an asynchronous event (refer to section 5.1.25.1.3).

### 5.1.12.1.4 Firmware Slot Information (Log Page Identifier 03h)

This log page is used to describe the firmware revision stored in each firmware slot supported. The firmware revision is indicated as an ASCII string. The log page also indicates the active slot number. The log page returned is defined in Figure 208.

**Figure 208: Firmware Slot Information Log Page**

Bytes	Description										
00	<b>Active Firmware Info (AFI):</b> Specifies information about the active firmware revision.										
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7</td> <td>Reserved</td> </tr> <tr> <td>6:4</td> <td><b>Next Active Firmware Slot (NAFS):</b> This field indicates the firmware slot that is going to be activated at the next Controller Level Reset. If this field is cleared to 0h, then the controller does not indicate the firmware slot that is going to be activated at the next Controller Level Reset.</td> </tr> <tr> <td>3</td> <td>Reserved</td> </tr> <tr> <td>2:0</td> <td><b>Current Active Firmware Slot (CAFS):</b> This field indicates the firmware slot from which the actively running firmware revision was loaded</td> </tr> </tbody> </table>	Bits	Description	7	Reserved	6:4	<b>Next Active Firmware Slot (NAFS):</b> This field indicates the firmware slot that is going to be activated at the next Controller Level Reset. If this field is cleared to 0h, then the controller does not indicate the firmware slot that is going to be activated at the next Controller Level Reset.	3	Reserved	2:0	<b>Current Active Firmware Slot (CAFS):</b> This field indicates the firmware slot from which the actively running firmware revision was loaded
	Bits	Description									
	7	Reserved									
	6:4	<b>Next Active Firmware Slot (NAFS):</b> This field indicates the firmware slot that is going to be activated at the next Controller Level Reset. If this field is cleared to 0h, then the controller does not indicate the firmware slot that is going to be activated at the next Controller Level Reset.									
3	Reserved										
2:0	<b>Current Active Firmware Slot (CAFS):</b> This field indicates the firmware slot from which the actively running firmware revision was loaded										
07:01	Reserved										
15:08	<b>Firmware Revision for Slot 1 (FRS1):</b> Contains the revision of the firmware downloaded to firmware slot 1. If no valid firmware revision is present or if this slot is unsupported, this field shall be cleared to 0h.										
23:16	<b>Firmware Revision for Slot 2 (FRS2):</b> Contains the revision of the firmware downloaded to firmware slot 2. If no valid firmware revision is present or if this slot is unsupported, this field shall be cleared to 0h.										
31:24	<b>Firmware Revision for Slot 3 (FRS3):</b> Contains the revision of the firmware downloaded to firmware slot 3. If no valid firmware revision is present or if this slot is unsupported, this field shall be cleared to 0h.										
39:32	<b>Firmware Revision for Slot 4 (FRS4):</b> Contains the revision of the firmware downloaded to firmware slot 4. If no valid firmware revision is present or if this slot is unsupported, this field shall be cleared to 0h.										
47:40	<b>Firmware Revision for Slot 5 (FRS5):</b> Contains the revision of the firmware downloaded to firmware slot 5. If no valid firmware revision is present or if this slot is unsupported, this field shall be cleared to 0h.										
55:48	<b>Firmware Revision for Slot 6 (FRS6):</b> Contains the revision of the firmware downloaded to firmware slot 6. If no valid firmware revision is present or if this slot is unsupported, this field shall be cleared to 0h.										
63:56	<b>Firmware Revision for Slot 7 (FRS7):</b> Contains the revision of the firmware downloaded to firmware slot 7. If no valid firmware revision is present or if this slot is unsupported, this field shall be cleared to 0h.										
511:64	Reserved										

### 5.1.12.1.5 Changed Attached Namespace List (Log Page Identifier 04h)

This log page is used to describe changes to attached namespaces for this controller, since the last time that this log page was read, that:

- a) have changed information in their Identify Namespace data structures (refer to section 1.5.49);
- b) were previously unattached to the controller and have since been attached to the controller;
- c) were previously attached to the controller and have since been detached from the controller; and
- d) were deleted.

The log page contains a Namespace List with up to 1,024 entries. If more than 1,024 namespaces have changed attributes since the last time the log page was read, the first entry in the log page shall be set to FFFFFFFFh and the remainder of the list shall be zero filled.

### 5.1.12.1.6 Commands Supported and Effects (Log Page Identifier 05h)

This log page is used to describe the commands that the controller supports and the effects of those commands on the state of the NVM subsystem. The log page is 4,096 bytes in size. There is one Commands Supported and Effects data structure per Admin command opcode value. For controllers that implement I/O Queues, there is one Commands Supported and Effects data structure per I/O command opcode value for a specified I/O Command Set based on:

- a) the I/O Command Set selected in CC.CSS, if CC.CSS is not set to 110b; and
- b) the Command Set Identifier field in CDW 14, if CC.CSS is set to 110b.

**Figure 209: Commands Supported and Effects Log Page**

Bytes	Description
03:00	<b>Admin Command Supported 0 (ACS0):</b> Contains the Commands Supported and Effects data structure (refer to Figure 210) for the Admin command with an opcode value of 0h.
07:04	<b>Admin Command Supported 1 (ACS1):</b> Contains the Commands Supported and Effects data structure (refer to Figure 210) for the Admin command with an opcode value of 1h.
...	...
1019:1016	<b>Admin Command Supported 254 (ACS254):</b> Contains the Commands Supported and Effects data structure (refer to Figure 210) for the Admin command with an opcode value of 254.
1023:1020	<b>Admin Command Supported 255 (ACS255):</b> Contains the Commands Supported and Effects data structure (refer to Figure 210) for the Admin command with an opcode value of 255.
1027:1024	<b>I/O Command Supported 0 (IOCS0):</b> Contains the Commands Supported and Effects data structure (refer to Figure 210) for the I/O command with an opcode value of 0h.
1031:1028	<b>I/O Command Supported 1 (IOCS1):</b> Contains the Commands Supported and Effects data structure (refer to Figure 210) for the I/O command with an opcode value of 1h.
...	...
2043:2040	<b>I/O Command Supported 254 (IOCS254):</b> Contains the Commands Supported and Effects data structure (refer to Figure 210) for the I/O command with an opcode value of 254.
2047:2044	<b>I/O Command Supported 255 (IOCS255):</b> Contains the Commands Supported and Effects data structure (refer to Figure 210) for the I/O command with an opcode value of 255.
4095:2048	Reserved

The Commands Supported and Effects data structure describes the overall possible effect of a command, including any optional features of the command.

Host software may take command effects into account when determining how to submit commands and actions to take after the command is complete. It is recommended that if a command may change a particular capability that host software re-enumerate and/or re-initialize the associated capability after the command is complete. For example, if a namespace capability change may occur, then host software is recommended to pause the use of the associated namespace, submit the command that may cause a namespace capability change and wait for its completion, and then re-issue the Identify command.

If the namespace is attached to multiple controllers, the host(s) associated with those controllers should coordinate their commands to meet the Command Submission and Execution requirements (refer to Figure 210). The details of this coordination are outside the scope of this specification.



**Figure 210: Commands Supported and Effects Data Structure**

Bits	Description																
31:20	<p><b>Command Scope (CSP):</b> This field defines the scope for the associated command. If the value of this field is 0h then no scope is reported. A command may have multiple scopes depending on the effects of the command based on the parameters passed to that command. For a command that supports multiple scopes, multiple bits may set to '1'.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>11:6</td> <td>Reserved</td> </tr> <tr> <td>5</td> <td><b>NVM Subsystem Scope (NSSCPE):</b> If this bit is set to '1', then the command performs actions that may impact the whole NVM subsystem. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that does not impact the whole NVM subsystem.</td> </tr> <tr> <td>4</td> <td><b>Domain Scope (DSCPE):</b> If this bit is set to '1', then the command performs actions that may impact a single Domain. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that does not impact a single Domain.</td> </tr> <tr> <td>3</td> <td><b>Endurance Group Scope (EGSCPE):</b> If this bit is set to '1', then the command performs actions that may impact Endurance Groups. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that does not impact a single Endurance Group.</td> </tr> <tr> <td>2</td> <td><b>NVM Set Scope (NSSCPE):</b> If this bit is set to '1', then the command performs actions that may impact NVM Sets. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that do not impact NVM Sets.</td> </tr> <tr> <td>1</td> <td><b>Controller Scope (CSCPE):</b> If this bit is set to '1', then the command performs actions that may impact controllers. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that do not impact controllers.</td> </tr> <tr> <td>0</td> <td><b>Namespace Scope (NSCPE):</b> If this bit is set to '1', then the command performs actions that may impact namespaces. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that do not impact namespaces.</td> </tr> </tbody> </table>	Bits	Description	11:6	Reserved	5	<b>NVM Subsystem Scope (NSSCPE):</b> If this bit is set to '1', then the command performs actions that may impact the whole NVM subsystem. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that does not impact the whole NVM subsystem.	4	<b>Domain Scope (DSCPE):</b> If this bit is set to '1', then the command performs actions that may impact a single Domain. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that does not impact a single Domain.	3	<b>Endurance Group Scope (EGSCPE):</b> If this bit is set to '1', then the command performs actions that may impact Endurance Groups. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that does not impact a single Endurance Group.	2	<b>NVM Set Scope (NSSCPE):</b> If this bit is set to '1', then the command performs actions that may impact NVM Sets. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that do not impact NVM Sets.	1	<b>Controller Scope (CSCPE):</b> If this bit is set to '1', then the command performs actions that may impact controllers. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that do not impact controllers.	0	<b>Namespace Scope (NSCPE):</b> If this bit is set to '1', then the command performs actions that may impact namespaces. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that do not impact namespaces.
	Bits	Description															
	11:6	Reserved															
	5	<b>NVM Subsystem Scope (NSSCPE):</b> If this bit is set to '1', then the command performs actions that may impact the whole NVM subsystem. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that does not impact the whole NVM subsystem.															
	4	<b>Domain Scope (DSCPE):</b> If this bit is set to '1', then the command performs actions that may impact a single Domain. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that does not impact a single Domain.															
	3	<b>Endurance Group Scope (EGSCPE):</b> If this bit is set to '1', then the command performs actions that may impact Endurance Groups. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that does not impact a single Endurance Group.															
	2	<b>NVM Set Scope (NSSCPE):</b> If this bit is set to '1', then the command performs actions that may impact NVM Sets. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that do not impact NVM Sets.															
	1	<b>Controller Scope (CSCPE):</b> If this bit is set to '1', then the command performs actions that may impact controllers. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that do not impact controllers.															
0	<b>Namespace Scope (NSCPE):</b> If this bit is set to '1', then the command performs actions that may impact namespaces. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that do not impact namespaces.																
19	<p><b>UUID Selection Supported (USS):</b> If this bit is set to '1', then the controller supports selection of a UUID by this command (refer to section 8.1.28). If this bit is cleared to '0', then the controller does not support selection of a UUID by this command.</p>																
18:16	<p><b>Command Submission and Execution (CSE):</b> This field defines the command submission and execution recommendations for the associated command.</p> <p>If the CSER field contains a non-zero value and the host supports the value in that field, then this field should be ignored by the host. If the CSER field is set to 01b, then this field shall be set to 001b.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>No command submission or execution restriction</td> </tr> <tr> <td>001b</td> <td>The command associated with this structure should only be submitted when there is no other outstanding command affecting the same namespace and another command should not be submitted that affects the same namespace until this command is complete.</td> </tr> <tr> <td>010b</td> <td>The command associated with this structure should only be submitted when there is no other outstanding command that affects any namespace and another command should not be submitted that affects any namespace until this command is complete.</td> </tr> <tr> <td>011b to 111b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	000b	No command submission or execution restriction	001b	The command associated with this structure should only be submitted when there is no other outstanding command affecting the same namespace and another command should not be submitted that affects the same namespace until this command is complete.	010b	The command associated with this structure should only be submitted when there is no other outstanding command that affects any namespace and another command should not be submitted that affects any namespace until this command is complete.	011b to 111b	Reserved						
	Value	Definition															
	000b	No command submission or execution restriction															
	001b	The command associated with this structure should only be submitted when there is no other outstanding command affecting the same namespace and another command should not be submitted that affects the same namespace until this command is complete.															
	010b	The command associated with this structure should only be submitted when there is no other outstanding command that affects any namespace and another command should not be submitted that affects any namespace until this command is complete.															
011b to 111b	Reserved																

**Figure 210: Commands Supported and Effects Data Structure**

Bits	Description								
15:14	<p><b>Command Submission and Execution Relaxations (CSER):</b> This field defines relaxed command submission and execution recommendations for the associated command.</p> <p>If this field is cleared to 00b, then the CSE field defines the command submission and execution recommendations for the associated command and this field has no effect.</p> <p>If this field is set to a non-zero value, then:</p> <ul style="list-style-type: none"> <li>if the host supports the value in this field, then the host should ignore the CSE field and use the value in this field as the command submission and execution recommendation; and</li> <li>if the host does not support the value in this field, then the host should ignore this field and use the value in the CSE field as the command submission and execution recommendation.</li> </ul> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>No relaxed command submission and execution recommendation is defined. The host should use the value in the CSE field as the command submission and execution recommendation.</td> </tr> <tr> <td>01b</td> <td>The command associated with this structure should only be submitted when there is no outstanding Admin command that affects any namespace and no Admin command should be submitted that affects any namespace until this command is complete.</td> </tr> <tr> <td>10b to 11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	No relaxed command submission and execution recommendation is defined. The host should use the value in the CSE field as the command submission and execution recommendation.	01b	The command associated with this structure should only be submitted when there is no outstanding Admin command that affects any namespace and no Admin command should be submitted that affects any namespace until this command is complete.	10b to 11b	Reserved
Value	Definition								
00b	No relaxed command submission and execution recommendation is defined. The host should use the value in the CSE field as the command submission and execution recommendation.								
01b	The command associated with this structure should only be submitted when there is no outstanding Admin command that affects any namespace and no Admin command should be submitted that affects any namespace until this command is complete.								
10b to 11b	Reserved								
13:05	Reserved								
04	<b>Controller Capability Change (CCC):</b> If this bit is set to '1', then this command may change controller capabilities. If this bit is cleared to '0', then this command does not modify controller capabilities. Controller capability changes include a firmware update that changes the capabilities reported in the CAP property.								
03	<b>Namespace Inventory Change (NIC):</b> If this bit is set to '1', then this command may change the number of namespaces or capabilities for multiple namespaces. If this bit is cleared to '0', then this command does not modify the number of namespaces or capabilities for multiple namespaces. Namespace inventory changes include adding or removing namespaces.								
02	<b>Namespace Capability Change (NCC):</b> If this bit is set to '1', then this command may change the capabilities of a single namespace. If this bit is cleared to '0', then this command does not modify any namespace capabilities for the specified namespace. Namespace capability changes include a logical format change.								
01	<b>Logical Block Content Change (LBCC):</b> If this bit is set to '1', then this command may modify user data content in one or more namespaces that contain formatted storage. If this bit is cleared to '0', then this command does not modify user data content in any namespace. User data content changes include a write to user data.  NOTE: This field applies to all user data content changes. The original name has been retained for historical continuity.								
00	<b>Command Supported (CSUPP):</b> If this bit is set to '1', then this command is supported by the controller. If this bit is cleared to '0', then this command is not supported by the controller and all other fields in this structure shall be cleared to 0h.								

**5.1.12.1.7 Device Self-test (Log Page Identifier 06h)**

This log page is used to indicate:

- the status of any device self-test operation in progress and the percentage complete of that operation; and
- the results of the last 20 device self-test operations.

The Self-test Result data structure (refer to Figure 212) contained in the Result Data Structure 1 field is always the result of the most recently completed or aborted self-test operation. The next entry in the Device Self-test Result list in the Device Self-test log page contains the results of the second most recent self-test

operation and so on. If fewer than 20 self-test operations have completed or been aborted, then the Device Self-test Result field shall be set to Fh (i.e., Entry is empty) and the Self-test Code field shall be cleared to 0h (i.e., Entry is empty) in the Device Self-test Status field in the unused Self-test Result Data Structure fields and all other fields in that Self-test Result data structure should be ignored by the host.

**Figure 211: Device Self-test Log Page**

Bytes	Description																					
00	<b>Current Device Self-Test Operation (CDSTO):</b> This field defines the current device self-test operation.																					
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:4</td> <td>Reserved</td> </tr> <tr> <td rowspan="7">3:0</td> <td rowspan="7"><b>Device Self-test Operation Status (DSTOS):</b> This field indicates the status of the current device self-test operation as defined in the following table. If a device self-test operation is in process (i.e., this field is set to 1h or 2h), then the controller shall not set this field to 0h until a new Self-test Result Data Structure is created (i.e., if a device self-test operation completes or is aborted, then the controller shall create a Self-test Result Data Structure prior to setting this field to 0h).</td> </tr> <tr> <td><table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>No device self-test operation in progress</td> </tr> <tr> <td>1h</td> <td>Short device self-test operation in progress</td> </tr> <tr> <td>2h</td> <td>Extended device self-test operation in progress</td> </tr> <tr> <td>3h to Dh</td> <td>Reserved</td> </tr> <tr> <td>Eh</td> <td>Vendor specific</td> </tr> <tr> <td>Fh</td> <td>Reserved</td> </tr> </tbody> </table></td> </tr> </tbody> </table>	Bits	Description	7:4	Reserved	3:0	<b>Device Self-test Operation Status (DSTOS):</b> This field indicates the status of the current device self-test operation as defined in the following table. If a device self-test operation is in process (i.e., this field is set to 1h or 2h), then the controller shall not set this field to 0h until a new Self-test Result Data Structure is created (i.e., if a device self-test operation completes or is aborted, then the controller shall create a Self-test Result Data Structure prior to setting this field to 0h).	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>No device self-test operation in progress</td> </tr> <tr> <td>1h</td> <td>Short device self-test operation in progress</td> </tr> <tr> <td>2h</td> <td>Extended device self-test operation in progress</td> </tr> <tr> <td>3h to Dh</td> <td>Reserved</td> </tr> <tr> <td>Eh</td> <td>Vendor specific</td> </tr> <tr> <td>Fh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	No device self-test operation in progress	1h	Short device self-test operation in progress	2h	Extended device self-test operation in progress	3h to Dh	Reserved	Eh	Vendor specific	Fh	Reserved
	Bits	Description																				
7:4	Reserved																					
3:0	<b>Device Self-test Operation Status (DSTOS):</b> This field indicates the status of the current device self-test operation as defined in the following table. If a device self-test operation is in process (i.e., this field is set to 1h or 2h), then the controller shall not set this field to 0h until a new Self-test Result Data Structure is created (i.e., if a device self-test operation completes or is aborted, then the controller shall create a Self-test Result Data Structure prior to setting this field to 0h).																					
		<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>No device self-test operation in progress</td> </tr> <tr> <td>1h</td> <td>Short device self-test operation in progress</td> </tr> <tr> <td>2h</td> <td>Extended device self-test operation in progress</td> </tr> <tr> <td>3h to Dh</td> <td>Reserved</td> </tr> <tr> <td>Eh</td> <td>Vendor specific</td> </tr> <tr> <td>Fh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h			No device self-test operation in progress	1h	Short device self-test operation in progress	2h	Extended device self-test operation in progress	3h to Dh	Reserved	Eh	Vendor specific	Fh	Reserved				
		Value	Definition																			
		0h	No device self-test operation in progress																			
		1h	Short device self-test operation in progress																			
		2h	Extended device self-test operation in progress																			
		3h to Dh	Reserved																			
Eh	Vendor specific																					
Fh	Reserved																					
01	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7</td> <td>Reserved</td> </tr> <tr> <td>6:0</td> <td><b>Device Self-test Completion Status (DSTCS):</b> This field indicates the percentage of the device self-test operation that is complete (e.g., a value of 25 indicates that 25% of the device self-test operation is complete and 75% remains to be tested). If the Device Self-test Operation Status (DSTOS) field in the Current Device Self-Test Operation field is cleared to 0h (i.e., there is no device self-test operation in progress), then this field should be ignored by the host.</td> </tr> </tbody> </table>	Bits	Description	7	Reserved	6:0	<b>Device Self-test Completion Status (DSTCS):</b> This field indicates the percentage of the device self-test operation that is complete (e.g., a value of 25 indicates that 25% of the device self-test operation is complete and 75% remains to be tested). If the Device Self-test Operation Status (DSTOS) field in the Current Device Self-Test Operation field is cleared to 0h (i.e., there is no device self-test operation in progress), then this field should be ignored by the host.															
Bits	Description																					
7	Reserved																					
6:0	<b>Device Self-test Completion Status (DSTCS):</b> This field indicates the percentage of the device self-test operation that is complete (e.g., a value of 25 indicates that 25% of the device self-test operation is complete and 75% remains to be tested). If the Device Self-test Operation Status (DSTOS) field in the Current Device Self-Test Operation field is cleared to 0h (i.e., there is no device self-test operation in progress), then this field should be ignored by the host.																					
03:02	Reserved																					
<b>Device Self-test Result List</b>																						
31:04	<b>Result Data Structure 1 (RDS1):</b> The newest Self-test Result data structure (refer to Figure 212)																					
59:32	<b>Result Data Structure 2 (RDS2):</b> The 2nd newest Self-test Result data structure (refer to Figure 212)																					
...	...																					
535:508	<b>Result Data Structure 19 (RDS19):</b> The 19th newest Self-test Result data structure (refer to Figure 212)																					
563:536	<b>Result Data Structure 20 (RDS20):</b> The 20th newest Self-test Result data structure (refer to Figure 212)																					

Figure 212: Self-test Result Data Structure

Bytes	Description																																																
00	<p><b>Device Self-test Status (DSTS):</b> This field indicates the device self-test code and the status of that operation.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:4</td> <td> <p><b>Self-test Code (DSTC):</b> This field indicates the Self-test Code value that was specified in the Device Self-test command that started the device self-test operation that this Self-test Result Data Structure describes.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Entry is empty</td> </tr> <tr> <td>1h</td> <td>Short device self-test operation</td> </tr> <tr> <td>2h</td> <td>Extended device self-test operation</td> </tr> <tr> <td>3h</td> <td>Host-Initiated Refresh operation</td> </tr> <tr> <td>4h to Dh</td> <td>Reserved</td> </tr> <tr> <td>Eh</td> <td>Vendor specific</td> </tr> <tr> <td>Fh</td> <td>Reserved</td> </tr> </tbody> </table> </td> </tr> <tr> <td>3:0</td> <td> <p><b>Device Self-test Result (DSTR):</b> This field indicates the result of the device self-test operation that this Self-test Result Data Structure describes.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Operation completed without error</td> </tr> <tr> <td>1h</td> <td>Operation was aborted by a Device Self-test command</td> </tr> <tr> <td>2h</td> <td>Operation was aborted by a Controller Level Reset</td> </tr> <tr> <td>3h</td> <td>Operation was aborted due to a removal of a namespace from the namespace inventory</td> </tr> <tr> <td>4h</td> <td>Operation was aborted due to the processing of a Format NVM command</td> </tr> <tr> <td>5h</td> <td>A fatal error or unknown test error occurred while the controller was executing the device self-test operation and the operation did not complete</td> </tr> <tr> <td>6h</td> <td>Operation completed with a segment that failed and the segment that failed is not known</td> </tr> <tr> <td>7h</td> <td>Operation completed with one or more failed segments and the first segment that failed is indicated in the Segment Number field</td> </tr> <tr> <td>8h</td> <td>Operation was aborted for unknown reason</td> </tr> <tr> <td>9h</td> <td>Operation was aborted due to a sanitize operation</td> </tr> <tr> <td>Ah to Eh</td> <td>Reserved</td> </tr> <tr> <td>Fh</td> <td>Entry is empty (i.e., does not contain a test result)</td> </tr> </tbody> </table> </td> </tr> </tbody> </table>	Bits	Description	7:4	<p><b>Self-test Code (DSTC):</b> This field indicates the Self-test Code value that was specified in the Device Self-test command that started the device self-test operation that this Self-test Result Data Structure describes.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Entry is empty</td> </tr> <tr> <td>1h</td> <td>Short device self-test operation</td> </tr> <tr> <td>2h</td> <td>Extended device self-test operation</td> </tr> <tr> <td>3h</td> <td>Host-Initiated Refresh operation</td> </tr> <tr> <td>4h to Dh</td> <td>Reserved</td> </tr> <tr> <td>Eh</td> <td>Vendor specific</td> </tr> <tr> <td>Fh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	Entry is empty	1h	Short device self-test operation	2h	Extended device self-test operation	3h	Host-Initiated Refresh operation	4h to Dh	Reserved	Eh	Vendor specific	Fh	Reserved	3:0	<p><b>Device Self-test Result (DSTR):</b> This field indicates the result of the device self-test operation that this Self-test Result Data Structure describes.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Operation completed without error</td> </tr> <tr> <td>1h</td> <td>Operation was aborted by a Device Self-test command</td> </tr> <tr> <td>2h</td> <td>Operation was aborted by a Controller Level Reset</td> </tr> <tr> <td>3h</td> <td>Operation was aborted due to a removal of a namespace from the namespace inventory</td> </tr> <tr> <td>4h</td> <td>Operation was aborted due to the processing of a Format NVM command</td> </tr> <tr> <td>5h</td> <td>A fatal error or unknown test error occurred while the controller was executing the device self-test operation and the operation did not complete</td> </tr> <tr> <td>6h</td> <td>Operation completed with a segment that failed and the segment that failed is not known</td> </tr> <tr> <td>7h</td> <td>Operation completed with one or more failed segments and the first segment that failed is indicated in the Segment Number field</td> </tr> <tr> <td>8h</td> <td>Operation was aborted for unknown reason</td> </tr> <tr> <td>9h</td> <td>Operation was aborted due to a sanitize operation</td> </tr> <tr> <td>Ah to Eh</td> <td>Reserved</td> </tr> <tr> <td>Fh</td> <td>Entry is empty (i.e., does not contain a test result)</td> </tr> </tbody> </table>	Value	Definition	0h	Operation completed without error	1h	Operation was aborted by a Device Self-test command	2h	Operation was aborted by a Controller Level Reset	3h	Operation was aborted due to a removal of a namespace from the namespace inventory	4h	Operation was aborted due to the processing of a Format NVM command	5h	A fatal error or unknown test error occurred while the controller was executing the device self-test operation and the operation did not complete	6h	Operation completed with a segment that failed and the segment that failed is not known	7h	Operation completed with one or more failed segments and the first segment that failed is indicated in the Segment Number field	8h	Operation was aborted for unknown reason	9h	Operation was aborted due to a sanitize operation	Ah to Eh	Reserved	Fh	Entry is empty (i.e., does not contain a test result)
	Bits	Description																																															
7:4	<p><b>Self-test Code (DSTC):</b> This field indicates the Self-test Code value that was specified in the Device Self-test command that started the device self-test operation that this Self-test Result Data Structure describes.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Entry is empty</td> </tr> <tr> <td>1h</td> <td>Short device self-test operation</td> </tr> <tr> <td>2h</td> <td>Extended device self-test operation</td> </tr> <tr> <td>3h</td> <td>Host-Initiated Refresh operation</td> </tr> <tr> <td>4h to Dh</td> <td>Reserved</td> </tr> <tr> <td>Eh</td> <td>Vendor specific</td> </tr> <tr> <td>Fh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	Entry is empty	1h	Short device self-test operation	2h	Extended device self-test operation	3h	Host-Initiated Refresh operation	4h to Dh	Reserved	Eh	Vendor specific	Fh	Reserved																																
Value	Definition																																																
0h	Entry is empty																																																
1h	Short device self-test operation																																																
2h	Extended device self-test operation																																																
3h	Host-Initiated Refresh operation																																																
4h to Dh	Reserved																																																
Eh	Vendor specific																																																
Fh	Reserved																																																
3:0	<p><b>Device Self-test Result (DSTR):</b> This field indicates the result of the device self-test operation that this Self-test Result Data Structure describes.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Operation completed without error</td> </tr> <tr> <td>1h</td> <td>Operation was aborted by a Device Self-test command</td> </tr> <tr> <td>2h</td> <td>Operation was aborted by a Controller Level Reset</td> </tr> <tr> <td>3h</td> <td>Operation was aborted due to a removal of a namespace from the namespace inventory</td> </tr> <tr> <td>4h</td> <td>Operation was aborted due to the processing of a Format NVM command</td> </tr> <tr> <td>5h</td> <td>A fatal error or unknown test error occurred while the controller was executing the device self-test operation and the operation did not complete</td> </tr> <tr> <td>6h</td> <td>Operation completed with a segment that failed and the segment that failed is not known</td> </tr> <tr> <td>7h</td> <td>Operation completed with one or more failed segments and the first segment that failed is indicated in the Segment Number field</td> </tr> <tr> <td>8h</td> <td>Operation was aborted for unknown reason</td> </tr> <tr> <td>9h</td> <td>Operation was aborted due to a sanitize operation</td> </tr> <tr> <td>Ah to Eh</td> <td>Reserved</td> </tr> <tr> <td>Fh</td> <td>Entry is empty (i.e., does not contain a test result)</td> </tr> </tbody> </table>	Value	Definition	0h	Operation completed without error	1h	Operation was aborted by a Device Self-test command	2h	Operation was aborted by a Controller Level Reset	3h	Operation was aborted due to a removal of a namespace from the namespace inventory	4h	Operation was aborted due to the processing of a Format NVM command	5h	A fatal error or unknown test error occurred while the controller was executing the device self-test operation and the operation did not complete	6h	Operation completed with a segment that failed and the segment that failed is not known	7h	Operation completed with one or more failed segments and the first segment that failed is indicated in the Segment Number field	8h	Operation was aborted for unknown reason	9h	Operation was aborted due to a sanitize operation	Ah to Eh	Reserved	Fh	Entry is empty (i.e., does not contain a test result)																						
Value	Definition																																																
0h	Operation completed without error																																																
1h	Operation was aborted by a Device Self-test command																																																
2h	Operation was aborted by a Controller Level Reset																																																
3h	Operation was aborted due to a removal of a namespace from the namespace inventory																																																
4h	Operation was aborted due to the processing of a Format NVM command																																																
5h	A fatal error or unknown test error occurred while the controller was executing the device self-test operation and the operation did not complete																																																
6h	Operation completed with a segment that failed and the segment that failed is not known																																																
7h	Operation completed with one or more failed segments and the first segment that failed is indicated in the Segment Number field																																																
8h	Operation was aborted for unknown reason																																																
9h	Operation was aborted due to a sanitize operation																																																
Ah to Eh	Reserved																																																
Fh	Entry is empty (i.e., does not contain a test result)																																																
01	<p><b>Segment Number (SEGN):</b> This field indicates the segment number (refer to section 8.1.7) where the first self-test failure occurred. If Device Self-test Status field bits [3:0] are not set to 7h, then this field should be ignored by the host.</p>																																																
02	<p><b>Valid Diagnostic Information (VDINFO):</b> This field indicates the diagnostic failure information that is reported.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:4</td> <td>Reserved</td> </tr> <tr> <td>3</td> <td><b>SC Valid (SCVLD):</b> If this bit is set to '1', then the contents of Status Code field are valid. If this bit is cleared to '0', then the contents of the Status Code field are invalid.</td> </tr> <tr> <td>2</td> <td><b>SCT Valid (SCTVLD):</b> If this bit is set to '1', then the contents of the Status Code Type field are valid. If this bit is cleared to '0', then the contents of the Status Code Type field are invalid.</td> </tr> <tr> <td>1</td> <td><b>FLBA Valid (FVLD):</b> If this bit is set to '1', then the contents of the Failing LBA field are valid. If this bit is cleared to '0', then the contents of the Failing LBA field are invalid.</td> </tr> <tr> <td>0</td> <td><b>NSID Valid (NSIDVLD):</b> If this bit is set to '1', then the contents of the Namespace Identifier field are valid. If this bit is cleared to '0', then the contents of the Namespace Identifier field are invalid.</td> </tr> </tbody> </table>	Bits	Description	7:4	Reserved	3	<b>SC Valid (SCVLD):</b> If this bit is set to '1', then the contents of Status Code field are valid. If this bit is cleared to '0', then the contents of the Status Code field are invalid.	2	<b>SCT Valid (SCTVLD):</b> If this bit is set to '1', then the contents of the Status Code Type field are valid. If this bit is cleared to '0', then the contents of the Status Code Type field are invalid.	1	<b>FLBA Valid (FVLD):</b> If this bit is set to '1', then the contents of the Failing LBA field are valid. If this bit is cleared to '0', then the contents of the Failing LBA field are invalid.	0	<b>NSID Valid (NSIDVLD):</b> If this bit is set to '1', then the contents of the Namespace Identifier field are valid. If this bit is cleared to '0', then the contents of the Namespace Identifier field are invalid.																																				
Bits	Description																																																
7:4	Reserved																																																
3	<b>SC Valid (SCVLD):</b> If this bit is set to '1', then the contents of Status Code field are valid. If this bit is cleared to '0', then the contents of the Status Code field are invalid.																																																
2	<b>SCT Valid (SCTVLD):</b> If this bit is set to '1', then the contents of the Status Code Type field are valid. If this bit is cleared to '0', then the contents of the Status Code Type field are invalid.																																																
1	<b>FLBA Valid (FVLD):</b> If this bit is set to '1', then the contents of the Failing LBA field are valid. If this bit is cleared to '0', then the contents of the Failing LBA field are invalid.																																																
0	<b>NSID Valid (NSIDVLD):</b> If this bit is set to '1', then the contents of the Namespace Identifier field are valid. If this bit is cleared to '0', then the contents of the Namespace Identifier field are invalid.																																																
03	Reserved																																																

**Figure 212: Self-test Result Data Structure**

Bytes	Description					
11:04	<b>Power On Hours (POH):</b> This field indicates the number of power-on hours at the time the device self-test operation was completed or aborted. This does not include time that the controller was powered and in a low power state condition.					
15:12	<b>Namespace Identifier (NSID):</b> This field indicates the namespace that the Failing LBA occurred on. The contents of this field are valid only when the NSID Valid bit is set to '1'.					
23:16	<b>Failing LBA (FLBA):</b> This field is I/O Command Set specific and is described in the applicable I/O Command Set specification.  NOTE: The original field name has been retained for historical continuity.					
24	<b>Status Code Type (STCT):</b> This field may contain additional information related to errors or conditions.					
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:3</td> <td>Reserved</td> </tr> <tr> <td>2:0</td> <td><b>Additional Status Code Info (ASTCI):</b> This field may contain additional information relating to errors or conditions that occurred during the device self-test operation represented in the same format used in the Status Code Type field of the completion queue entry (refer to Figure 101). The contents of this field are valid only when the SCT Valid bit is set to '1'.</td> </tr> </tbody> </table>	Bits	Description	7:3	Reserved	2:0
Bits	Description					
7:3	Reserved					
2:0	<b>Additional Status Code Info (ASTCI):</b> This field may contain additional information relating to errors or conditions that occurred during the device self-test operation represented in the same format used in the Status Code Type field of the completion queue entry (refer to Figure 101). The contents of this field are valid only when the SCT Valid bit is set to '1'.					
25	<b>Status Code (STC):</b> This field may contain additional information relating to errors or conditions that occurred during the device self-test operation represented in the same format used in the Status Code field of the completion queue entry (refer to section 4.2.3). The contents of this field are valid only when the SC Valid bit is set to '1'.					
27:26	<b>Vendor Specific (VS):</b> This field is vendor specific.					

**5.1.12.1.8 Telemetry Host-Initiated (Log Page Identifier 07h)**

This log page consists of a header (i.e., bytes 511:0 of the log page) describing the log and zero or more Telemetry Data Blocks (refer to section 8.1.27). The header shall always be available even if there is no Telemetry Host-Initiated Data available. All Telemetry Data Blocks are 512 bytes in size. The controller shall initiate a capture of the internal controller state or internal NVM subsystem state to this log page if the controller processes a Get Log Page command for this log with the Create Telemetry Host-Initiated Data bit set to '1' in the Log Specific Parameter field. If the host specifies a Log Page Offset Lower value that is not a multiple of 512 bytes in the Get Log Page command for this log, then the controller shall abort the command with a status code of Invalid Field in Command.

The Log Specific Parameter field in Command Dword 10 (refer to Figure 197) for this log page is defined in Figure 213.

**Figure 213: Telemetry Host-Initiated Log Specific Parameter Field**

Bits	Description														
14:12	Reserved														
11:09	<b>Maximum Created Data Area (MCDA):</b> If the MCDAS bit is set to '1' in the LID Specific Parameter field (refer to Figure 215) and the Create Telemetry Host-Initiated Data bit is set to '1', then this field specifies the data areas the host is requesting to be created in the log page. Data areas not requested shall not be created.														
	<table border="1"> <thead> <tr> <th>Values</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>The controller determines the data areas to be created in the log page.</td> </tr> <tr> <td>001b</td> <td>Data Area 1</td> </tr> <tr> <td>010b</td> <td>Data Area 1 through Data Area 2</td> </tr> <tr> <td>011b</td> <td>Data Area 1 through Data Area 3</td> </tr> <tr> <td>100b</td> <td>Data Area 1 through Data Area 4</td> </tr> <tr> <td>101b to 111b</td> <td>Reserved</td> </tr> </tbody> </table>	Values	Description	000b	The controller determines the data areas to be created in the log page.	001b	Data Area 1	010b	Data Area 1 through Data Area 2	011b	Data Area 1 through Data Area 3	100b	Data Area 1 through Data Area 4	101b to 111b	Reserved
	Values	Description													
	000b	The controller determines the data areas to be created in the log page.													
	001b	Data Area 1													
	010b	Data Area 1 through Data Area 2													
	011b	Data Area 1 through Data Area 3													
100b	Data Area 1 through Data Area 4														
101b to 111b	Reserved														
If the MCDAS bit is cleared to '0' or the Create Telemetry Host-Initiated Data bit is cleared to '0', then this field shall be ignored by the controller.															

**Figure 213: Telemetry Host-Initiated Log Specific Parameter Field**

Bits	Description
08	<p><b>Create Telemetry Host-Initiated Data (CTHID):</b> If this bit is set to '1', then the controller shall capture the Telemetry Host-Initiated Data representing the internal state of the controller at the time the associated Get Log Page command is processed. If this bit is cleared to '0', then the controller shall not update the Telemetry Host-Initiated Data. The Host-Initiated Data shall not change until the controller processes:</p> <ul style="list-style-type: none"> <li>a) a subsequent Telemetry Host-Initiated Log with this bit set to '1';</li> <li>b) a Firmware Commit command; or</li> <li>c) a power on reset.</li> </ul>

The Telemetry Host-Initiated Data consists of:

- a) Three areas, if the Data Area 4 Support (DA4S) bit is cleared to '0' in the Log Page Attributes field: Telemetry Host-Initiated Data Area 1, Telemetry Host-Initiated Data Area 2, and Telemetry Host-Initiated Data Area 3; or
- b) Four areas, if the DA4S bit is set to '1' in the Log Page Attributes field: Telemetry Host-Initiated Data Area 1, Telemetry Host-Initiated Data Area 2, Telemetry Host-Initiated Data Area 3 and Telemetry Host-Initiated Data Area 4.

All areas start at Telemetry Host-Initiated Data Area Block 1. The last block of each area is indicated in Telemetry Host-Initiated Data Area y Last Block, respectively. The telemetry data captured and the size of that data is implementation dependent.

The size of the log page is variable and:

- If the DA4S bit is cleared to '0' in the Log Page Attributes field, the size may be calculated using the Telemetry Host-Initiated Data Area 3 Last Block field.
- If the DA4S bit is set to '1' in the Log Page Attributes field and the Extended Telemetry Data Area 4 Supported (ETDAS) field is set to 1h in the Host Behavior Support feature (refer to section 5.1.25.1.14), then the size of the log page may be calculated using the Telemetry Host-Initiated Data Area 4 Last Block field.
- If the DA4S bit is set to '1' in the Log Page Attributes field and the Extended Telemetry Data Area 4 Supported (ETDAS) field is cleared to 0h in the Host Behavior Support feature (refer to section 5.1.25.1.14), then the size of the log page may be calculated using the Telemetry Host-Initiated Data Area 3 Last Block field.

The controller shall return data for all blocks requested:

- If the DA4S bit is cleared to '0' in the Log Page Attributes field, then the data beyond the last block in Telemetry Host-Initiated Data Area 3 Last Block is undefined.
- If the DA4S bit is set to '1' in the Log Page Attributes field and the Extended Telemetry Data Area 4 Supported (ETDAS) field is set to 1h in the Host Behavior Support feature, then the data beyond the last block in Telemetry Host-Initiated Data Area 4 Last Block is undefined.
- If the DA4S bit is set to '1' in the Log Page Attributes field and the Extended Telemetry Data Area 4 Supported (ETDAS) field is cleared to 0h in the Host Behavior Support feature, then the data beyond the last block in Telemetry Host-Initiated Data Area 3 Last Block is undefined.

If the host requests a data transfer that is not a multiple of 512 bytes, then the controller shall return an error of Invalid Field in Command.

If the MCDAS bit (refer to Figure 215) is set to '1' and a Get Log Page command with the Create Telemetry Host-Initiated Data bit is set to '1', then the maximum data area to be created in the Telemetry Host-Initiated log page shall be less than or equal to the MCDA field in the Log Specific Parameter field in Command Dword 10 (refer to Figure 213).

**Figure 214: Telemetry Host-Initiated Log Page**

Bytes	Description										
00	<b>Log Page Identifier (LID):</b> This field shall be set to 07h.										
04:01	Reserved										
07:05	<b>IEEE OUI Identifier (IEEE):</b> Contains the Organization Unique Identifier (OUI) for the controller vendor that is able to interpret the data. If this field is cleared to 0h, then no IEEE OUI Identifier is present. The OUI shall be a valid IEEE/RAC assigned identifier that is registered at <a href="http://standards.ieee.org/develop/regauth/oui/public.html">http://standards.ieee.org/develop/regauth/oui/public.html</a> .										
09:08	<b>Telemetry Host-Initiated Data Area 1 Last Block (THDA1LB):</b> Contains the value of the last block of Telemetry Host-Initiated Data Area 1. If the Telemetry Host-Initiated Data Area 1 does not contain data, then this field shall be cleared to 0h.  If this field is not 0h, then Telemetry Host-Initiated Data Area 1 begins at block 1h and ends at the block indicated in this field.										
11:10	<b>Telemetry Host-Initiated Data Area 2 Last Block (THDA2LB):</b> Contains the value of the last block of Telemetry Host-Initiated Data Area 2. This value shall be greater than or equal to the value in the Telemetry Host-Initiated Data Area 1 Last Block field.  If this field is not 0h, then Telemetry Host-Initiated Data Area 2 begins at block 1h and ends at the block indicated in this field.										
13:12	<b>Telemetry Host-Initiated Data Area 3 Last Block (THDA3LB):</b> Contains the value of the last block of Telemetry Host-Initiated Data Area 3. This value shall be greater than or equal to the value in the Telemetry Host-Initiated Data Area 2 Last Block field.  If this field is not 0h, then Telemetry Host-Initiated Data Area 3 begins at block 1h and ends at the block contained in this field.										
15:14	Reserved										
19:16	<b>Telemetry Host-Initiated Data Area 4 Last Block (THDA4LB):</b> Contains the value of the last block of Telemetry Host-Initiated Data Area 4. If the Data Area 4 Support (DA4S) bit is set to '1' in the Log Page Attributes field is set to '1', then this value shall be greater than or equal to the value in the Telemetry Host-Initiated Data Area 3 Last Block field.  If this field is not 0h, then Telemetry Host-Initiated Data Area 4 begins at block 1h and ends at the block contained in this field.										
379:20	Reserved										
380	<p><b>Telemetry Host-Initiated Scope (THS):</b> This field shall indicate the scope of the Telemetry Host-Initiated log page. Implementations compliant with versions of this specification later than 2.0 shall not set this field to a value of 0h.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Value</th> <th>Scope</th> </tr> </thead> <tbody> <tr> <td>00h</td> <td>Not reported</td> </tr> <tr> <td>01h</td> <td>Controller</td> </tr> <tr> <td>02h</td> <td>NVM subsystem</td> </tr> <tr> <td>03h to FFh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Scope	00h	Not reported	01h	Controller	02h	NVM subsystem	03h to FFh	Reserved
Value	Scope										
00h	Not reported										
01h	Controller										
02h	NVM subsystem										
03h to FFh	Reserved										
381	<b>Telemetry Host-Initiated Data Generation Number (THDGN):</b> Contains a value that is incremented each time the internal controller state or internal NVM subsystem state for this log page is captured. If the value of this field is FFh, then the field shall be cleared to 0h when incremented (i.e., rolls over to 0h).										
382	<b>Telemetry Controller-Initiated Data Available (TCDA):</b> Contains the value of Telemetry Controller-Initiated Data Available field in the Telemetry Controller-Initiated log (refer to Figure 216).										
383	<b>Telemetry Controller-Initiated Data Generation Number (TCDGN):</b> Contains the value of the Telemetry Controller-Initiated Data Generation Number field in the Telemetry Controller-Initiated log (refer to Figure 216).										
511:384	<b>Reason Identifier (RID):</b> Contains a vendor specific identifier that describes the operating conditions of the controller at the time of capture. The Reason Identifier field should provide an identification of unique operating conditions of the controller.										
1023:512	<b>Telemetry Host-Initiated Data Block 1 (THDB1):</b> Contains Telemetry Data Block 1 for the Telemetry Host-Initiated Log.										
1535:1024	<b>Telemetry Host-Initiated Data Block 2 (THDB2):</b> Contains Telemetry Data Block 2 for the Telemetry Host-Initiated Log.										

**Figure 214: Telemetry Host-Initiated Log Page**

Bytes	Description
...	...
$(n*512)+511:(n*512)$	<b>Telemetry Host-Initiated Data Block <math>n</math> (THDBN):</b> Contains Telemetry Data Block $n$ for the Telemetry Host-Initiated Log.

#### 5.1.12.1.8.1 Telemetry Host-Initiated LID Specific Parameter Field

Figure 215 specifies the format for the LID Specific Parameter field in the Supported Log Pages log page (refer to section 5.1.12.1.1) for the Telemetry Host-Initiated log page.

**Figure 215: Telemetry Host-Initiated Log Page - LID Specific Parameter Field**

Bits	Description
15:1	Reserved
0	<b>Maximum Created Data Area Support (MCDAS):</b> If this bit is set to '1', then the controller supports the Maximum Created Data Area field in Log Specific Parameter field for a Telemetry Host-Initiated log page. If this bit is cleared to '0', then the controller does not support the Maximum Created Data Area field in Log Specific Parameter field for a Telemetry Host-Initiated log page.

#### 5.1.12.1.9 Telemetry Controller-Initiated (Log Page Identifier 08h)

This log page consists of a header (i.e., bytes 511:0 of the log page) describing the log and zero or more Telemetry Data Blocks (refer to section 8.1.27). The header shall always be available even if there is no Telemetry Controller-Initiated Data available. All Telemetry Data Blocks are 512 bytes in size. This log is a controller-initiated capture of the controller's internal state or NVM subsystem's internal state. The Telemetry Controller-Initiated Data for Data Area 1 through Data Area 3 shall persist across all resets. The Telemetry Controller-Initiated Data for Data Area 4 may persist across Controller Level Resets. If the host specifies a Log Page Offset Lower value that is not a multiple of 512 bytes in the Get Log Page command for this log, then the controller shall return an error of Invalid Field in Command.

The Telemetry Controller-Initiated Data consists of:

- a) three areas, if the Data Area 4 Support (DA4S) bit is cleared to '0' in the Log Page Attributes field: Telemetry Controller-Initiated Data Area 1, Telemetry Controller-Initiated Data Area 2, and Telemetry Controller-Initiated Data Area 3; or
- b) four areas, if the DA4S bit is set to '1' in the Log Page Attributes field: Telemetry Controller-Initiated Data Area 1, Telemetry Controller-Initiated Data Area 2, Telemetry Controller-Initiated Data Area 3 and Telemetry Controller-Initiated Data Area 4.

All areas start at Telemetry Controller-Initiated Data Area Block 1. The last block of each area is indicated in the Telemetry Controller-Initiated Data Area  $y$  Last Block, respectively. The telemetry data captured and its size is implementation dependent.

The size of the log page is variable and:

- If the DA4S bit is cleared to '0' in the Log Page Attributes field, the size may be calculated using the Telemetry Controller-Initiated Data Area 3 Last Block field.
- If the DA4S bit is set to '1' in the Log Page Attributes field and the Extended Telemetry Data Area 4 Supported (ETDAS) field is set to 1h in the Host Behavior Support feature (refer to section 5.1.25.1.14), then the size of the log page may be calculated using the Telemetry Controller-Initiated Data Area 4 Last Block field.
- If the DA4S bit is set to '1' in the Log Page Attributes field and the Extended Telemetry Data Area 4 Supported (ETDAS) field is cleared to 0h in the Host Behavior Support feature (refer to section 5.1.25.1.14), then the size of the log page may be calculated using the Telemetry Controller-Initiated Data Area 3 Last Block field.

The controller shall return data for all blocks requested:



- If the DA4S bit is cleared to '0' in the Log Page Attributes field, then the data beyond the last block in Telemetry Controller-Initiated Data Area 3 Last Block is undefined.
- If the DA4S bit is set to '1' in the Log Page Attributes field, then the data beyond the last block in Telemetry Controller-Initiated Data Area 4 Last Block is undefined.
- If the DA4S bit is set to '1' in the Log Page Attributes field and the Extended Telemetry Data Area 4 Supported (ETDAS) field is cleared to 0h in the Host Behavior Support feature, then the data beyond the last block in Telemetry Controller-Initiated Data Area 3 Last Block is undefined.

If the host requests a data transfer that is not a multiple of 512 bytes, then the controller shall abort the command with the status code of Invalid Field in Command.

**Figure 216: Telemetry Controller-Initiated Log Page**

Bytes	Description										
00	<b>Log Page Identifier (LID):</b> This field shall be set to 08h.										
04:01	Reserved										
07:05	<b>IEEE OUI Identifier (IEEE):</b> Contains the Organization Unique Identifier (OUI) for the controller vendor that is able to interpret the data. If this field is cleared to 0h, then no IEEE OUI Identifier is present. The OUI shall be a valid IEEE/RAC assigned identifier that is registered at <a href="http://standards.ieee.org/develop/regauth/oui/public.html">http://standards.ieee.org/develop/regauth/oui/public.html</a> .										
09:08	<b>Telemetry Controller-Initiated Data Area 1 Last Block (TCDA1LB):</b> Contains the value of the last block of Telemetry Controller-Initiated Data Area 1. If the Telemetry Controller-Initiated Data Area 1 does not contain data, then this field shall be cleared to 0h.  If this field is not 0h, then Telemetry Controller-Initiated Data Area 1 begins at block 1 and ends at the block indicated in this field.										
11:10	<b>Telemetry Controller-Initiated Data Area 2 Last Block (TCDA2LB):</b> Contains the value of the last block of Telemetry Controller-Initiated Data Area 2. This value shall be greater than or equal to the value in the Telemetry Controller-Initiated Data Area 1 Last Block field.  If this field is not 0h, then Telemetry Controller-Initiated Data Area 2 begins at block 1h and ends at the block indicated in this field.										
13:12	<b>Telemetry Controller-Initiated Data Area 3 Last Block (TCDA3LB):</b> Contains the value of the last block of Telemetry Controller-Initiated Data Area 3. This value shall be greater than or equal to the value in the Telemetry Controller-Initiated Data Area 2 Last Block field.  If this field is not 0h, then Telemetry Controller-Initiated Data Area 3 begins at block 1h and ends at the block indicated in this field.										
15:14	Reserved										
19:16	<b>Telemetry Controller-Initiated Data Area 4 Last Block (TCDA4LB):</b> Contains the value of the last block of Telemetry Controller-Initiated Data Area 4. If the DA4S bit is set to '1' in the Log Page Attributes field, then this value shall be greater than or equal to the value in the Telemetry Controller-Initiated Data Area 3 Last Block field.  If this field is not 0h, then Telemetry Controller-Initiated Data Area 4 begins at block 1h and ends at the block contained in this field.										
380:20	Reserved										
381	<p><b>Telemetry Controller-Initiated Scope (TCS):</b> This field shall indicate the scope of the Telemetry Controller-Initiated log page. Implementations compliant with versions of this specification later than 2.0 shall not set this field to a value of 0h.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Value</th> <th>Scope</th> </tr> </thead> <tbody> <tr> <td>00h</td> <td>Not reported</td> </tr> <tr> <td>01h</td> <td>Controller</td> </tr> <tr> <td>02h</td> <td>NVM subsystem</td> </tr> <tr> <td>03h to FFh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Scope	00h	Not reported	01h	Controller	02h	NVM subsystem	03h to FFh	Reserved
Value	Scope										
00h	Not reported										
01h	Controller										
02h	NVM subsystem										
03h to FFh	Reserved										

**Figure 216: Telemetry Controller-Initiated Log Page**

Bytes	Description
382	<p><b>Telemetry Controller-Initiated Data Available (TCDA):</b> If this field is cleared to 0h, then the log page consists of only the 512-byte header and does not contain saved internal controller state or saved internal NVM subsystem state available to be reported in response to a Get Log Page command issued to the Admin Submission Queue.</p> <p>If this field is set to 1h, then the log page contains saved internal controller state or saved internal NVM subsystem state to be reported in response to a Get Log Page command issued to the Admin Submission Queue. If this field is set to 1h, then it shall not be cleared to 0h until a Get Log Page command with Retain Asynchronous Event bit cleared to '0' for the Telemetry Controller-Initiated log page completes successfully. This value is persistent across power states and reset.</p> <p>Regardless of the value of this field, the log page may contain saved internal controller state or saved internal NVM subsystem state available to be reported in response to a Get Log Page command issued out-of-band to a Management Endpoint (refer to the NVM Express Management Interface Specification).</p> <p>Other values are reserved.</p>
383	<p><b>Telemetry Controller-Initiated Data Generation Number (TCDGN):</b> Contains a value that is incremented each time the internal controller state or the NVM subsystem state for this log page is captured. If the value of this field is FFh, then the field shall be cleared to 0h when incremented (i.e., rolls over to 0h). This field is persistent across power cycles.</p>
511:384	<p><b>Reason Identifier (RID):</b> Contains a vendor specific identifier that describes the operating conditions of the controller at the time of capture. The Controller-Initiated Reason Identifier field should provide an identification of unique operating conditions of the controller.</p>
1023:512	<p><b>Telemetry Controller-Initiated Data Block 1 (TCDB1):</b> Contains Telemetry Data Block 1 for the Telemetry Controller -Initiated Log captured at a vendor specific time.</p>
1535:1024	<p><b>Telemetry Controller-Initiated Data Block 2 (TCDB2):</b> Contains Telemetry Data Block 2 for the Telemetry Controller -Initiated Log captured at a vendor specific time.</p>
...	...
( $n \times 512$ ):511:( $n \times 512$ )	<p><b>Telemetry Controller-Initiated Data Block <math>n</math> (TCDB<math>n</math>):</b> Contains Telemetry Data Block <math>n</math> for the Telemetry Controller-Initiated log captured at a vendor specific time.</p>

#### 5.1.12.1.10 Endurance Group Information (Log Page Identifier 09h)

This log page is used to provide endurance information based on the Endurance Group (refer to section 3.2.3). An Endurance Group contains capacity that may be allocated to zero or more NVM Sets. Capacity that has not been allocated to an NVM Set is unallocated Endurance Group capacity. The information provided is over the life of the Endurance Group. The Endurance Group Identifier is specified in the Log Specific Identifier field in Command Dword 11 of the Get Log Page command as defined in Figure 217. The log page is 512 bytes in size.

**Figure 217: Endurance Group Identifier - Log Specific Identifier**

Bits	Description
15:00	<p><b>Endurance Group Identifier (ENDGID):</b> This field specifies the identifier for the Endurance Group (refer to section 3.2.3) used for this log page.</p>

**Figure 218: Endurance Group Information Log Page**

Bytes	Description												
00	<p><b>Endurance Group Critical Warning (EGCW):</b> This field indicates critical warnings for the state of the Endurance Group. Each bit corresponds to a critical warning type; multiple bits may be set to '1'. If a bit is cleared to '0', then that critical warning does not apply. Critical warnings may result in an asynchronous event notification to the host. Bits in this field represent the state at the time the Get Log Page command is processed and may not reflect the state at the time a related asynchronous event notification, if any, occurs or occurred.</p> <p>If a bit is set to '1' in all Endurance Groups in the NVM subsystem, then the corresponding bit shall be set to '1' in the Critical Warning field of the SMART / Health Information log page (refer to Figure 206).</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:4</td> <td>Reserved</td> </tr> <tr> <td>3</td> <td><b>Endurance Group Read-Only (EGRO):</b> If this bit is set to '1', then all namespaces in the Endurance Group have been placed in read only mode for reasons other than a change in the write protect state of the namespace. The controller shall not set this bit to '1' if the read-only condition on the Endurance Group is a result of a change in the write protection state of all namespaces in the Endurance Group.</td> </tr> <tr> <td>2</td> <td><b>Endurance Group Degraded Reliability (EGDR):</b> If this bit is set to '1', then the Endurance Group reliability has been degraded due to significant media related errors or any internal error that degrades NVM subsystem reliability.</td> </tr> <tr> <td>1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td><b>Endurance Group Available Spare Below (EGASB):</b> If this bit is set to '1', then the available spare capacity of the Endurance Group has fallen below the threshold.</td> </tr> </tbody> </table>	Bits	Description	7:4	Reserved	3	<b>Endurance Group Read-Only (EGRO):</b> If this bit is set to '1', then all namespaces in the Endurance Group have been placed in read only mode for reasons other than a change in the write protect state of the namespace. The controller shall not set this bit to '1' if the read-only condition on the Endurance Group is a result of a change in the write protection state of all namespaces in the Endurance Group.	2	<b>Endurance Group Degraded Reliability (EGDR):</b> If this bit is set to '1', then the Endurance Group reliability has been degraded due to significant media related errors or any internal error that degrades NVM subsystem reliability.	1	Reserved	0	<b>Endurance Group Available Spare Below (EGASB):</b> If this bit is set to '1', then the available spare capacity of the Endurance Group has fallen below the threshold.
	Bits	Description											
	7:4	Reserved											
	3	<b>Endurance Group Read-Only (EGRO):</b> If this bit is set to '1', then all namespaces in the Endurance Group have been placed in read only mode for reasons other than a change in the write protect state of the namespace. The controller shall not set this bit to '1' if the read-only condition on the Endurance Group is a result of a change in the write protection state of all namespaces in the Endurance Group.											
	2	<b>Endurance Group Degraded Reliability (EGDR):</b> If this bit is set to '1', then the Endurance Group reliability has been degraded due to significant media related errors or any internal error that degrades NVM subsystem reliability.											
	1	Reserved											
0	<b>Endurance Group Available Spare Below (EGASB):</b> If this bit is set to '1', then the available spare capacity of the Endurance Group has fallen below the threshold.												
01	<p><b>Endurance Group Features (EGFEAT):</b> This field defines features of the Endurance Group.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td><b>Endurance Group Media (EGRMEDIA):</b> If this bit is set to '1', then the Endurance Group stores data on rotational media (refer to section 8.1.23). If this bit is cleared to '0', then the Endurance Group does not store data on rotational media.</td> </tr> </tbody> </table>	Bits	Description	7:1	Reserved	0	<b>Endurance Group Media (EGRMEDIA):</b> If this bit is set to '1', then the Endurance Group stores data on rotational media (refer to section 8.1.23). If this bit is cleared to '0', then the Endurance Group does not store data on rotational media.						
	Bits	Description											
	7:1	Reserved											
0	<b>Endurance Group Media (EGRMEDIA):</b> If this bit is set to '1', then the Endurance Group stores data on rotational media (refer to section 8.1.23). If this bit is cleared to '0', then the Endurance Group does not store data on rotational media.												
02	Reserved												
03	<b>Available Spare (AVSP):</b> Contains a normalized percentage (0% to 100%) of the remaining spare capacity available for the Endurance Group.												
04	<b>Available Spare Threshold (AVSPT):</b> If the Available Spare falls below the threshold indicated in this field, an asynchronous event completion may occur. The value is indicated as a normalized percentage (0% to 100%). The values 101 to 255 are reserved.												
05	<p><b>Percentage Used (PUSED):</b> Contains a vendor specific estimate of the percentage of life used for the Endurance Group based on the actual usage and the manufacturer's prediction of NVM life. A value of 100 indicates that the estimated endurance of the NVM in the Endurance Group has been consumed, but may not indicate an NVM failure. The value is allowed to exceed 100. Percentages greater than 254 shall be represented as 255. This value shall be updated once per power-on hour when the controller is not in a sleep state.</p> <p>Refer to the JEDEC JESD218B-02 standard for SSD device life and endurance measurement techniques.</p>												
07:06	<b>Domain Identifier (DID):</b> This field indicates the identifier of the domain that contains this Endurance Group. If the NVM subsystem supports multiple domains, this field shall be set to a non-zero value. If this field is cleared to 0h, then the NVM subsystem does not support multiple domains.												
31:08	Reserved												
47:32	<p><b>Endurance Estimate (EE):</b> This field is an estimate of the total number of data bytes that may be written to the Endurance Group over the lifetime of the Endurance Group assuming a write amplification of 1 (i.e., no increase in the number of write operations performed by the device beyond the number of write operations requested by a host). This value is reported in billions (i.e., a value of 1 corresponds to 1,000,000,000 bytes written) and is rounded up (e.g., one indicates the number of bytes written is from 1 to 1,000,000,000, three indicates the number of bytes written is from 2,000,000,001 to 3,000,000,000).</p> <p>A value of 0h indicates that the controller does not report an Endurance Estimate.</p>												

**Figure 218: Endurance Group Information Log Page**

Bytes	Description
63:48	<b>Data Units Read (DUR):</b> Contains the total number of data bytes that have been read from the Endurance Group. This value does not include controller reads due to internal operations such as garbage collection. This value is reported in billions (i.e., a value of 1 corresponds to 1,000,000,000 bytes read) and is rounded up (e.g., one indicates the number of bytes read is from 1 to 1,000,000,000, three indicates the number of bytes read is from 2,000,000,001 to 3,000,000,000).  A value of 0h indicates that the controller does not report the number of Data Units Read.
79:64	<b>Data Units Written (DUW):</b> Contains the total number of data bytes that have been written to the Endurance Group. This value does not include controller writes due to internal operations such as garbage collection. This value is reported in billions (i.e., a value of 1 corresponds to 1,000,000,000 bytes written) and is rounded up (e.g., one indicates the number of bytes written is from 1 to 1,000,000,000, three indicates the number of bytes written is from 2,000,000,001 to 3,000,000,000).  A value of 0h indicates that the controller does not report the number of Data Units Written.
95:80	<b>Media Units Written (MUW):</b> Contains the total number of data bytes that have been written to the Endurance Group including both host and controller writes (e.g., garbage collection). This value is reported in billions (i.e., a value of 1 corresponds to 1,000,000,000 bytes written) and is rounded up (e.g., one indicates the number of bytes written is from 1 to 1,000,000,000, three indicates the number of bytes written is from 2,000,000,001 to 3,000,000,000).  A value of 0h indicates that controller does not report the number of Media Units Written.
111:96	<b>Host Read Commands (HRC):</b> Contains the number of Endurance Group Host Read Commands completed by the controller.  Refer to the specific I/O Command Set specification for the list of Endurance Group Host Read Commands that affect this field.
127:112	<b>Host Write Commands (HWC):</b> Contains the number of User Data Out Commands completed by the controller.  Refer to the specific I/O Command Set specification for the list of User Data Out Commands that affect this field.
143:128	<b>Media and Data Integrity Errors (MDIE):</b> Contains the number of occurrences where the controller detected an unrecovered data integrity error for the Endurance Group. Errors such as uncorrectable ECC, CRC checksum failure, or LBA tag mismatch are included in this field.
159:144	<b>Number of Error Information Log Entries (NEILE):</b> Contains the number of Error Information Log Entries over the life of the controller for the Endurance Group.
175:160	<b>Total Endurance Group Capacity (TEGCAP):</b> This field indicates the total NVM capacity in this Endurance Group. The value is in bytes. If this field is cleared to 0h, then the NVM subsystem does not report the total NVM capacity in this Endurance Group.
191:176	<b>Unallocated Endurance Group Capacity (UEGCAP):</b> This field indicates the unallocated NVM capacity in this Endurance Group. The value is in bytes. If this field is cleared to 0h, then the NVM subsystem does not report the unallocated NVM capacity in this Endurance Group.
511:192	Reserved

**5.1.12.1.11 Predictable Latency Per NVM Set (Log Page Identifier 0Ah)**

This log page may be used to determine the current window for the specified NVM Set when Predictable Latency Mode is enabled and any events that have occurred for the specified NVM Set. There is one log page for each NVM Set when Predictable Latency Mode is supported. The NVM Set for which the log page is to be returned is specified in the Log Specific Identifier field in Command Dword 11 of the Get Log Page command as defined in Figure 219. The log page is 512 bytes in size.

**Figure 219: NVM Set Identifier – Log Specific Identifier**

Bits	Description
15:00	<b>NVM Set Identifier (NVMSETID):</b> This field specifies the identifier for the NVM Set (refer to section 3.2.2) used for this log page.

The log page indicates typical values and reliable estimates for attributes associated with the Deterministic Window and the Non-Deterministic Window of the specified NVM Set. The Typical, Maximum, and Minimum values are static and worst-case values over the lifetime of the NVM subsystem.

After the controller successfully completes a read of this log page with Retain Asynchronous Event bit cleared to '0', then reported events are cleared to '0' for the specified NVM Set and the field corresponding to the specified NVM Set is cleared to '0' in the Predictable Latency Event Aggregate log page. Coordination between two or more hosts is beyond the scope of this specification.

**Figure 220: Predictable Latency Per NVM Set Log Page**

Bytes	Description														
00	<b>Status (STSNVMS):</b> This field indicates the status of the specified NVM Set.														
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:3</td> <td>Reserved</td> </tr> </tbody> </table>	Bits	Description	7:3	Reserved										
	Bits	Description													
	7:3	Reserved													
2:0	<b>Predictable Latency Mode Window (PLMW):</b> This field indicates the window for the NVM Set when Predictable Latency Mode is enabled.														
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>Not used (Predictable Latency Mode not enabled)</td> </tr> <tr> <td>001b</td> <td>Deterministic Window (DTWIN)</td> </tr> <tr> <td>010b</td> <td>Non-Deterministic Window (NDWIN)</td> </tr> <tr> <td>011b to 111b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	000b	Not used (Predictable Latency Mode not enabled)	001b	Deterministic Window (DTWIN)	010b	Non-Deterministic Window (NDWIN)	011b to 111b	Reserved				
	Value	Definition													
	000b	Not used (Predictable Latency Mode not enabled)													
001b	Deterministic Window (DTWIN)														
010b	Non-Deterministic Window (NDWIN)														
011b to 111b	Reserved														
01	Reserved														
03:02	<b>Event Type (ETYP):</b> This field specifies the event(s) that occurred for the NVM Set indicated. Multiple bits may be set to '1'. All bits are cleared to '0' after the log page is read with Retain Asynchronous Event bit cleared to '0'.														
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15</td> <td><b>Deterministic Excursion Autonomous Transition (DEAT):</b> Autonomous transition from DTWIN to NDWIN due to Deterministic Excursion</td> </tr> <tr> <td>14</td> <td><b>Max Value Exceeded Autonomous Transition (MVEAT):</b> Autonomous transition from DTWIN to NDWIN due to typical or maximum value exceeded</td> </tr> <tr> <td>13:03</td> <td>Reserved</td> </tr> <tr> <td>02</td> <td><b>DTWIN Time Warning (DTTW)</b></td> </tr> <tr> <td>01</td> <td><b>DTWIN Writes Warning (DTWW)</b></td> </tr> <tr> <td>00</td> <td><b>DTWIN Reads Warning (DTRW)</b></td> </tr> </tbody> </table>	Bits	Description	15	<b>Deterministic Excursion Autonomous Transition (DEAT):</b> Autonomous transition from DTWIN to NDWIN due to Deterministic Excursion	14	<b>Max Value Exceeded Autonomous Transition (MVEAT):</b> Autonomous transition from DTWIN to NDWIN due to typical or maximum value exceeded	13:03	Reserved	02	<b>DTWIN Time Warning (DTTW)</b>	01	<b>DTWIN Writes Warning (DTWW)</b>	00	<b>DTWIN Reads Warning (DTRW)</b>
	Bits	Description													
	15	<b>Deterministic Excursion Autonomous Transition (DEAT):</b> Autonomous transition from DTWIN to NDWIN due to Deterministic Excursion													
	14	<b>Max Value Exceeded Autonomous Transition (MVEAT):</b> Autonomous transition from DTWIN to NDWIN due to typical or maximum value exceeded													
	13:03	Reserved													
	02	<b>DTWIN Time Warning (DTTW)</b>													
01	<b>DTWIN Writes Warning (DTWW)</b>														
00	<b>DTWIN Reads Warning (DTRW)</b>														
31:04	Reserved														
<b>Typical, Maximum, and Minimum Values</b>															
39:32	<b>DTWIN Reads Typical (DTWRT):</b> Indicates the typical number of 4 KiB random reads that may be performed in the Deterministic Window. Refer to section 8.1.18.														
47:40	<b>DTWIN Writes Typical (DTWWT):</b> Indicates the typical number of writes in units of the Optimal Write Size that may be performed in the Deterministic Window. Refer to section 8.1.18.														
55:48	<b>DTWIN Time Maximum (DTWTM):</b> Indicates the maximum time in milliseconds that the NVM Set is able to remain in a Deterministic Window before entering a Non-Deterministic Window. Refer to section 8.1.18.														
63:56	<b>NDWIN Time Minimum High (NTWTMH):</b> Indicates the minimum time in milliseconds that the NVM Set needs to remain in the Non-Deterministic Window before entering a Deterministic Window. This is the time necessary to prepare for remaining in the Deterministic Window for DTWIN Time Maximum. Refer to section 8.1.18.														
71:64	<b>NDWIN Time Minimum Low (NTWTML):</b> Indicates the minimum time in milliseconds that the NVM Set needs to remain in the Non-Deterministic Window before entering a Deterministic Window. This is regardless of the amount of time spent in the previous Deterministic Window. Refer to section 8.1.18.														
127:72	Reserved														
<b>Reliable Estimates</b>															
135:128	<b>DTWIN Reads Estimate (DTWRE):</b> Indicates a reliable estimate of the number of 4 KiB random reads remaining in the current Deterministic Window, if applicable. This value decrements from DTWIN Reads Typical to 0h based on host read activity and operating conditions. Refer to section 8.1.18.1.														

**Figure 220: Predictable Latency Per NVM Set Log Page**

Bytes	Description
143:136	<b>DTWIN Writes Estimate (DTWWE):</b> Indicates a reliable estimate of the number of writes in units of the Optimal Write Size remaining in the current Deterministic Window, if applicable. This value decrements from DTWIN Writes Typical to 0h based on host write activity and operating conditions. Refer to section 8.1.18.1.
151:144	<b>DTWIN Time Estimate (DTWTE):</b> Indicates a reliable estimate of the time in milliseconds remaining in the current Deterministic Window, if applicable. Refer to section 8.1.18.1.
511:152	Reserved

#### 5.1.12.1.12 Predictable Latency Event Aggregate (Log Page Identifier 0Bh)

This log page indicates if a Predictable Latency Event (refer to section 8.1.18) has occurred for a particular NVM Set. If a Predictable Latency Event has occurred, the details of the particular event are included in the Predictable Latency Per NVM Set log page for that NVM Set. An asynchronous event is generated when an entry for an NVM Set is newly added to this log page.

This log page shall not contain an entry (i.e., an NVM Set Identifier) that is cleared to 0h.

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

The log page size is limited by the NVM Set Identifier Maximum value reported in the Identify Controller data structure (refer to Figure 312). If the host reads beyond the end of the log page, zeroes are returned. The log page is defined in Figure 221.

**Figure 221: Predictable Latency Event Aggregate Log Page**

Bytes	Description
<b>Header</b>	
07:00	<b>Number of Entries (NUMENT):</b> This field indicates the number of entries in the list. The maximum number of entries in the list corresponds to the NVM Set Identifier Maximum field reported in the Identify Controller data structure. A value of 0h indicates there are no entries in the list.
<b>NVM Set Identifier List</b>	
09:08	<b>Entry 1:</b> Indicates the NVM Set that has a Predictable Latency Event pending that has the numerically smallest NVM Set Identifier, if any.
11:10	<b>Entry 2:</b> Indicates the NVM Set that has a Predictable Latency Event pending that has the second numerically smallest NVM Set Identifier, if any.
...	...
(NUMENT*2) + 9: (NEMENT*2) + 8	<b>Entry NUMENT:</b> Indicates the NVM Set that has a Predictable Latency Event pending that has the numerically largest NVM Set Identifier, if any.

#### 5.1.12.1.13 Asymmetric Namespace Access (Log Page Identifier 0Ch)

This log page consists of a header describing the log and descriptors containing the asymmetric namespace access information for ANA Groups (refer to section 8.1.1.2) that contain namespaces that are attached to the controller processing the command. If ANA Reporting (refer to section 8.1.1) is supported, this log page is supported. ANA Group Descriptors shall be returned in ascending ANA Group Identifier order.

If the Index Offset Supported bit is cleared to '0' in the LID Support and Effects data structure for this log page (refer to Figure 204), then:

- if the RGO bit is cleared to '0' in Command Dword 10, then the LPOL field in Command Dword 12 and the LPOU field in Command Dword 13 of the Get Log Page command should be cleared to 0h.

If the Index Offset Supported bit is set to '1' in the LID Supported and Effects data structure for this log page (refer to Figure 204) and the OT bit is set to '1', then an index value specified in the Log Page Offset Lower (LPOL) field (refer to Figure 199) and the Log Page Offset Upper (LPOU) field (refer to Figure 200) in the Get Log Page command shall be used to index to the header or an ANA Group Descriptor within the list of ANA Group Descriptors as shown in Figure 223 (e.g., specifying an index offset of 0 returns this log page starting at byte offset 0h, specifying an index offset of 1 returns this log page starting at the offset of ANA Group Descriptor 0, specifying an index offset of 2 returns this log page starting at the offset of ANA Group Descriptor 1, etc.). Due to the complexity of using a byte offset with a variable sized ANA Group Descriptor, the host should use an index offset by setting the OT bit to '1' in Command Dword 14 of the Get Log Page command.

If the host performs multiple Get Log Page commands to read the ANA log page (e.g., using the LPOL field or the LPOU field), the host should re-read the header of the log page and ensure that the Change Count field in the Asymmetric Namespace Access log matches the original value read. If it does not match, then the data captured is not consistent and the ANA log page should be re-read.

The Log Specific Parameter field in Command Dword 10 (refer to Figure 197) for this log page is defined in Figure 222.

**Figure 222: Asymmetric Namespace Access Log Specific Parameter Field**

Bits	Description
14:09	Reserved
08	<b>Return Groups Only (RGO):</b> If this bit is set to '1', then the controller shall return ANA Group Descriptors with the Number of NSID Values field in each ANA Group Descriptor cleared to 0h (i.e., no Namespace Identifiers are returned). If this bit is cleared to '0', then the controller shall return ANA Group Descriptors that contain the Namespace Identifiers of attached namespaces that are members of the ANA Group described by that ANA Group Descriptor and the Number of NSID Values field set to the number of Namespace Identifier values in that ANA Group Descriptor.

**Figure 223: Asymmetric Namespace Access Log Page**

Bytes	Description
<b>Header</b>	
07:00	<b>Change Count (CHGC):</b> This field contains a 64-bit incrementing Asymmetric Namespace Access log change count, indicating an identifier for this set of asymmetric namespace access information. The count starts at 0h following a Controller Level Reset and is incremented each time the contents of the log page change (e.g., not only if an Asymmetric Namespace Access Change Asynchronous Event Notification is generated). If the value of this field is FFFFFFFF_FFFFFFFFh, then the field shall be cleared to 0h when incremented (i.e., rolls over to 0h).
09:08	<b>Number of ANA Group Descriptors (NAGD):</b> This field indicates the number of ANA Group Descriptors available in the log page. The log page shall contain one ANA Group Descriptor for each ANA Group that contains namespaces that are attached to the controller.  If, for an ANA Group, there are no namespaces attached to the controller processing the command, then no ANA Group Descriptor is returned for that ANA Group (i.e., an ANA Group Descriptor is returned only if that ANA Group contains namespaces that are attached to the controller processing the command).  If no namespaces are attached to the controller, then the log page does not contain any ANA Group Descriptors and this field is cleared to 0h.
15:10	Reserved
<b>List of ANA Group Descriptors</b>	
n:16	<b>ANA Group Descriptor 0:</b> This is the first ANA Group Descriptor (refer to Figure 224), if any.
m:n+1	<b>ANA Group Descriptor 1:</b> This is the second ANA Group Descriptor (refer to Figure 224), if any.
...	...
x:y	<b>ANA Group Descriptor NAGD-1:</b> This is the last ANA Group Descriptor (refer to Figure 224), if any.

The format of the ANA Group Descriptor is defined in Figure 224. Namespace Identifiers shall be listed in ascending NSID order.

**Figure 224: ANA Group Descriptor format**

Bytes	Description																												
<b>Header</b>																													
03:00	<b>ANA Group ID (AGID):</b> The ANA Group Identifier associated with all namespaces in the ANA Group (refer to section 8.1.1.2) described by this ANA Group Descriptor.																												
07:04	<b>Number of NSID Values (NNV):</b> This field indicates the number of Namespace Identifier values in this ANA Group Descriptor.  If the RGO bit is set to '1', then this field is cleared to 0h.																												
15:08	<b>Change Count (CHGC):</b> This field contains a 64-bit incrementing count, indicating an identifier for the information contained in this ANA Group Descriptor. A value of 0h indicates that the controller does not report a Change Count for this ANA Group Descriptor. If a Change Count is reported, then the count starts at 1h following a Controller Level Reset and is incremented each time the data in this ANA Group Descriptor change. If the value of this field is FFFFFFFF_FFFFFFFFh, then the field shall be set to 1h when incremented (i.e., rolls over to 1h).  If this field contains 0h, the host should examine this ANA Group Descriptor for any changes and not use this field as an indicator that a change has occurred.																												
16	<b>Asymmetric Namespace Access State Attributes (ANASA):</b>																												
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>07:04</td> <td>Reserved</td> </tr> <tr> <td rowspan="6">03:00</td> <td><b>Asymmetric Namespace Access State (ANAS):</b> This field indicates the Asymmetric Namespace Access state for all namespaces in this ANA Group when accessed through this controller.</td> </tr> <tr> <td> <table border="1"> <thead> <tr> <th>Value</th> <th>Asymmetric Namespace Access State</th> <th>Reference</th> </tr> </thead> <tbody> <tr> <td>01h</td> <td>ANA Optimized state</td> <td>8.1.1.4</td> </tr> <tr> <td>02h</td> <td>ANA Non-Optimized state</td> <td>8.1.1.5</td> </tr> <tr> <td>03h</td> <td>ANA Inaccessible state</td> <td>8.1.1.6</td> </tr> <tr> <td>04h</td> <td>ANA Persistent Loss state</td> <td>8.1.1.7</td> </tr> <tr> <td>0Fh</td> <td>ANA Change state</td> <td>8.1.1.8</td> </tr> <tr> <td>All other values</td> <td colspan="2">Reserved</td> </tr> </tbody> </table> </td> </tr> </tbody> </table>	Bits	Description	07:04	Reserved	03:00	<b>Asymmetric Namespace Access State (ANAS):</b> This field indicates the Asymmetric Namespace Access state for all namespaces in this ANA Group when accessed through this controller.	<table border="1"> <thead> <tr> <th>Value</th> <th>Asymmetric Namespace Access State</th> <th>Reference</th> </tr> </thead> <tbody> <tr> <td>01h</td> <td>ANA Optimized state</td> <td>8.1.1.4</td> </tr> <tr> <td>02h</td> <td>ANA Non-Optimized state</td> <td>8.1.1.5</td> </tr> <tr> <td>03h</td> <td>ANA Inaccessible state</td> <td>8.1.1.6</td> </tr> <tr> <td>04h</td> <td>ANA Persistent Loss state</td> <td>8.1.1.7</td> </tr> <tr> <td>0Fh</td> <td>ANA Change state</td> <td>8.1.1.8</td> </tr> <tr> <td>All other values</td> <td colspan="2">Reserved</td> </tr> </tbody> </table>	Value	Asymmetric Namespace Access State	Reference	01h	ANA Optimized state	8.1.1.4	02h	ANA Non-Optimized state	8.1.1.5	03h	ANA Inaccessible state	8.1.1.6	04h	ANA Persistent Loss state	8.1.1.7	0Fh	ANA Change state	8.1.1.8	All other values	Reserved	
	Bits	Description																											
	07:04	Reserved																											
03:00	<b>Asymmetric Namespace Access State (ANAS):</b> This field indicates the Asymmetric Namespace Access state for all namespaces in this ANA Group when accessed through this controller.																												
	<table border="1"> <thead> <tr> <th>Value</th> <th>Asymmetric Namespace Access State</th> <th>Reference</th> </tr> </thead> <tbody> <tr> <td>01h</td> <td>ANA Optimized state</td> <td>8.1.1.4</td> </tr> <tr> <td>02h</td> <td>ANA Non-Optimized state</td> <td>8.1.1.5</td> </tr> <tr> <td>03h</td> <td>ANA Inaccessible state</td> <td>8.1.1.6</td> </tr> <tr> <td>04h</td> <td>ANA Persistent Loss state</td> <td>8.1.1.7</td> </tr> <tr> <td>0Fh</td> <td>ANA Change state</td> <td>8.1.1.8</td> </tr> <tr> <td>All other values</td> <td colspan="2">Reserved</td> </tr> </tbody> </table>	Value	Asymmetric Namespace Access State	Reference	01h		ANA Optimized state	8.1.1.4	02h	ANA Non-Optimized state	8.1.1.5	03h	ANA Inaccessible state	8.1.1.6	04h	ANA Persistent Loss state	8.1.1.7	0Fh	ANA Change state	8.1.1.8	All other values	Reserved							
	Value	Asymmetric Namespace Access State	Reference																										
	01h	ANA Optimized state	8.1.1.4																										
	02h	ANA Non-Optimized state	8.1.1.5																										
	03h	ANA Inaccessible state	8.1.1.6																										
04h	ANA Persistent Loss state	8.1.1.7																											
0Fh	ANA Change state	8.1.1.8																											
All other values	Reserved																												
31:17	Reserved																												
<b>Namespace Identifier List</b>																													
35:32	<b>Namespace Identifier 0:</b> The Namespace Identifier of the first namespace that is a member of this ANA Group.																												
39:36	<b>Namespace Identifier 1:</b> The Namespace Identifier of the second namespace, if any, that is a member of this ANA Group.																												
...	...																												
(((NNV-1)*4) + 35): (((NNV-1)*4) + 32)	<b>Namespace Identifier NNV-1:</b> The Namespace Identifier of the last namespace that is a member of this ANA Group.																												

**5.1.12.1.14 Persistent Event (Log Page Identifier 0Dh)**

The Persistent Event log page contains information about significant events not specific to a particular command. The information in this log page shall be retained across power cycles and resets. NVM subsystems should be designed for minimal loss of event information upon power failure. This log page consists of a header describing the log and zero or more Persistent Events (refer to section 5.1.12.1.14.1).

This log page is global to the NVM subsystem.

The Log Specific Parameter field in Command Dword 10 (refer to Figure 197) for this log page is defined in Figure 225.



A sanitize operation may alter this log page (e.g., remove or modify events to prevent derivation of user data from log page information, refer to section 8.1.24). The events removed from this log page by a sanitize operation are unspecified.

Persistent Event Log events specified in this section should be reported in an order such that more recent events are generally reported earlier in the log data than older events. The method by which the NVM subsystem determines the order in which events occurred is vendor specific.

The number of events supported is vendor specific. The supported maximum size for the Persistent Event Log is indicated in the PELS field of the Identify Controller data structure (refer to Figure 312). The number of events supported and the supported maximum size should be large enough that the number of events or the size of the Persistent Event Log data does not reach the maximum supported size over the usable life of the NVM subsystem.

The controller shall log all supported events at each event occurrence unless the controller determines that the same event is occurring at a frequency that exceeds a vendor specific threshold for the frequency of event creation. If the same event is occurring at a frequency that exceeds a vendor specific threshold then the vendor may suppress further entries for the same event. A controller may indicate if events have been suppressed in vendor specific event data.

It is vendor specific which events are deleted (e.g., important events may be retained and events that are newer than an important event that was retained may be deleted) to make room for future events if:

- a) the size of the Persistent Event Log data reaches the maximum supported size;
- b) the number of events reaches the largest reportable number of events; or
- c) an event category reaches the largest reportable number of events for that category (e.g., information regarding 1,000 occurrences of changes to the timestamp is stored in internal data structures and extracted for reporting as Timestamp Change events in the Persistent Event Log and more than 1,000 Timestamp Change events have occurred).

Events that affect multiple controllers (e.g., an NVM Subsystem Reset) should be logged once by a controller selected by the vendor and not logged by any other controllers.

The Action field in the Log Specific Parameter field (refer to Figure 225) specifies whether:

- a) A persistent event log reporting context is created at the start of processing this Get Log Page command and log page data, if any, is read from the log page data associated with that log reporting context;
- b) Log page data is read from the log page data associated with a preexisting log reporting context; or
- c) The persistent event log reporting context, if any, is released.

The persistent event log reporting context is vendor specific information that the controller creates for determining what information is included in the persistent event log page data (e.g., the persistent event log reporting context may be the persistent event log page data or may contain a set of pointers to the events to report).

The controller should retain the persistent event log reporting context:

- a) Until the controller processes:
  - a) a Get Log Page command requesting the Persistent Event log page with the Action field set to 02h (i.e., Release Context);
  - b) an NVM Subsystem Reset; or
  - c) a Controller Level Reset;or
- b) For a vendor specific time long enough to allow retrieval of the Persistent Event log page data.

Persistent Event Log events that occur while a persistent event log reporting context exists shall not be reported in the existing reporting context but shall be logged.

The host is expected to issue a Get Log Page command with the Action field set to 02h to release the persistent event log reporting context after reading the persistent event log page data.

If the Persistent Event Log is not read with a single Get Log Page command, then host software should read the Generation Number field in the Persistent Event Log header after establishing a reporting context but before reading the remainder of the log and then re-read the Generation Number field after it has read the entire log. If the generation numbers do not match, then:

- the reporting context may have been lost while reading the log;
- the Persistent Event Log contents read may be invalid; and
- host software should re-read the log.

**Figure 225: Persistent Event Log Specific Parameter Field**

Bits	Description										
14:10	Reserved										
09:08	<p><b>Action (ACT):</b> This field specifies the action the controller shall take during processing this Get Log Page command.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td> <p><b>Read Log Data:</b> Return Persistent Event log page data starting at the address indicated by the LPOU field and the LPOL field in the Get Log Page command. If the controller does not have a persistent event log reporting context, then the controller shall abort the command with a status code of Command Sequence Error.</p> </td> </tr> <tr> <td>01b</td> <td> <p><b>Establish Context and Read Log Data:</b> The controller shall:</p> <ol style="list-style-type: none"> <li>determine the length of the Persistent Event log page data;</li> <li>determine the set of events to report in the Persistent Event log page data; and</li> <li>establish a persistent event log reporting context to store information describing the Persistent Event log page data to be reported and track Persistent Event log page data accesses.</li> </ol> <p>After establishing a persistent event log reporting context, the controller shall return Persistent Event log page data starting at the address indicated by the LPOU field and the LPOL field in the Get Log Page command.</p> <p>If a persistent event log reporting context already exists, then the controller shall abort the command with a status code of Command Sequence Error.</p> </td> </tr> <tr> <td>10b</td> <td> <p><b>Release Context:</b> The controller shall:</p> <ul style="list-style-type: none"> <li>release the persistent event log reporting context, if any. It is not an error if the controller does not have a persistent event log reporting context; and</li> <li>not return any Persistent Event log page data (i.e., the controller ignores the NUMDU field, NUMDL field, LPOL field, and the LPOU field).</li> </ul> </td> </tr> <tr> <td>11b</td> <td> <p><b>Establish Context and Read 512 Bytes of Header:</b> The controller shall:</p> <ol style="list-style-type: none"> <li>determine the length of the Persistent Event log page data;</li> <li>determine the set of events to report in the Persistent Event log page data; and</li> <li>if a reporting context does not already exist, then establish a persistent event log reporting context to store information describing the Persistent Event log page data to be reported and track Persistent Event log page data accesses.</li> </ol> <p>If a persistent event log reporting context did not already exist when the Get Log Page command was processed, then the controller shall:</p> <ol style="list-style-type: none"> <li>establish a persistent event log reporting context; and</li> <li>after establishing the context, return 512 bytes of the Persistent Event Log Header starting at offset 0h with the Reporting Context Exists bit cleared to '0' with a status code of Successful Completion.</li> </ol> <p>If a persistent event log reporting context already existed when the Get Log Page command was processed, then the controller shall return 512 bytes of the Persistent Event Log Header starting at offset 0h with the Reporting Context Exists bit set to '1' with a status code of Successful Completion.</p> <p>The 512 bytes of the Persistent Event Log Header shall be returned regardless of the values in the LPOL, LPOU, NUMDL, and NUMDU fields (i.e., the controller shall ignore the LPOL, LPOU, NUMDL, and NUMDU fields). Hosts should ensure allocation of at least 512 bytes in the data buffer to avoid possible data corruption.</p> </td> </tr> </tbody> </table>	Value	Definition	00b	<p><b>Read Log Data:</b> Return Persistent Event log page data starting at the address indicated by the LPOU field and the LPOL field in the Get Log Page command. If the controller does not have a persistent event log reporting context, then the controller shall abort the command with a status code of Command Sequence Error.</p>	01b	<p><b>Establish Context and Read Log Data:</b> The controller shall:</p> <ol style="list-style-type: none"> <li>determine the length of the Persistent Event log page data;</li> <li>determine the set of events to report in the Persistent Event log page data; and</li> <li>establish a persistent event log reporting context to store information describing the Persistent Event log page data to be reported and track Persistent Event log page data accesses.</li> </ol> <p>After establishing a persistent event log reporting context, the controller shall return Persistent Event log page data starting at the address indicated by the LPOU field and the LPOL field in the Get Log Page command.</p> <p>If a persistent event log reporting context already exists, then the controller shall abort the command with a status code of Command Sequence Error.</p>	10b	<p><b>Release Context:</b> The controller shall:</p> <ul style="list-style-type: none"> <li>release the persistent event log reporting context, if any. It is not an error if the controller does not have a persistent event log reporting context; and</li> <li>not return any Persistent Event log page data (i.e., the controller ignores the NUMDU field, NUMDL field, LPOL field, and the LPOU field).</li> </ul>	11b	<p><b>Establish Context and Read 512 Bytes of Header:</b> The controller shall:</p> <ol style="list-style-type: none"> <li>determine the length of the Persistent Event log page data;</li> <li>determine the set of events to report in the Persistent Event log page data; and</li> <li>if a reporting context does not already exist, then establish a persistent event log reporting context to store information describing the Persistent Event log page data to be reported and track Persistent Event log page data accesses.</li> </ol> <p>If a persistent event log reporting context did not already exist when the Get Log Page command was processed, then the controller shall:</p> <ol style="list-style-type: none"> <li>establish a persistent event log reporting context; and</li> <li>after establishing the context, return 512 bytes of the Persistent Event Log Header starting at offset 0h with the Reporting Context Exists bit cleared to '0' with a status code of Successful Completion.</li> </ol> <p>If a persistent event log reporting context already existed when the Get Log Page command was processed, then the controller shall return 512 bytes of the Persistent Event Log Header starting at offset 0h with the Reporting Context Exists bit set to '1' with a status code of Successful Completion.</p> <p>The 512 bytes of the Persistent Event Log Header shall be returned regardless of the values in the LPOL, LPOU, NUMDL, and NUMDU fields (i.e., the controller shall ignore the LPOL, LPOU, NUMDL, and NUMDU fields). Hosts should ensure allocation of at least 512 bytes in the data buffer to avoid possible data corruption.</p>
	Value	Definition									
	00b	<p><b>Read Log Data:</b> Return Persistent Event log page data starting at the address indicated by the LPOU field and the LPOL field in the Get Log Page command. If the controller does not have a persistent event log reporting context, then the controller shall abort the command with a status code of Command Sequence Error.</p>									
	01b	<p><b>Establish Context and Read Log Data:</b> The controller shall:</p> <ol style="list-style-type: none"> <li>determine the length of the Persistent Event log page data;</li> <li>determine the set of events to report in the Persistent Event log page data; and</li> <li>establish a persistent event log reporting context to store information describing the Persistent Event log page data to be reported and track Persistent Event log page data accesses.</li> </ol> <p>After establishing a persistent event log reporting context, the controller shall return Persistent Event log page data starting at the address indicated by the LPOU field and the LPOL field in the Get Log Page command.</p> <p>If a persistent event log reporting context already exists, then the controller shall abort the command with a status code of Command Sequence Error.</p>									
10b	<p><b>Release Context:</b> The controller shall:</p> <ul style="list-style-type: none"> <li>release the persistent event log reporting context, if any. It is not an error if the controller does not have a persistent event log reporting context; and</li> <li>not return any Persistent Event log page data (i.e., the controller ignores the NUMDU field, NUMDL field, LPOL field, and the LPOU field).</li> </ul>										
11b	<p><b>Establish Context and Read 512 Bytes of Header:</b> The controller shall:</p> <ol style="list-style-type: none"> <li>determine the length of the Persistent Event log page data;</li> <li>determine the set of events to report in the Persistent Event log page data; and</li> <li>if a reporting context does not already exist, then establish a persistent event log reporting context to store information describing the Persistent Event log page data to be reported and track Persistent Event log page data accesses.</li> </ol> <p>If a persistent event log reporting context did not already exist when the Get Log Page command was processed, then the controller shall:</p> <ol style="list-style-type: none"> <li>establish a persistent event log reporting context; and</li> <li>after establishing the context, return 512 bytes of the Persistent Event Log Header starting at offset 0h with the Reporting Context Exists bit cleared to '0' with a status code of Successful Completion.</li> </ol> <p>If a persistent event log reporting context already existed when the Get Log Page command was processed, then the controller shall return 512 bytes of the Persistent Event Log Header starting at offset 0h with the Reporting Context Exists bit set to '1' with a status code of Successful Completion.</p> <p>The 512 bytes of the Persistent Event Log Header shall be returned regardless of the values in the LPOL, LPOU, NUMDL, and NUMDU fields (i.e., the controller shall ignore the LPOL, LPOU, NUMDL, and NUMDU fields). Hosts should ensure allocation of at least 512 bytes in the data buffer to avoid possible data corruption.</p>										

The log page returned is defined in Figure 226.

Figure 226: Persistent Event Log Page

Bytes	Description
<b>Persistent Event Log Header</b>	
00	<b>Log Page Identifier (LID):</b> This field shall be set to 0Dh.
03:01	Reserved
07:04	<b>Total Number of Events (TNEV):</b> Contains the number of event entries in this log page.
15:08	<b>Total Log Length (TLL):</b> Contains the total number of bytes of persistent event log page data available, including the header.
16	<b>Log Revision (LREV):</b> Contains a number indicating the revision of the Get Log Page data structure that this log page data complies with. Shall be set to 03h. This revision applies to the Persistent Event Log and the Persistent Event Format (refer to Figure 227). This revision does not apply to the contents of the Event Data field in the Persistent Event Format as that field is covered by the Event Type Revision (refer to Figure 227).
17	Reserved
19:18	<b>Log Header Length (LHL):</b> This field contains the length in bytes of the log header information that follows. The total length of the log header in bytes is the value in this field plus 20.
27:20	<b>Timestamp (TSTMP):</b> Shall contain a timestamp of the time at which the persistent event log reporting context was established. The value returned shall use the Timestamp data structure defined in Figure 396.
43:28	<b>Power on Hours (POH):</b> This field indicates the number of power-on hours at the time the Persistent Event log was retrieved. This may not include time that the controller was powered and in a non-operational state.
51:44	<b>Power Cycle Count (PWRCC):</b> Contains the number of power cycles for this controller.
53:52	<b>PCI Vendor ID (VID):</b> This is the same value as reported in the Identify Controller data structure PCI Vendor ID field (i.e., bytes 01:00).
55:54	<b>PCI Subsystem Vendor ID (SSVID):</b> This is the same value as reported in the Identify Controller data structure PCI Subsystem Vendor ID field (i.e., bytes 03:02).
75:56	<b>Serial Number (SN):</b> This field contains the same value as reported in the Serial Number field of the Identify Controller data structure, bytes 23:04.
115:76	<b>Model Number (MN):</b> This field contains the same value as reported in the Model Number field of the Identify Controller data structure, bytes 63:24.
371:116	<b>NVM Subsystem NVMe Qualified Name (SUBNQN):</b> This field contains the same value as reported in the NVM Subsystem NVMe Qualified Name field of the Identify Controller data structure, bytes 1023:768. If the NVM Subsystem NVMe Qualified Name field of the Identify Controller data structure is not supported, then all bytes of this field shall be cleared to 0h.
373:372	<b>Generation Number (GNUM):</b> Contains a value that is incremented each time a persistent event log reporting context is established and the log page returns different data than when this log page last established a reporting context. If the value of this field is FFFFh, then the field shall be cleared to 0h when incremented (i.e., rolls over to 0h).

**Figure 226: Persistent Event Log Page**

Bytes	Description																				
377:374	<p><b>Reporting Context Information (RCI):</b> This field contains information about the persistent event log reporting context.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>31:19</td> <td>Reserved</td> </tr> <tr> <td>18</td> <td> <p><b>Reporting Context Exists (RCE):</b> This bit indicates the persistent event log reporting context. If this bit is set to '1', then a persistent event log reporting context already existed when the Get Log Page command that requested this log page was processed. If this bit is cleared to '0', then a persistent event log reporting context did not already exist when the Get Log Page command that requested this log page was processed.</p> </td> </tr> <tr> <td>17:16</td> <td> <p><b>Reporting Context Port Identifier Type (RCPIT):</b> If the RCE bit is set to '1', then this field indicates the type of port identifier reported in the Reporting Context Port Identifier (RCPID) field. If the RCE bit is cleared to '0', then this field shall be cleared to 00b.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>A persistent event log reporting context does not already exist.</td> </tr> <tr> <td>01b</td> <td>The reporting context was established by an NVM subsystem port.</td> </tr> <tr> <td>10b</td> <td>The reporting context was established by a Management Endpoint (refer to the NVM Express Management Interface Specification).</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table> </td> </tr> <tr> <td>15:00</td> <td> <p><b>Reporting Context Port Identifier (RCPID):</b> If the RCE bit is cleared to '0', then this field shall be cleared to 0h.</p> <p>If the RCE bit is set to '1', then this field contains a Port Identifier of the type indicated in the RCPIT field.</p> <p>If the RCPIT field is set to 01b, then this field shall contain the Port Identifier of the NVM subsystem port that established the reporting context as defined in the Primary Controller Capabilities data structure (refer to section 5.1.13.3.1).</p> <p>If the RCPIT field is set to 10b, then:</p> <ul style="list-style-type: none"> <li>the least-significant byte of this field shall contain the Port Identifier of the Management Endpoint (refer to the NVM Express Management Interface Specification); and</li> <li>the most-significant byte of this field shall be cleared to 0h.</li> </ul> </td> </tr> </tbody> </table>	Bits	Description	31:19	Reserved	18	<p><b>Reporting Context Exists (RCE):</b> This bit indicates the persistent event log reporting context. If this bit is set to '1', then a persistent event log reporting context already existed when the Get Log Page command that requested this log page was processed. If this bit is cleared to '0', then a persistent event log reporting context did not already exist when the Get Log Page command that requested this log page was processed.</p>	17:16	<p><b>Reporting Context Port Identifier Type (RCPIT):</b> If the RCE bit is set to '1', then this field indicates the type of port identifier reported in the Reporting Context Port Identifier (RCPID) field. If the RCE bit is cleared to '0', then this field shall be cleared to 00b.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>A persistent event log reporting context does not already exist.</td> </tr> <tr> <td>01b</td> <td>The reporting context was established by an NVM subsystem port.</td> </tr> <tr> <td>10b</td> <td>The reporting context was established by a Management Endpoint (refer to the NVM Express Management Interface Specification).</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	A persistent event log reporting context does not already exist.	01b	The reporting context was established by an NVM subsystem port.	10b	The reporting context was established by a Management Endpoint (refer to the NVM Express Management Interface Specification).	11b	Reserved	15:00	<p><b>Reporting Context Port Identifier (RCPID):</b> If the RCE bit is cleared to '0', then this field shall be cleared to 0h.</p> <p>If the RCE bit is set to '1', then this field contains a Port Identifier of the type indicated in the RCPIT field.</p> <p>If the RCPIT field is set to 01b, then this field shall contain the Port Identifier of the NVM subsystem port that established the reporting context as defined in the Primary Controller Capabilities data structure (refer to section 5.1.13.3.1).</p> <p>If the RCPIT field is set to 10b, then:</p> <ul style="list-style-type: none"> <li>the least-significant byte of this field shall contain the Port Identifier of the Management Endpoint (refer to the NVM Express Management Interface Specification); and</li> <li>the most-significant byte of this field shall be cleared to 0h.</li> </ul>
	Bits	Description																			
	31:19	Reserved																			
	18	<p><b>Reporting Context Exists (RCE):</b> This bit indicates the persistent event log reporting context. If this bit is set to '1', then a persistent event log reporting context already existed when the Get Log Page command that requested this log page was processed. If this bit is cleared to '0', then a persistent event log reporting context did not already exist when the Get Log Page command that requested this log page was processed.</p>																			
17:16	<p><b>Reporting Context Port Identifier Type (RCPIT):</b> If the RCE bit is set to '1', then this field indicates the type of port identifier reported in the Reporting Context Port Identifier (RCPID) field. If the RCE bit is cleared to '0', then this field shall be cleared to 00b.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>A persistent event log reporting context does not already exist.</td> </tr> <tr> <td>01b</td> <td>The reporting context was established by an NVM subsystem port.</td> </tr> <tr> <td>10b</td> <td>The reporting context was established by a Management Endpoint (refer to the NVM Express Management Interface Specification).</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	A persistent event log reporting context does not already exist.	01b	The reporting context was established by an NVM subsystem port.	10b	The reporting context was established by a Management Endpoint (refer to the NVM Express Management Interface Specification).	11b	Reserved										
Value	Definition																				
00b	A persistent event log reporting context does not already exist.																				
01b	The reporting context was established by an NVM subsystem port.																				
10b	The reporting context was established by a Management Endpoint (refer to the NVM Express Management Interface Specification).																				
11b	Reserved																				
15:00	<p><b>Reporting Context Port Identifier (RCPID):</b> If the RCE bit is cleared to '0', then this field shall be cleared to 0h.</p> <p>If the RCE bit is set to '1', then this field contains a Port Identifier of the type indicated in the RCPIT field.</p> <p>If the RCPIT field is set to 01b, then this field shall contain the Port Identifier of the NVM subsystem port that established the reporting context as defined in the Primary Controller Capabilities data structure (refer to section 5.1.13.3.1).</p> <p>If the RCPIT field is set to 10b, then:</p> <ul style="list-style-type: none"> <li>the least-significant byte of this field shall contain the Port Identifier of the Management Endpoint (refer to the NVM Express Management Interface Specification); and</li> <li>the most-significant byte of this field shall be cleared to 0h.</li> </ul>																				
479:378	Reserved																				

**Figure 226: Persistent Event Log Page**

Bytes	Description																																																												
511:480	<p><b>Supported Events Bitmap (SEB):</b> This field contains a bitmap indicating support for the persistent event log events. Each bit in the bitmap corresponds to the value for the event type (refer to Figure 229) (e.g., bit 222 decimal, DEh, corresponds to event type value DEh, Vendor Specific Event). A bit set to '1' indicates that the corresponding event is supported. A bit cleared to '0' indicates that the corresponding event is not supported.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> <th>Reference</th> </tr> </thead> <tbody> <tr> <td>255:224</td> <td>Reserved</td> <td></td> </tr> <tr> <td>223</td> <td><b>TCG Defined</b></td> <td>TCG Storage Interface Interactions Specification</td> </tr> <tr> <td>222</td> <td><b>Vendor Specific Event Supported (VSES)</b></td> <td>5.1.12.1.14.2.15</td> </tr> <tr> <td>221:15</td> <td>Reserved</td> <td></td> </tr> <tr> <td>14</td> <td><b>Sanitize Media Verification Event Support (SMVES)</b></td> <td>5.1.12.1.14.2.14</td> </tr> <tr> <td>13</td> <td><b>Thermal Excursion Event Support (TEES)</b></td> <td>5.1.12.1.14.2.13</td> </tr> <tr> <td>12</td> <td><b>Telemetry Log Create Event Support (TLCES)</b></td> <td>5.1.12.1.14.2.12</td> </tr> <tr> <td>11</td> <td><b>Set Feature Event Support (SFES)</b></td> <td>5.1.12.1.14.2.11</td> </tr> <tr> <td>10</td> <td><b>Sanitize Completion Event Support (SCES)</b></td> <td>5.1.12.1.14.2.10</td> </tr> <tr> <td>09</td> <td><b>Sanitize Start Event Support (SSES)</b></td> <td>5.1.12.1.14.2.9</td> </tr> <tr> <td>08</td> <td><b>Format NVM Completion Event Support (FNCES)</b></td> <td>5.1.12.1.14.2.8</td> </tr> <tr> <td>07</td> <td><b>Format NVM Start Event Support (FNSES)</b></td> <td>5.1.12.1.14.2.7</td> </tr> <tr> <td>06</td> <td><b>Change Namespace Event Support (CNES)</b></td> <td>5.1.12.1.14.2.6</td> </tr> <tr> <td>05</td> <td><b>NVM Subsystem Hardware Error Event Support (NSHEES)</b></td> <td>5.1.12.1.14.2.5</td> </tr> <tr> <td>04</td> <td><b>Power-on or Reset Event Supported (PRES)</b></td> <td>5.1.12.1.14.2.4</td> </tr> <tr> <td>03</td> <td><b>Timestamp Change Event Supported (TCES)</b></td> <td>5.1.12.1.14.2.3</td> </tr> <tr> <td>02</td> <td><b>Firmware Commit Event Supported (FCES)</b></td> <td>5.1.12.1.14.2.2</td> </tr> <tr> <td>01</td> <td><b>SMART / Health Log Snapshot Event Supported (SHLSES)</b></td> <td>5.1.12.1.14.2.1</td> </tr> <tr> <td>00</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Bits	Description	Reference	255:224	Reserved		223	<b>TCG Defined</b>	TCG Storage Interface Interactions Specification	222	<b>Vendor Specific Event Supported (VSES)</b>	5.1.12.1.14.2.15	221:15	Reserved		14	<b>Sanitize Media Verification Event Support (SMVES)</b>	5.1.12.1.14.2.14	13	<b>Thermal Excursion Event Support (TEES)</b>	5.1.12.1.14.2.13	12	<b>Telemetry Log Create Event Support (TLCES)</b>	5.1.12.1.14.2.12	11	<b>Set Feature Event Support (SFES)</b>	5.1.12.1.14.2.11	10	<b>Sanitize Completion Event Support (SCES)</b>	5.1.12.1.14.2.10	09	<b>Sanitize Start Event Support (SSES)</b>	5.1.12.1.14.2.9	08	<b>Format NVM Completion Event Support (FNCES)</b>	5.1.12.1.14.2.8	07	<b>Format NVM Start Event Support (FNSES)</b>	5.1.12.1.14.2.7	06	<b>Change Namespace Event Support (CNES)</b>	5.1.12.1.14.2.6	05	<b>NVM Subsystem Hardware Error Event Support (NSHEES)</b>	5.1.12.1.14.2.5	04	<b>Power-on or Reset Event Supported (PRES)</b>	5.1.12.1.14.2.4	03	<b>Timestamp Change Event Supported (TCES)</b>	5.1.12.1.14.2.3	02	<b>Firmware Commit Event Supported (FCES)</b>	5.1.12.1.14.2.2	01	<b>SMART / Health Log Snapshot Event Supported (SHLSES)</b>	5.1.12.1.14.2.1	00	Reserved	
	Bits	Description	Reference																																																										
	255:224	Reserved																																																											
	223	<b>TCG Defined</b>	TCG Storage Interface Interactions Specification																																																										
	222	<b>Vendor Specific Event Supported (VSES)</b>	5.1.12.1.14.2.15																																																										
	221:15	Reserved																																																											
	14	<b>Sanitize Media Verification Event Support (SMVES)</b>	5.1.12.1.14.2.14																																																										
	13	<b>Thermal Excursion Event Support (TEES)</b>	5.1.12.1.14.2.13																																																										
	12	<b>Telemetry Log Create Event Support (TLCES)</b>	5.1.12.1.14.2.12																																																										
	11	<b>Set Feature Event Support (SFES)</b>	5.1.12.1.14.2.11																																																										
	10	<b>Sanitize Completion Event Support (SCES)</b>	5.1.12.1.14.2.10																																																										
	09	<b>Sanitize Start Event Support (SSES)</b>	5.1.12.1.14.2.9																																																										
	08	<b>Format NVM Completion Event Support (FNCES)</b>	5.1.12.1.14.2.8																																																										
	07	<b>Format NVM Start Event Support (FNSES)</b>	5.1.12.1.14.2.7																																																										
	06	<b>Change Namespace Event Support (CNES)</b>	5.1.12.1.14.2.6																																																										
	05	<b>NVM Subsystem Hardware Error Event Support (NSHEES)</b>	5.1.12.1.14.2.5																																																										
	04	<b>Power-on or Reset Event Supported (PRES)</b>	5.1.12.1.14.2.4																																																										
	03	<b>Timestamp Change Event Supported (TCES)</b>	5.1.12.1.14.2.3																																																										
02	<b>Firmware Commit Event Supported (FCES)</b>	5.1.12.1.14.2.2																																																											
01	<b>SMART / Health Log Snapshot Event Supported (SHLSES)</b>	5.1.12.1.14.2.1																																																											
00	Reserved																																																												
<b>Persistent Event Log Events</b>																																																													
(M-1)+512:512	<b>Persistent Event 0:</b> This field contains the first persistent event log entry (refer to Figure 227) where M is the length of this persistent event.																																																												
...	...																																																												
(TLL-1):(TLL-K)	<b>Persistent Event N:</b> This field contains the last persistent event log entry (refer to Figure 227) where K is the length of this persistent event, TLL is the size specified in the Total Log Length field, and N is the value of the TNEV field minus 1h.																																																												

The format of the Persistent Events in the Persistent Event log is shown in Figure 227.

**Figure 227: Persistent Event Format**

Bytes	Description
<b>Persistent Event Log Event Header</b>	
00	<b>Event Type (ET):</b> This field indicates the event type for this entry. Refer to section 5.1.12.1.14.1 for the definition of the event types.
01	<b>Event Type Revision (ETR):</b> This field contains a number indicating the revision of the event data structure for the event indicated by the Event Type field that this event data complies with.
02	<b>Event Header Length (EHL):</b> This field contains the length in bytes of the event header information that follows. The total length of the event header in bytes is the value in this field plus 3. The host should use the value in this field to calculate the offset to the start of the Vendor Specific Information field.

**Figure 227: Persistent Event Format**

Bytes	Description																	
03	<b>Event Header Additional Information (EHAI):</b> This field indicates if additional information is present in this event header.																	
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:2</td> <td>Reserved</td> </tr> <tr> <td rowspan="5">1:0</td> <td><b>Port Identifier Type (PIT):</b> This field indicates the type of port identifier reported in the Port Identifier (PELPID) field. Implementations that are compliant with NVM Express Base Specification Revision 2.0 and later shall set this field to a non-zero value.</td> </tr> <tr> <td> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>The Port Identifier Type is not reported and the Port Identifier (PELPID) field does not apply.</td> </tr> <tr> <td>01b</td> <td>This event is associated with an NVM subsystem port.</td> </tr> <tr> <td>10b</td> <td>This event is associated with a Management Endpoint (refer to the NVM Express Management Interface Specification).</td> </tr> <tr> <td>11b</td> <td>This event is not associated with any port.</td> </tr> </tbody> </table> </td> </tr> </tbody> </table>	Bits	Description	7:2	Reserved	1:0	<b>Port Identifier Type (PIT):</b> This field indicates the type of port identifier reported in the Port Identifier (PELPID) field. Implementations that are compliant with NVM Express Base Specification Revision 2.0 and later shall set this field to a non-zero value.	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>The Port Identifier Type is not reported and the Port Identifier (PELPID) field does not apply.</td> </tr> <tr> <td>01b</td> <td>This event is associated with an NVM subsystem port.</td> </tr> <tr> <td>10b</td> <td>This event is associated with a Management Endpoint (refer to the NVM Express Management Interface Specification).</td> </tr> <tr> <td>11b</td> <td>This event is not associated with any port.</td> </tr> </tbody> </table>	Value	Definition	00b	The Port Identifier Type is not reported and the Port Identifier (PELPID) field does not apply.	01b	This event is associated with an NVM subsystem port.	10b	This event is associated with a Management Endpoint (refer to the NVM Express Management Interface Specification).	11b	This event is not associated with any port.
	Bits	Description																
	7:2	Reserved																
1:0	<b>Port Identifier Type (PIT):</b> This field indicates the type of port identifier reported in the Port Identifier (PELPID) field. Implementations that are compliant with NVM Express Base Specification Revision 2.0 and later shall set this field to a non-zero value.																	
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>The Port Identifier Type is not reported and the Port Identifier (PELPID) field does not apply.</td> </tr> <tr> <td>01b</td> <td>This event is associated with an NVM subsystem port.</td> </tr> <tr> <td>10b</td> <td>This event is associated with a Management Endpoint (refer to the NVM Express Management Interface Specification).</td> </tr> <tr> <td>11b</td> <td>This event is not associated with any port.</td> </tr> </tbody> </table>	Value	Definition	00b	The Port Identifier Type is not reported and the Port Identifier (PELPID) field does not apply.		01b	This event is associated with an NVM subsystem port.	10b	This event is associated with a Management Endpoint (refer to the NVM Express Management Interface Specification).	11b	This event is not associated with any port.						
	Value	Definition																
	00b	The Port Identifier Type is not reported and the Port Identifier (PELPID) field does not apply.																
	01b	This event is associated with an NVM subsystem port.																
10b	This event is associated with a Management Endpoint (refer to the NVM Express Management Interface Specification).																	
11b	This event is not associated with any port.																	
05:04	<b>Controller Identifier (CNTLID):</b> This field contains the NVM subsystem unique controller identifier for the controller that created this event. If the event is controller specific, then the event data is associated with that controller. If the event is not controller specific, then this is the controller that the NVM subsystem selected for creating the event.																	
13:06	<b>Event Timestamp (ETSTP):</b> This field contains a timestamp of the time when this event occurred using the Timestamp data structure defined in Figure 395.																	
15:14	<p><b>Port Identifier (PELPID):</b> If the PIT field in the EHAI field is cleared to 00b or set to 11b, then this field shall be cleared to 0h.</p> <p>If the PIT field in the EHAI field is not cleared to 00b or set to 11b, then this field contains a Port Identifier of the type indicated in the PIT field.</p> <p>If the PIT field in the EHAI field is set to 01b, then this field shall contain the Port Identifier of the NVM subsystem port associated with this event as defined in the Primary Controller Capabilities data structure (refer to section 5.1.13.3.1).</p> <p>If the PIT field in the EHAI field is set to 10b, then:</p> <ul style="list-style-type: none"> <li>the least-significant byte of this field shall contain the Port Identifier of the Management Endpoint associated with this event (refer to the NVM Express Management Interface Specification); and</li> <li>the most-significant byte of this field shall be cleared to 0h.</li> </ul>																	
19:16	Reserved																	
21:20	<b>Vendor Specific Information Length (VSIL):</b> This field indicates the length in bytes of the Vendor Specific Information. If no Vendor Specific Information is present, then this field shall be cleared to 0h. The length of the Vendor Specific Information is included in the Event Length field (bytes 23:22). Information associated with this event that is not able to be described in the event data structure fields may be reported in Vendor Specific Information fields in this event.																	
23:22	<b>Event Length (EL):</b> This field indicates the length in bytes of the vendor specific information, if any, and the persistent event log event data that follows. The total length of the event in bytes is the value in this field plus the value in the Event Header Length field plus 3.																	
<b>Vendor Specific Information, if any</b>																		
EHL+2+VSIL:EHL+3	<b>Vendor Specific Information (VSI):</b> This field contains the vendor specific information, if any. This field is omitted if there is no vendor specific information (i.e., if VSIL is cleared to 0h).																	
<b>Persistent Event Log Event Data</b>																		
EHL+EL+2: EHL+3+VSIL	<b>Event Data (ED):</b> This field contains persistent event log event data, if any (refer to section 5.1.12.1.14.1).																	

### 5.1.12.1.14.1 Persistent Event Log Page LID Specific Parameter Field

Figure 228 specifies the format for the LID Specific Parameter field in the Supported Log Pages log page (refer to section 5.1.12.1.1) for the Persistent Event log page.

**Figure 228: Persistent Event LID Specific Parameter Field**

Bits	Description
15:1	Reserved
0	<p><b>Establish Context and Read 512 Bytes of Header Supported (ECRH):</b> If this bit is cleared to '0', then the controller does not support the Establish Context and Read 512 Bytes of Header action (refer to Figure 225).</p> <p>If this bit is set to '1', then the controller supports:</p> <ul style="list-style-type: none"> <li>the Establish Context and Read 512 Bytes of Header action; and</li> <li>the Generation Number field in the Persistent Event log page.</li> </ul> <p>If the Persistent Event log page is supported by the controller, then implementations compliant with NVM Express Base Specification, Revision 2.0 and later shall set this bit to '1'.</p>

### 5.1.12.1.14.2 Persistent Event Log Events

The values that may be reported in the Event Type field (refer to section 5.1.12.1.14) of the event header for events in the Persistent Event log are defined in Figure 229.

**Figure 229: Persistent Event Log Event Types**

Type	O/M <sup>1</sup>	Event	Reference Section
00h		Reserved	
01h	NOTE 2	SMART / Health Log Snapshot	5.1.12.1.14.2.1
02h	M	Firmware Commit	5.1.12.1.14.2.2
03h	M	Timestamp Change	5.1.12.1.14.2.3
04h	M	Power-on or Reset	5.1.12.1.14.2.4
05h	M	NVM Subsystem Hardware Error	5.1.12.1.14.2.5
06h	NOTE 3	Change Namespace	5.1.12.1.14.2.6
07h	NOTE 3	Format NVM Start	5.1.12.1.14.2.7
08h	NOTE 3	Format NVM Completion	5.1.12.1.14.2.8
09h	NOTE 3	Sanitize Start	5.1.12.1.14.2.9
0Ah	NOTE 3	Sanitize Completion	5.1.12.1.14.2.10
0Bh	O	Set Feature	5.1.12.1.14.2.11
0Ch	O	Telemetry Log Created	5.1.12.1.14.2.12
0Dh	O	Thermal Excursion	5.1.12.1.14.2.13
0Eh	O	Sanitize Media Verification Event	5.1.12.1.14.2.14
0Fh to DDh		Reserved	
DEh	O	Vendor Specific Event	5.1.12.1.14.2.15
DFh	O	TCG Defined	5.1.12.1.14.2.16
E0h to FFh		Reserved	
Notes:			
1. O/M definition: O = Optional, M = Mandatory.			
2. Mandatory for NVMe over PCIe implementations, Optional for NVMe over Fabrics implementations.			
3. Mandatory if the command used to initiate the activity reported by the event is supported.			

#### 5.1.12.1.14.2.1 SMART / Health Log Snapshot Event (Event Type 01h)

NVM subsystems that support the Persistent Event Log shall create a SMART / Health Log Snapshot Event:

- If virtualization management is not implemented, then for every controller in the NVM subsystem; or
- If virtualization management is implemented, then for every primary controller,

at least once every 24 power on hours at a time determined by the controller.



The SMART / Health Log Snapshot Event shall set the Persistent Event Log Event Header:

- a) Event Type field to 01h; and
- b) Event Type Revision field to 01h.

The SMART / Health Log Snapshot Event data is specified in Figure 230.

**Figure 230: SMART / Health Log Snapshot Event Data Format (Event Type 01h)**

Bytes	Description
511:00	<b>Event Data (ED):</b> Contains a snapshot of the SMART/Health Information Log data specified in Figure 206.

#### 5.1.12.1.14.2.2 Firmware Commit Event (Event Type 02h)

A firmware commit event shall be recorded in the Persistent Event Log when a Firmware Commit command is completed. The Firmware Commit Event shall set the Persistent Event Log Event Format Header:

- a) Event Type field to 02h; and
- b) Event Type Revision field to 01h.

The Firmware Commit Event data is specified in Figure 231.

**Figure 231: Firmware Commit Event Data Format (Event Type 02h)**

Bytes	Description
07:00	<b>Old Firmware Revision (OFR):</b> Contains the firmware revision of the active firmware before this firmware commit event.
15:08	<b>New Firmware Revision (NFR):</b> Contains the firmware revision for the firmware that was requested to become active.
16	<b>Firmware Commit Action (FCA):</b> Contains the value from the Commit Action field in the Firmware Commit command.
17	<b>Firmware Slot (FSLT):</b> Contains the value from the Firmware Slot field in the Firmware Commit command.
18	<b>Status Code Type for Firmware Commit Command (STCTFCC):</b> Contains the status code type from the completion queue entry for the Firmware Commit command.
19	<b>Status Returned for Firmware Commit Command (SRFCC):</b> Contains the status code from the completion queue entry for the Firmware Commit command.
21:20	<b>Vendor Assigned Firmware Commit Result Code (VAFCRC):</b> Contains a vendor specific value that provides more information about the result of the firmware commit. A value of 0h indicates that no vendor assigned firmware commit result code is provided.

#### 5.1.12.1.14.2.3 Timestamp Change Event (Event Type 03h)

The Timestamp Change Event (refer to Figure 232) contains the current timestamp, reported in the event header, and the timestamp from the time at which the timestamp was changed (i.e., the old timestamp).

The Timestamp Change Event shall set the Persistent Event Log Event Format Header:

- a) Event Type field to 03h; and
- b) Event Type Revision field to 01h.

The Timestamp Change Event data is specified in Figure 232.

**Figure 232: Timestamp Change Event Format (Event Type 03h)**

Bytes	Description
07:00	<b>Previous Timestamp (PTSTP):</b> Contains a timestamp of the time immediately before the timestamp was changed (i.e., the old timestamp) using the Timestamp data structure as defined in Figure 395.
15:08	<b>Milliseconds Since Reset (MSR):</b> Contains the time since the last Controller Level Reset reported in milliseconds.

#### 5.1.12.1.14.2.4 Power-on or Reset Event (Event Type 04h)

A Power-on or Reset event shall be recorded in the Persistent Event Log when an NVM Subsystem Reset (e.g., due to a power-on) or a Controller Level Reset is completed. The Power-on or Reset Event reports information about resets due to power-on or other events that cause resets (refer to section 3.7) followed by descriptors reporting information about the controller at the time the reset occurred, including timestamp values for all controllers for use in synchronization of timestamp values across controllers.

The controller shall set the Persistent Event Log Event Format Header:

- a) Event Type field to 04h; and
- b) Event Type Revision field to 01h.

The Power-on or Reset Event data is specified in Figure 233.

**Figure 233: Power-on or Reset Event (Event Type 04h)**

Bytes	Description
07:00	<b>Firmware Revision (FREV):</b> Contains the firmware revision that becomes effective when CC.EN transitions from '0' to '1'.
EL-VSIL-1:08 <sup>1</sup>	<b>Reset Information List (RIL):</b> Contains a list of one or more Controller Reset Information descriptors (refer to Figure 234). If virtualization management is not implemented, then the list shall contain a Controller Reset Information descriptor for every controller in the NVM subsystem. If virtualization management is implemented, then the list shall contain a Controller Reset Information descriptor for every primary controller.  The Controller Reset Information descriptor is shown in Figure 234.
Notes:	
1. Refer to Figure 227 for the definitions of EL and VSIL.	

**Figure 234: Controller Reset Information descriptor**

Bytes	Description										
01:00	<b>Controller ID (CNTLID):</b> Contains the Controller ID for the controller with the timestamp in the Controller Timestamp field.										
02	<b>Firmware Activation (FA):</b> Contains a code indicating if this event triggered a firmware activation.										
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00h</td> <td>Indicates that this event did not trigger a firmware activation on the controller.</td> </tr> <tr> <td>01h</td> <td>Indicates that new firmware was activated on the controller due to this power on or reset.</td> </tr> <tr> <td>02h</td> <td>Indicates that an attempt to activate new firmware on the controller due to this power-on or reset failed.</td> </tr> <tr> <td>03h to FFh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00h	Indicates that this event did not trigger a firmware activation on the controller.	01h	Indicates that new firmware was activated on the controller due to this power on or reset.	02h	Indicates that an attempt to activate new firmware on the controller due to this power-on or reset failed.	03h to FFh	Reserved
	Value	Definition									
	00h	Indicates that this event did not trigger a firmware activation on the controller.									
01h	Indicates that new firmware was activated on the controller due to this power on or reset.										
02h	Indicates that an attempt to activate new firmware on the controller due to this power-on or reset failed.										
03h to FFh	Reserved										
03	<b>Operation in Progress (OIP):</b>										
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td><b>Reset During NVM Format (RDNF):</b> If this bit is set to '1', then a Format NVM command was in progress for a namespace attached to the controller when this reset event occurred. If this bit is cleared to '0', then no Format NVM command was in progress for any namespace attached to the controller when this reset event occurred.</td> </tr> </tbody> </table>	Bits	Description	7:1	Reserved	0	<b>Reset During NVM Format (RDNF):</b> If this bit is set to '1', then a Format NVM command was in progress for a namespace attached to the controller when this reset event occurred. If this bit is cleared to '0', then no Format NVM command was in progress for any namespace attached to the controller when this reset event occurred.				
Bits	Description										
7:1	Reserved										
0	<b>Reset During NVM Format (RDNF):</b> If this bit is set to '1', then a Format NVM command was in progress for a namespace attached to the controller when this reset event occurred. If this bit is cleared to '0', then no Format NVM command was in progress for any namespace attached to the controller when this reset event occurred.										
15:04	Reserved										
19:16	<b>Controller Power Cycle (CPWRC):</b> Contains the power cycle count for the controller indicated in the Controller ID field.										
27:20	<b>Power on milliseconds (POM):</b> Contains the power on hours in milliseconds since being manufactured. This may not include time that the controller was powered and in a non-operational power state.  The resolution of this field is vendor specific (e.g., an NVM subsystem that only counts power on time in hours only reports values corresponding to whole hours).										
35:28	<b>Controller Timestamp (CTSTP):</b> Contains a timestamp for the controller specified in the Controller ID field at the time when this event occurred using the Timestamp data structure defined in Figure 395.										

**5.1.12.1.14.2.5 NVM Subsystem Hardware Error Event (Event Type 05h)**

An NVM Subsystem Hardware Error event shall be recorded in the Persistent Event Log when a supported NVM subsystem hardware error event is detected. Which of the NVM subsystem hardware error events are supported is vendor specific. The NVM subsystem hardware error event shall set the Persistent Event Log Event Format Header:

- Event Type field to 05h; and
- Event Type Revision Field to 02h.

All detected NVM Subsystem Hardware Error events supported by the NVM subsystem shall be logged unless otherwise specified (e.g., suppressed due to reoccurrence frequency (refer to section 5.1.12.1.14)). NVM Subsystem Hardware Error event fields reporting information that is not available (e.g., due to a PCIe optional feature that is not implemented) shall be cleared to 0h unless otherwise specified in the NVM Subsystem Hardware Error Event code description.

The NVM Subsystem Hardware Error Event data is specified in Figure 235.

**Figure 235: NVM Subsystem Hardware Error Event Format (Event Type 05h)**

Bytes	Description
01:00	<b>NVM Subsystem Hardware Error Event Code (NSHEEC):</b> This field contains a code (refer to Figure 236) indicating the type of NVM subsystem hardware error that is being reported.
03:02	Reserved
M+3:04	<b>Additional Hardware Error Information (AHEI):</b> This field contains additional information about the hardware error event indicated in the NVM Subsystem Hardware Error Event Code field (refer to Figure 236). Where M is the number of bytes of additional hardware error information.  This field is omitted if the subsystem hardware error being reported does not contain additional hardware error information (i.e., if the number of bytes of additional hardware error information, M, is 0h).

**Figure 236: NVM Subsystem Hardware Error Event Codes**

Value	Definition
00h	Reserved
01h	<b>PCIe Correctable Error:</b> Indicates that the NVM subsystem has detected that a PCIe correctable error occurred.  Refer to Figure 238 for the format of the Additional Hardware Error Information field.
02h	<b>PCIe Uncorrectable Non fatal Error:</b> Indicates that the NVM subsystem has detected that a PCIe uncorrectable non-fatal error occurred.  Refer to Figure 238 for the format of the Additional Hardware Error Information field.
03h	<b>PCIe Uncorrectable Fatal Error:</b> Indicates that the NVM subsystem has detected that a PCIe uncorrectable fatal error occurred.  Refer to Figure 238 for the format of the Additional Hardware Error Information field.
04h	<b>PCIe Link Status Change:</b> Indicates that a change in the values reported in the PCI Express Link Status register (refer to the PCI Express Link Status section in the NVMe over PCIe Transport Specification) have changed due to an attempt to correct unreliable link operation.  The Additional Hardware Error Information field shall be set to the contents of the PCI Express Link Status register at the time of the event.
05h	<b>PCIe Link Not Active:</b> Indicates that the Data Link Control and Management State Machine (refer to PCI Express Base Specification) has transitioned out of the DL_Active state without a corresponding event (e.g., without an indication from the host that the link is to be disabled).  This NVM subsystem hardware error event does not contain additional hardware error information.

Figure 236: NVM Subsystem Hardware Error Event Codes

Value	Definition								
06h	<p><b>Critical Warning Condition:</b> Indicates the NVM subsystem has detected a condition that causes a bit in the Critical Warning field of the SMART / Health Information log (refer to section 5.1.12.1.3) to be set to '1'.</p> <p>Bits in this field represent the associated state at the time of this event.</p> <p>The Additional Hardware Error Information field shall be set at the time of the event using the same format as is specified for the Critical Warning field of the SMART / Health Information log.</p>								
07h	<p><b>Endurance Group Critical Warning Condition:</b> Indicates that the NVM subsystem has detected a condition that causes a bit in the Critical Warning field of an Endurance Group Information log page (refer to section 5.1.12.1.10) to be set to '1'.</p> <p>Bits in this field represent the state at the time this event is added to the Persistent Event log page.</p> <p>The Additional Hardware Error Information field shall be four bytes long and contain the following information:</p> <table border="1"> <thead> <tr> <th>Bytes</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td><b>Endurance Group Critical Warning (EGCW):</b> This field shall be set at the time this event is added to the Persistent Event log page using the same format as is specified for the Critical Warning field of the Endurance Group Information log page.</td> </tr> <tr> <td>1</td> <td>Reserved</td> </tr> <tr> <td>3:2</td> <td><b>Endurance Group Identifier (EGID):</b> This field shall be set to the Endurance Group Identifier for the associated Endurance Group.</td> </tr> </tbody> </table>	Bytes	Description	0	<b>Endurance Group Critical Warning (EGCW):</b> This field shall be set at the time this event is added to the Persistent Event log page using the same format as is specified for the Critical Warning field of the Endurance Group Information log page.	1	Reserved	3:2	<b>Endurance Group Identifier (EGID):</b> This field shall be set to the Endurance Group Identifier for the associated Endurance Group.
Bytes	Description								
0	<b>Endurance Group Critical Warning (EGCW):</b> This field shall be set at the time this event is added to the Persistent Event log page using the same format as is specified for the Critical Warning field of the Endurance Group Information log page.								
1	Reserved								
3:2	<b>Endurance Group Identifier (EGID):</b> This field shall be set to the Endurance Group Identifier for the associated Endurance Group.								
08h	<p><b>Unexpected Power Loss:</b> Indicates that the controller incremented the Unexpected Power Losses field value in the SMART / Health Information Log.</p> <p>Note that this field was previously named Unsafe Shutdown.</p> <p>Refer to Figure 237 for the format of the Additional Hardware Error Information field.</p>								
09h	<p><b>Controller Fatal Status:</b> Indicates that the Controller Fatal Status (CSTS.CFS) bit has been set to '1'.</p> <p>This NVM subsystem hardware error event does not contain additional hardware error information.</p>								
0Ah	<p><b>Media and Data Integrity Status:</b> Indicates that a completion queue entry contained a Media and Data Integrity status code (refer to Figure 106) other than 86h (i.e., Access Denied) or 87h (i.e., Deallocated or Unwritten logical Block).</p> <p>The Additional Hardware Error Information field shall be set to the contents of the completion queue entry.</p>								
0Bh	<p><b>Controller Ready Timeout Exceeded:</b> Indicates that:</p> <ol style="list-style-type: none"> <li>the controller was not ready to process at least one command without error as described in section 3.5.4.1 within the amount of time indicated by: <ul style="list-style-type: none"> <li>the Controller Ready With Media Timeout (CRTO.CRWMT) field in Controller Ready With Media mode (CC.CRIME is cleared to '0'); or</li> <li>the Controller Ready Independent of Media Timeout (CRTO.CRIMT) field in Controller Ready Independent of Media mode (CC.CRIME is set to '1'),</li> </ul> since the controller was enabled; or</li> <li>at least one namespace attached to the controller or media required to process at least one Admin command was not ready within the amount of time indicated by the Controller Ready With Media Timeout (CRTO.CRWMT) field since the controller was enabled by transitioning CC.EN from '0' to '1'.</li> </ol> <p>Refer to Figure 239 for the format of the Additional Hardware Error Information field.</p>								
0Ch to FFh	Reserved								

**Figure 237: Additional Hardware Error Information for Unexpected Power Loss Errors**

Bytes	Description					
15:0	<b>Unexpected Power Loss (UPL):</b> This field shall indicate the value from the Unexpected Power Losses field in the SMART / Health Information log at the time of the event.					
16	<b>Unexpected Power Loss Information (UPLI):</b> This field contains additional information about the unexpected power loss.					
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td><b>Unexpected Power Loss Due to Out-Of-Band Activity (UPLOA):</b> If the shutdown was unsafe because media was not in a shutdown state when main power was lost because an Admin command that accesses media as defined by Figure 84 was processed via the out-of-band mechanism with the Ignore Shutdown bit set to '1' (refer to the NVM Express Management Interface Specification) while shutdown processing was reported as in progress or was reported as complete (i.e., while the value of the CSTS.SHST field was set to 01b or 10b), then this bit shall be set to '1'; otherwise, this bit shall be cleared to '0'.</td> </tr> </tbody> </table>	Bits	Description	7:1	Reserved	0
Bits	Description					
7:1	Reserved					
0	<b>Unexpected Power Loss Due to Out-Of-Band Activity (UPLOA):</b> If the shutdown was unsafe because media was not in a shutdown state when main power was lost because an Admin command that accesses media as defined by Figure 84 was processed via the out-of-band mechanism with the Ignore Shutdown bit set to '1' (refer to the NVM Express Management Interface Specification) while shutdown processing was reported as in progress or was reported as complete (i.e., while the value of the CSTS.SHST field was set to 01b or 10b), then this bit shall be set to '1'; otherwise, this bit shall be cleared to '0'.					

**Figure 238: Additional Hardware Error Information for correctable and uncorrectable PCIe errors**

Bytes	Description					
01:00	<b>PCIe Device Status Register (PCIEDSR):</b> Contains the contents of the PCI Device Status Register (refer to the PCI Express Base Specification) at the time of the event.					
02	<b>PCIe AER Support (PCIEAS):</b>					
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td><b>PCIe AER Supported (PCIEAERS):</b> If this bit is set to '1', then the PCIe AER (refer to the PCI Express specification) is supported and that the PCIe AER Error Status field, PCIe AER Error Mask field, PCIe AER Header Log Register field, and the PCIe AER TLP Prefix Log Register field are reported. If this bit is cleared to '0', then the PCIe AER is not supported and that the PCIe AER Error Status field, PCIe AER Error Mask field, PCIe AER Header Log Register field, and PCIe AER TLP Prefix Log Register field are not reported (i.e., bytes 79:16 are not reported).</td> </tr> </tbody> </table>	Bits	Description	7:1	Reserved	0
Bits	Description					
7:1	Reserved					
0	<b>PCIe AER Supported (PCIEAERS):</b> If this bit is set to '1', then the PCIe AER (refer to the PCI Express specification) is supported and that the PCIe AER Error Status field, PCIe AER Error Mask field, PCIe AER Header Log Register field, and the PCIe AER TLP Prefix Log Register field are reported. If this bit is cleared to '0', then the PCIe AER is not supported and that the PCIe AER Error Status field, PCIe AER Error Mask field, PCIe AER Header Log Register field, and PCIe AER TLP Prefix Log Register field are not reported (i.e., bytes 79:16 are not reported).					
15:03	Reserved					
31:16	<b>PCIe AER Error Status (PCIEAES):</b> Contains the contents of:					
	<ul style="list-style-type: none"> <li>a) the PCIe AER Correctable Error Status Register (refer to the Advanced Error Reporting Capability section in the NVMe over PCIe Transport Specification) at the time of the event if the error is a correctable error; or</li> <li>b) The PCIe AER Uncorrectable Error Status Register (refer to the Advanced Error Reporting Capability section in the NVMe over PCIe Transport Specification), at the time of the event if the error is an uncorrectable error.</li> </ul>					
47:32	<b>PCIe AER Error Mask (PCIEAEM):</b> Contains the contents of:					
	<ul style="list-style-type: none"> <li>a) the PCIe AER Correctable Error Mask Register (refer to the Advanced Error Reporting Capability section in the NVMe over PCIe Transport Specification) at the time of the event if the error is a correctable error; or</li> <li>b) the PCIe AER Uncorrectable Error Mask Register (refer to the Advanced Error Reporting Capability section in the NVMe over PCIe Transport Specification) at the time of the event if the error is an uncorrectable error.</li> </ul>					
63:48	<b>PCIe AER Header Log Register (PCIEAHLR):</b> Contains the contents of the PCIe AER Header Log Register (refer to the Advanced Error Reporting Capability section in the NVMe over PCIe Transport Specification), if supported, at the time of the event.					

**Figure 238: Additional Hardware Error Information for correctable and uncorrectable PCIe errors**

Bytes	Description
79:64	<b>PCIe AER TLP Prefix Log Register (PCIeATPLR):</b> Contains the contents of the PCIe AER TLP Prefix Log Register (refer to the Advanced Error Reporting Capability section in the NVMe over PCIe Transport Specification), if supported, at the time of the event.

**Figure 239: Additional Hardware Error Information for Controller Ready Timeout Exceeded errors**

Bytes	Description												
0	<b>Controller State (CST):</b> Indicates the state of the controller at the time the Controller Ready Timeout Exceeded error occurred.												
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:4</td> <td>Reserved</td> </tr> <tr> <td>3</td> <td> <b>Controller Not Ready (CNR):</b> Indicates the controller was not ready to process at least one command without error as described in section 3.5.4.1 within the amount of time indicated by: <ul style="list-style-type: none"> <li>a) the Controller Ready With Media Timeout (CRTO.CRWM) field in Controller Ready With Media mode (CC.CRIME is cleared to '0'); or</li> <li>b) the Controller Ready Independent of Media Timeout (CRTO.CRIMT) field in Controller Ready Independent of Media mode (CC.CRIME is set to '1'),</li> </ul> since the controller was enabled. </td> </tr> <tr> <td>2</td> <td> <b>Admin Command Media Not Ready (ACMNR):</b> Indicates media required to process at least one Admin command was not ready within the amount of time indicated by the Controller Ready With Media Timeout (CRTO.CRWM) field since the controller was enabled by transitioning CC.EN from '0' to '1'. </td> </tr> <tr> <td>1</td> <td> <b>Namespace Not Ready (NNR):</b> Indicates at least one namespace attached to the controller was not ready within the amount of time indicated by the Controller Ready With Media Timeout (CRTO.CRWM) field since the controller was enabled by transitioning CC.EN from '0' to '1'. </td> </tr> <tr> <td>0</td> <td> <b>Controller Ready Independent of Media Enable (CRIME):</b> Indicates the value of the CC.CRIME bit when the Controller Ready Timeout Exceeded error occurred. </td> </tr> </tbody> </table>	Bits	Description	7:4	Reserved	3	<b>Controller Not Ready (CNR):</b> Indicates the controller was not ready to process at least one command without error as described in section 3.5.4.1 within the amount of time indicated by: <ul style="list-style-type: none"> <li>a) the Controller Ready With Media Timeout (CRTO.CRWM) field in Controller Ready With Media mode (CC.CRIME is cleared to '0'); or</li> <li>b) the Controller Ready Independent of Media Timeout (CRTO.CRIMT) field in Controller Ready Independent of Media mode (CC.CRIME is set to '1'),</li> </ul> since the controller was enabled.	2	<b>Admin Command Media Not Ready (ACMNR):</b> Indicates media required to process at least one Admin command was not ready within the amount of time indicated by the Controller Ready With Media Timeout (CRTO.CRWM) field since the controller was enabled by transitioning CC.EN from '0' to '1'.	1	<b>Namespace Not Ready (NNR):</b> Indicates at least one namespace attached to the controller was not ready within the amount of time indicated by the Controller Ready With Media Timeout (CRTO.CRWM) field since the controller was enabled by transitioning CC.EN from '0' to '1'.	0	<b>Controller Ready Independent of Media Enable (CRIME):</b> Indicates the value of the CC.CRIME bit when the Controller Ready Timeout Exceeded error occurred.
	Bits	Description											
	7:4	Reserved											
	3	<b>Controller Not Ready (CNR):</b> Indicates the controller was not ready to process at least one command without error as described in section 3.5.4.1 within the amount of time indicated by: <ul style="list-style-type: none"> <li>a) the Controller Ready With Media Timeout (CRTO.CRWM) field in Controller Ready With Media mode (CC.CRIME is cleared to '0'); or</li> <li>b) the Controller Ready Independent of Media Timeout (CRTO.CRIMT) field in Controller Ready Independent of Media mode (CC.CRIME is set to '1'),</li> </ul> since the controller was enabled.											
2	<b>Admin Command Media Not Ready (ACMNR):</b> Indicates media required to process at least one Admin command was not ready within the amount of time indicated by the Controller Ready With Media Timeout (CRTO.CRWM) field since the controller was enabled by transitioning CC.EN from '0' to '1'.												
1	<b>Namespace Not Ready (NNR):</b> Indicates at least one namespace attached to the controller was not ready within the amount of time indicated by the Controller Ready With Media Timeout (CRTO.CRWM) field since the controller was enabled by transitioning CC.EN from '0' to '1'.												
0	<b>Controller Ready Independent of Media Enable (CRIME):</b> Indicates the value of the CC.CRIME bit when the Controller Ready Timeout Exceeded error occurred.												
3:1	Reserved												

**5.1.12.1.14.2.6 Change Namespace Event (Event Type 06h)**

The Changed Namespace Event (refer to Figure 240) persists the host parameters used for successful Namespace Management commands. This event type is specific to namespaces that are associated with I/O Command Sets that specify logical blocks (e.g., the NVM Command Set and the Zoned Namespace Command Set). The event contains a Persistent Event Log Event header and the Change Namespace Event data.

The Changed Namespace Event shall set the Persistent Event Log Event Format Header:

- Event Type field to 06h; and
- Event Type Revision Field to 02h.

**Figure 240: Change Namespace Event Data Format (Event Type 06h)**

Bytes	Description
03:00	<b>Namespace Management CDW10 (NMCDW10):</b> Contains the value from command Dword 10 of the Namespace Management command that initiated the namespace change event (refer to Figure 367).
07:04	Reserved

**Figure 240: Change Namespace Event Data Format (Event Type 06h)**

Bytes	Description
15:08	<b>Namespace Size (NSZE):</b> For a create operation, contains the NSZE value from the Identify Namespace data structure in the Namespace Management command (refer to Figure 369). For a delete operation that specifies a single namespace this field contains the value from the NSZE field of the Identify Namespace data structure (refer to the I/O Command Set specific Identify Namespace data structure in the applicable I/O Command Set specification) for the namespace being deleted. For a delete operation that specifies all namespaces this field is reserved.
23:16	Reserved
31:24	<b>Namespace Capacity (NCAP):</b> For a creation operation, contains the NCAP value from the Identify Namespace data structure in the Namespace Management command (refer to Figure 369). For a delete operation that specifies a single namespace this field contains the value from the NCAP field of the Identify Namespace data structure (refer to the I/O Command Set specific Identify Namespace data structure in the applicable I/O Command Set specification) for the namespace being deleted. For a delete operation that specifies all namespaces this field is reserved. For I/O Command Sets that do not define this field, this field is considered reserved.
32	<b>Formatted LBA Size (FLBAS):</b> Refer to the applicable I/O Command Set specification for details. For I/O Command Sets that do not define this field, this field is considered reserved.  NOTE: This field applies to all User Data Formats. The original name has been retained for historical continuity.
33	<b>End-to-end Data Protection Type Settings (DPS):</b> Refer to the applicable I/O Command Set specification for details. For I/O Command Sets that do not define this field, this field is considered reserved.
34	<b>Namespace Multi-path I/O and Namespace Sharing Capabilities (NMIC):</b> For a create operation, contains the NMIC value from the Namespace Management – Host Software Specified Fields data structure defined in the applicable I/O Command Set specification. For a delete operation that specifies a single namespace this field contains the value from the NMIC field of the I/O Command Set Independent Identify Namespace data (refer to Figure 319 <sup>1</sup> ) for the namespace being deleted. For a delete operation that specifies all namespaces this field is reserved.
35	Reserved
39:36	<b>ANA Group Identifier (ANAGRPID):</b> For a create operation, contains the ANAGRPID value from the Namespace Management – Host Software Specified Fields data structure defined in the applicable I/O Command Set specification, if specified, or contains the ANAGRPID value from the I/O Command Set Independent Identify Namespace data (refer to Figure 319 <sup>1</sup> ) after the namespace was created if an ANA Group Identifier was not specified in the command. For a delete operation that specifies a single namespace this field contains the value from the ANAGRPID field of the I/O Command Set Independent Identify Namespace data (refer to Figure 319 <sup>1</sup> ) for the namespace being deleted. For a delete operation that specifies all namespaces this field is reserved.  If ANA Groups are not supported, then the ANAGRPID field shall be cleared to 0h.
41:40	<b>NVM Set Identifier (NVMSETID):</b> For a create operation, contains the NVMSETID value from the Namespace Management – Host Software Specified Fields data structure defined in the applicable I/O Command Set specification. For a delete operation that specifies a single namespace this field contains the value from the NVMSETID field of the Identify Namespace data (refer to Figure 319 <sup>1</sup> ) for the namespace being deleted. For a delete operation that specifies all namespaces this field is reserved.
43:42	<b>Endurance Group Identifier (ENDGID):</b> For a create operation, contains the ENDGID value from the Namespace Management – Host Software Specified Fields data structure defined in the applicable I/O Command Set specification. For a delete operation that specifies a single namespace, this field contains the value from the ENDGID field of the Identify Namespace data structure (refer to Figure 319 <sup>1</sup> ) for the namespace being deleted. For a delete operation that specifies all namespaces this field is reserved.
47:44	<b>Namespace ID (NSID):</b> For a create operation, contains the NSID for the namespace that was created. For a delete operation, contains the NSID for the namespace being deleted or FFFFFFFFh for a delete operation specifying all namespaces.

**Figure 240: Change Namespace Event Data Format (Event Type 06h)**

Bytes	Description
Notes:	
1. For controllers that implement the NVM Command Set, this field contains the value of the associated field from the Identify Namespace data structure (refer to the NVM Command Set Identify Namespace data structure in the NVM Command Set Specification).	

**5.1.12.1.14.2.7 Format NVM Start Event (Event Type 07h)**

A Format NVM Start event shall be recorded in the Persistent Event Log after successfully validating the command parameters of a Format NVM command (refer to section 5.1.10) and before modifying any of the contents of the NVM.

The Format NVM Start event shall set the Persistent Event Log Event Format Header:

- Event Type field to 07h; and
- Event Type Revision field to 01h.

**Figure 241: Format NVM Start Event Data Format (Event Type 07h)**

Bytes	Description
03:00	<b>Namespace Identifier (NSID):</b> Contains the namespace identifier specified in the Format NVM command.
04	<b>Format NVM Attributes (FNA):</b> Contains the value from the identify controller FNA field.
07:05	Reserved
11:08	<b>Format NVM CDW10 (FMCDW10):</b> Contains the value from command Dword 10 of the Format NVM command (refer to Figure 189).

**5.1.12.1.14.2.8 Format NVM Completion Event (Event Type 08h)**

A Format NVM Completion event shall be recorded in the Persistent Event Log at the completion of a Format NVM command that resulted in modification of the contents of the NVM.

The Format NVM Completion event shall set the Persistent Event Log Event Format Header:

- Event Type field to 08h; and
- Event Type Revision field to 02h.

**Figure 242: Format NVM Completion Event Data Format (Event Type 08h)**

Bytes	Description	
03:00	<b>Namespace Identifier (NSID):</b> Contains the namespace identifier specified in the Format NVM command.	
04	<b>Smallest Format Progress Indicator (SFPI):</b> For a Format NVM command that formats a single namespace, this field contains the smallest numerical value that was available for reporting in the Remaining Format NVM (RFNVM) field of the FPI field of the I/O Command Set Independent Identify Namespace data structure (i.e., if the format did not complete successfully and the FPI field is supported, then this field contains the percentage of the namespace that remained to be formatted at the time the Format NVM command completed, refer to Figure 319) during the format operation. For a Format NVM command that formats all namespaces this field shall be cleared to 0h.	
05	<b>Format NVM Status (FNVMS):</b> Contains the status of the format operation.	
	<b>Bits</b>   <b>Description</b>	
	7:2	Reserved
	1	<b>Incomplete Format (INCPLTF):</b> If this bit is set to '1', then the format operation modified some or all of the user data but did not complete successfully. If this bit is set to '1', then the Format NVM Error bit shall be set to '1'. If this bit is cleared to '0', then the format operation either did not modify any user data or the format operation completed successfully.
0	<b>Format NVM Error (FNVME):</b> If this bit is set to '1', then the format operation did not complete successfully. If this bit is cleared to '0', then the format operation completed successfully.	



**Figure 242: Format NVM Completion Event Data Format (Event Type 08h)**

Bytes	Description						
07:06	<b>Completion Information (CINFO):</b> Contains a vendor specific value that may provide more information about the completion of the format operation (e.g., if the format operation did not complete successfully, then this field may contain a vendor specific code that indicates a vendor specific reason).						
09:08	<b>Status Info (INFO):</b> Identifies the status for the NVM Format command.						
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:1</td> <td><b>Status (STATUS):</b> This field indicates the value that was reported in the Status field for the completion queue entry (refer to Figure 100), if any, for the Format NVM command associated with this event. If no completion queue entry was reported, then this field shall be cleared to 0h.</td> </tr> <tr> <td>0</td> <td><b>Phase Tag (P):</b> This field may indicate the Phase Tag posted for the command.</td> </tr> </tbody> </table>	Bits	Description	15:1	<b>Status (STATUS):</b> This field indicates the value that was reported in the Status field for the completion queue entry (refer to Figure 100), if any, for the Format NVM command associated with this event. If no completion queue entry was reported, then this field shall be cleared to 0h.	0	<b>Phase Tag (P):</b> This field may indicate the Phase Tag posted for the command.
	Bits	Description					
15:1	<b>Status (STATUS):</b> This field indicates the value that was reported in the Status field for the completion queue entry (refer to Figure 100), if any, for the Format NVM command associated with this event. If no completion queue entry was reported, then this field shall be cleared to 0h.						
0	<b>Phase Tag (P):</b> This field may indicate the Phase Tag posted for the command.						
11:10	Reserved						

**5.1.12.1.14.2.9 Sanitize Start Event (Event Type 09h)**

A Sanitize Start event shall be recorded in the Persistent Event Log at the start of a sanitize operation (i.e., the sanitization target transitions to the Restricted Processing state or the Unrestricted Processing state, as described in section 8.1.24.3).

The Sanitize Start event shall set the Persistent Event Log Event Format Header:

- Event Type field to 09h; and
- Event Type Revision field to 01h.

**Figure 243: Sanitize Start Event Data Format (Event Type 09h)**

Bytes	Description
03:00	<b>Sanitize Capabilities (SANICAP):</b> Contains the contents of the SANICAP field from the Identify Controller data structure.
07:04	<b>Sanitize CDW10 (SCDW10):</b> Contains the value from command Dword 10 of the Sanitize command (refer to Figure 372).
11:08	<b>Sanitize CDW11 (SCDW11):</b> Contains the value from command Dword 11 of the Sanitize command (refer to Figure 373).

**5.1.12.1.14.2.10 Sanitize Completion Event (Event Type 0Ah)**

A Sanitize Completion event shall be recorded in the Persistent Event Log at the completion of a sanitize operation (i.e., the sanitization target transitions to the Idle state, the Restricted Failure state, or the Unrestricted Failure state, as described in section 8.1.24.3).

The Sanitize Completion event shall set the Persistent Event Log Event Format Header:

- Event Type field to 0Ah; and
- Event Type Revision field to 01h.

**Figure 244: Sanitize Completion Event Data Format (Event Type 0Ah)**

Bytes	Description
1:0	<b>Sanitize Progress (SPROG):</b> Contains the sanitize progress at the time of this event using the format specified for the SPROG field in the Sanitize Status log page (refer to section 5.1.12.1.33).
3:2	<b>Sanitize Status (SSTAT):</b> Contains the sanitize status for the time of this event using the format specified for the SSTAT field in the Sanitize Status log page. (e.g., the Global Data Erase bit indicates the status at the time of this event).
5:4	<b>Completion Information (CINFO):</b> Contains a vendor specific value that may provide more information about the completion of the sanitize operation (e.g., if the sanitize operation did not complete successfully, then this field may contain a vendor specific code that indicates a vendor specific reason).
7:6	Reserved

### 5.1.12.1.14.2.11 Set Feature Event (Event Type 0Bh)

The Set Feature Event persists the data of a successful Set Features command. The event contains a Persistent Event Log Event header and the Set Feature Event data (refer to Figure 246).

The Set Feature Event shall set the Persistent Event Log Event Format Header:

- Event Type field to 0Bh; and
- Event Type Revision Field to 01h.

A Set Feature Event shall be recorded in the Persistent Event Log when the following criteria are met:

- A Set Features command completes successfully;
- The Feature Identifier in that Set Features command is supported to be logged in the Persistent Event Log; and
- There is a change to the controller settings for the Feature Identifier in that Set Features command (i.e., the same setting is not set again).

A Set Feature Event may be recorded in the Persistent Event Log when there is no change to the controller settings for the Feature Identifier in that Set Features command if the following criteria are met:

- A Set Features command completes successfully; and
- The Feature Identifier in that Set Features command is supported to be logged in the Persistent Event Log.

The Feature Identifiers that may be supported to be logged in the Persistent Event Log are shown in Figure 245.

**Figure 245: Feature Persistent Event Logging Requirements**

Feature Name	Feature Identifier	Controller PE Logging Requirements <sup>1</sup>		
		I/O	Admin	Discovery
Arbitration	01h	O	P	P
Power Management	02h	NR	NR	P
	03h	Refer to the applicable I/O Command Set specification		
Temperature Threshold	04h	O	O	P
	05h	Refer to the applicable I/O Command Set specification		
Volatile Write Cache	06h	O	P	P
Number of Queues	07h	O	P	P
Interrupt Coalescing	08h	O	O	P
Interrupt Vector Configuration	09h	O	O	P
	0Ah	Refer to the applicable I/O Command Set specification		
Asynchronous Event Configuration	0Bh	NR	NR	NR
Autonomous Power State Transition	0Ch	O	O	P
Host Memory Buffer	0Dh	O	O	P
Timestamp	0Eh	P	P	P
Keep Alive Timer	0Fh	O	O	O
Host Controlled Thermal Management	10h	O	O	P
Non-Operational Power State Config	11h	O	O	P
Read Recovery Level Config	12h	O	O	P
Predictable Latency Mode Config	13h	O	P	P
Predictable Latency Mode Window	14h	P	O	P
	15h	Refer to the applicable I/O Command Set specification		
Host Behavior Support	16h	O	O	P
Sanitize Config	17h	O	O	P
Endurance Group Event Configuration	18h	O	O	P
I/O Command Set Profile	19h	O	P	P
Spinup Control	1Ah	O	P	P
Power Loss Signaling Config	1Bh	?	?	?
	1Ch	Refer to the applicable I/O Command Set specification		

**Figure 245: Feature Persistent Event Logging Requirements**

Feature Name	Feature Identifier	Controller PE Logging Requirements <sup>1</sup>		
		I/O	Admin	Discovery
Flexible Data Placement	1Dh	O	P	P
Flexible Data Placement Events	1Eh	O	P	P
Namespace Admin Label	1Fh	O	O	P
	20h	Refer to the applicable I/O Command Set specification		
Embedded Management Controller Address	78h	O	O	O
Host Management Agent Address	79h	O	O	O
Enhanced Controller Metadata	7Dh	O	O	O
Controller Metadata	7Eh	O	O	O
Namespace Metadata	7Fh	O	O	O
Software Progress Marker	80h	NR	NR	P
Host Identifier	81h	O	O	P
Reservation Notification Mask	82h	O	P	P
Reservation Persistence	83h	O	P	P
Namespace Write Protection Config	84h	O	O	P
Boot Partition Write Protection Config	85h	O	O	P

Notes:  
 1. O/M/P/NR definition: O = Optional, M = Mandatory, P = Prohibited, NR = Not Recommended

The Command Dwords and Memory Buffer logged in the Set Feature Event data use the same formats as the formats defined by the Set Features and Get Features commands.

**Figure 246: Set Feature Event Data Format**

Bytes	Description										
03:00	<b>Set Feature Event Layout (SFEL):</b> Defines the number of Command Dwords and the amount of data in the Memory Buffer from the Set Features command associated with this event.										
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>31:16</td> <td><b>Memory Buffer Count (MBC):</b> Defines the number of bytes from the memory buffer that are logged in the Memory Buffer field. A value of 0h indicates that the Memory Buffer field does not exist.</td> </tr> <tr> <td>15:04</td> <td>Reserved</td> </tr> <tr> <td>03</td> <td><b>Logged Command Completion Dword 0 (LCCDW0):</b> If this bit is set to '1', then Dword 0 of the command completion for the Set Features command is included in the log. If this bit is cleared to '0', then Dword 0 of the command completion command for the Set Features command is not included in the log.</td> </tr> <tr> <td>02:00</td> <td><b>Dword Count (DWC):</b> contains the number of consecutive Dwords starting with Dword 10 from the Set Features command that are reported in the Command Dwords field. The values 0h and 7h are reserved.</td> </tr> </tbody> </table>	Bits	Description	31:16	<b>Memory Buffer Count (MBC):</b> Defines the number of bytes from the memory buffer that are logged in the Memory Buffer field. A value of 0h indicates that the Memory Buffer field does not exist.	15:04	Reserved	03	<b>Logged Command Completion Dword 0 (LCCDW0):</b> If this bit is set to '1', then Dword 0 of the command completion for the Set Features command is included in the log. If this bit is cleared to '0', then Dword 0 of the command completion command for the Set Features command is not included in the log.	02:00	<b>Dword Count (DWC):</b> contains the number of consecutive Dwords starting with Dword 10 from the Set Features command that are reported in the Command Dwords field. The values 0h and 7h are reserved.
	Bits	Description									
	31:16	<b>Memory Buffer Count (MBC):</b> Defines the number of bytes from the memory buffer that are logged in the Memory Buffer field. A value of 0h indicates that the Memory Buffer field does not exist.									
15:04	Reserved										
03	<b>Logged Command Completion Dword 0 (LCCDW0):</b> If this bit is set to '1', then Dword 0 of the command completion for the Set Features command is included in the log. If this bit is cleared to '0', then Dword 0 of the command completion command for the Set Features command is not included in the log.										
02:00	<b>Dword Count (DWC):</b> contains the number of consecutive Dwords starting with Dword 10 from the Set Features command that are reported in the Command Dwords field. The values 0h and 7h are reserved.										
(Dword Count * 4)+3:4	<b>Command Dwords (CDWS):</b> Contains a sequential list of Command Dwords from the Set Features command starting with Command Dword 10. The number of entries in the list is specified by the DWC field. All non-reserved Command Dwords specified by the Set Features command for the Feature Identifier shall be logged. The Command Dwords are ordered as defined by the Common Command Format in Figure 92.										
(Memory Buffer Count) + (Dword Count * 4) + 3: (Dword Count * 4) + 4	<b>Memory Buffer (MBUF):</b> Contains the data in the memory buffer for the Set Features command. If the Memory Buffer Count field is cleared to 0h, then this field does not exist in the logged event.										

**Figure 246: Set Feature Event Data Format**

Bytes	Description
Memory Buffer Count + (Dword Count * 4) + 7: Memory Buffer Count + (Dword Count * 4) + 4	<b>Command Completion Dword 0 (CCDW0):</b> If the Logged Command Completion Dword 0 bit is set to '1', then this field contains the Dword 0 value from the Set Features command completion. If the Logged Command Completion Dword 0 bit is cleared to '0', then this field is not logged.

**5.1.12.1.14.2.12 Telemetry Log Create Event (Event Type 0Ch)**

A Telemetry Log Create event may be created if the controller determines that a Telemetry Host-Initiated log page (refer to section 5.1.12.1.8) or that a Telemetry Controller-Initiated log page (refer to section 5.1.12.1.9) has been generated.

The Telemetry Log Create Event shall set the Persistent Event Log Event Format Header:

- Event Type field to 0Ch; and
- Event Type Revision Field to 01h.

**Figure 247: Telemetry Log Create Event Data Format (Event Type 0Ch)**

Bytes	Description
511:00	<b>Telemetry Initiated Log (TIL):</b> Contains a copy of the values from the first 512 bytes of the Telemetry Host-Initiated log page (refer to Figure 214) or the first 512 bytes of the Telemetry Controller-Initiated log page (refer to Figure 216).

**5.1.12.1.14.2.13 Thermal Excursion Event (Event Type 0Dh)**

A Thermal Excursion event shall be recorded in the Persistent Event Log if the Composite Temperature has transitioned from a temperature that is less than:

- the WCTEMP, if any, (refer to Figure 312) to a temperature that is greater than or equal to the WCTEMP, if any; or
- the CCTEMP, if any, (refer to Figure 312), to a temperature that is greater than or equal to the CCTEMP, if any,

unless recording of the event causes a vendor specific frequency of threshold reports for this threshold to be exceeded.

A Thermal Excursion event may be recorded in the Persistent Event Log if the Composite Temperature has transitioned from a temperature:

- that is less than TMT1 (refer to section 5.1.25.1.9), if any, to a temperature that is greater than or equal to TMT1, if any (i.e., light throttling has started);
- that is less than TMT2 (refer to section 5.1.25.1.9), if any, to a temperature that is greater than or equal to TMT2, if any (i.e., heavy throttling has started);
- that is less than a vendor specific temperature where thermal throttling occurs due to self-throttling to a temperature that is greater than that vendor specific temperature (i.e., self-throttling has started);
- outside of a temperature threshold to a value that is within all temperature thresholds (i.e., the temperature returns to normal);
- at which thermal throttling is occurring to a temperature at which thermal throttling is stopped; or
- that is greater than an under temperature threshold (refer to section 5.1.25.1.3) to a temperature that is less than or equal to an under temperature threshold,

unless recording of the event causes a vendor specific frequency of threshold reports for this threshold to be exceeded.

The Thermal Excursion event shall set the Persistent Event Log Event Format Header;

- Event Type field to 0Dh; and

- Event Type Revision field to 01h.

**Figure 248: Thermal Excursion Event Data Format (Event Type 0Dh)**

Bytes	Description	
0	<b>Over Temperature (OTMP):</b> Contains the absolute value of the difference (i.e., delta) in Kelvins between the temperature indicated in the threshold field and temperature measured at the time of the event.	
1	<b>Threshold (THRESH):</b> Contains an indicator for the temperature threshold crossing that is being reported.	
	<b>Value</b>	<b>Definition</b>
	<b>High Temperature Transitions</b>	
	1h	The Composite Temperature has transitioned from a temperature that is less than WCTEMP to a temperature that is greater than or equal to WCTEMP.
	2h	The Composite Temperature has transitioned from a temperature that is less than CCTEMP to a temperature that is greater than or equal to CCTEMP.
	3h	The Composite Temperature has transitioned from a temperature that is less than TMT1 to a temperature is greater than or equal to TMT1 (i.e., vendor specific thermal management actions that minimize the impact on performance, such as light throttling, have started).
	4h	The Composite Temperature has transitioned from a temperature that is less than TMT2 to a temperature that is greater than or equal to TMT2 (i.e., vendor specific thermal management actions that may impact performance, such as heavy throttling, have started).
	5h	The Composite Temperature has transitioned from a temperature where no vendor specific thermal management actions are taken to a temperature that is greater than or equal to a vendor specific temperature at which vendor specific thermal management actions have started (e.g., self-throttling).
	<b>Normal Temperature Transitions</b>	
	88h	The Composite Temperature has transitioned from a temperature that is greater than or equal to WCTEMP or is less than or equal to an under temperature threshold to a temperature that is between WCTEMP and that under temperature threshold (i.e., the temperature has transitioned to a normal temperature).
	89h	The Composite Temperature has transitioned from a temperature that is greater than a temperature where thermal management actions are being performed and that is not greater than or equal to WCTEMP to a temperature where thermal management actions are stopped.
	<b>Low Temperature Transitions</b>	
	B0h	The Composite Temperature has transition from a temperature that is greater than an under temperature threshold to a temperature that is less than or equal to an under temperature threshold.
	F0h to FFh	Vendor specific
All other values	Reserved	

#### 5.1.12.1.14.2.14 Sanitize Media Verification Event (Event Type 0Eh)

A Sanitize Media Verification event shall be recorded in the Persistent Event log page upon transition to the Media Verification state (refer to section 8.1.24.3.6).

No Event Data (refer to Figure 227) is defined for this event.

The Sanitize Media Verification event shall set the Persistent Event Log Event Format Header:

- Event Type field to 0Eh;
- Event Type Revision field to 01h; and
- Event Length field to 0h.

#### 5.1.12.1.14.2.15 Vendor Specific Event (Event Type DEh)

The Vendor Specific Event (refer to Figure 249) contains a set of Vendor Specific Event Descriptors that describe an event that the vendor has determined is a significant event which should be reported to a host in the persistent event log and that is not described by any of the other persistent event log events.

The Vendor Specific Event Descriptors follow the format shown in Figure 250 and contain vendor specific data of the type indicated in the Vendor Specific Event Data Type field of the Vendor Specific Event Descriptor.

If a UUID Index is specified in the Get Log Page command (refer to section 5.1.12), then the controller shall return:

- a) Vendor specific events based on that UUID index; and
- b) Vendor specific events defined by the NVM subsystem manufacturer.

The controller shall set the Vendor Specific Event Format Header:

- a) Event Type field to DEh; and
- b) Event Type Revision field to 01h.

The Vendor Specific Event data is specified in Figure 249.

**Figure 249: Vendor Specific Event Format (Event Type DEh)**

Bytes	Description
<b>Vendor Specific Event Descriptor List</b>	
M-1:0	<b>Vendor Specific Event Descriptor 0:</b> Contains the first vendor specific event descriptor (refer to Figure 250). Where M is the length of this vendor specific event descriptor.
...	...
EL-VSIL-1: EL-VSIL-K <sup>1</sup>	<b>Vendor Specific Event Descriptor N:</b> Contains the last vendor specific event descriptor (refer to Figure 250). Where K is the length of this vendor specific event descriptor (refer to Figure 250).
Notes:	
1. Refer to Figure 227 for the definitions of EL and VSIL.	

The format of the Vendor Specific Event Descriptor is shown in Figure 250.

**Figure 250: Vendor Specific Event Descriptor Format**

Bytes	Description
01:00	<b>Vendor Specific Event Code (VSEC):</b> Contains a vendor specific code that uniquely identifies the type of event that is described in the data that follows. All vendor specific events of the same type should report the same Vendor Specific Event Code field value.
02	<b>Vendor Specific Event Data Type (VSEDt):</b> Contains a code indicating the type of data reported in the Vendor Specific Event Data field (refer to Figure 251).
03	<b>UUID Index (UIDX):</b> UUID Index (refer to Figure 658) at the time of this event for the vendor that defined this event.
05:04	<b>Vendor Specific Event Data Length (VSEDL):</b> Contains the length in bytes of the Vendor Specific Event Data field.
<b>Vendor Specific Event Data, if any (i.e., VSEDL &gt; 0)</b>	
VSEDL+5:06	<b>Vendor Specific Event Data (VSED):</b> Contains vendor specific data that is associated with this event and is of the type specified in the Vendor Specific Event Data Type field.

The Vendor Specific Event Data Types that are able to be reported in a Vendor Specific Event Descriptor are shown in Figure 251.

**Figure 251: Vendor Specific Event Data Type Codes**

Value	Description
00h	Reserved

**Figure 251: Vendor Specific Event Data Type Codes**

Value	Description
01h	<b>Event Name:</b> The Vendor Specific Event Data field contains a null terminated ASCII string with a vendor specific name for the value in the Vendor Specific Event Code field.  The value reported in this field shall be the same for every vendor specific event containing a vendor specific event code that is the same as the value in the Vendor Specific Event Code field in this event.  If a Vendor Specific Event Descriptor specifying this data type is reported, then that descriptor shall be the first Vendor Specific Event Descriptor in that event.
02h	<b>ASCII String:</b> The Vendor Specific Event Data field contains a null terminated ASCII string.
03h	<b>Binary:</b> The Vendor Specific Event Data field contains binary data. The byte ordering in the binary data is determined by the NVM subsystem vendor.
04h	<b>Signed Integer:</b> The Vendor Specific Event Data field contains a 64-bit signed integer in two's complement form.
05h to FFh	Reserved

**5.1.12.1.14.2.16 TCG Defined Event (Event Type DFh)**

The TCG Defined Event shall set the Persistent Event Log Event Format Header:

- Event Type field to DFh.

The Event Type Revision Field and the TCG Defined Event data are reserved for TCG.

**5.1.12.1.15 Endurance Group Event Aggregate (Log Page Identifier 0Fh)**

This log page indicates if an Endurance Group Event (refer to section 3.2.3) has occurred for a particular Endurance Group. If an Endurance Group Event has occurred, the details of the particular event are included in the Endurance Group Information log page for that Endurance Group. An asynchronous event is generated when an entry for an Endurance Group is newly added to this log page.

If there is an enabled Endurance Group Event pending for an Endurance Group, then the Endurance Group Event Aggregate log page includes an entry for that Endurance Group. The log page is an ordered list by Endurance Group Identifier. For example, if Endurance Group Events are pending for Endurance Group 2, 1, and 7, then the log page shall have entries in numerical order of 1, 2, and 7. A particular Endurance Group entry is removed from this log page after the Get Log Page command is completed successfully with the Retain Asynchronous Event bit cleared to '0' for the Endurance Group Information log page for that Endurance Group.

The log page size is limited by the Endurance Group Identifier Maximum value reported in the Identify Controller data structure (refer to Figure 312). If the host reads beyond the end of the log page, zeroes are returned. The log page is defined in Figure 252.

**Figure 252: Endurance Group Event Aggregate Log Page**

Bytes	Description
<b>Header</b>	
07:00	<b>Number of Entries (NUMENT):</b> This field indicates the number of entries in the list. The maximum number of entries in the list corresponds to the Endurance Group Identifier Maximum field reported in the Identify Controller data structure. A value of 0h indicates there are no entries in the list.
<b>Endurance Group Identified List</b>	
09:08	<b>Entry 1:</b> Indicates the Endurance Group that has an Endurance Group Event pending that has the numerically smallest Endurance Group Identifier, if any.
11:10	<b>Entry 2:</b> Indicates the Endurance Group that has an Endurance Group Event pending that has the second numerically smallest Endurance Group Identifier, if any.
...	...
2*NUMENT+7: 2*NUMENT+6	<b>Entry NUMENT:</b> Indicates the Endurance Group that has an Endurance Group Event pending that has the numerically largest Endurance Group Identifier, if any.

### 5.1.12.1.16 Media Unit Status (Log Page Identifier 10h)

This log page is used to describe the configuration and wear of Media Units (refer to section 8.1.4). The log page contains one Media Unit Status Descriptor for each Media Unit accessible by the specified domain. Each Media Unit Status Descriptor (refer to Figure 255) indicates the configuration of the Media Unit (e.g., to which Endurance Group the Media Unit is assigned, to which NVM Set the Media Unit is assigned, to which Channels the Media Unit is attached) and indications of wear (e.g., the Available Spare field and the Percentage Used field). The indications of wear change as the Media Unit is written and read.

If the NVM subsystem supports multiple domains, then the controller reports the Media Unit Status log page for the domain specified in the Log Specific Identifier field (refer to Figure 253), if accessible. If the information is not accessible, then the log page is not available (refer to section 8.1.1.10). If the Log Specific Identifier field is cleared to 0h, then the specified domain is the domain containing the controller that is processing the command.

Media Unit Identifier values (refer to Figure 255) begin with 0h and increase sequentially. If the NVM subsystem supports multiple domains, then the Media Unit Identifier values are unique within the specified domain. If the NVM subsystem does not support multiple domains, then the Media Unit Identifier values are unique within the NVM subsystem.

Media Unit Status Descriptors are listed in ascending order by Media Unit Identifier.

Requirements for supporting the Media Unit Status log page are defined in section 8.1.4.

**Figure 253: Domain Identifier – Log Specific Identifier**

Bits	Description
15:00	<p><b>Domain Identifier (DID):</b> This field specifies the identifier for the domain (refer to section 3.2.5) used for this log page.</p> <p>If the NVM subsystem does not support multiple domains, then this field is reserved.</p> <p>If this field specifies a non-zero Domain Identifier that is not reported in the Domain List (refer to section 5.1.13.2.15), then the controller shall abort the command with a status code of Invalid Field in Command.</p>

**Figure 254: Media Unit Status Log Page**

Bytes	Description
<b>Header</b>	
01:00	<b>Number of Media Unit Status Descriptors (NMU):</b> This field indicates the number of Media Unit Status Descriptors in the log page. If this field is cleared to 0h, then no Media Unit Status Descriptors are reported.
03:02	<b>Number of Channels (CCHANS):</b> This field indicates the number of Channels accessible by the controller. A value of 0h indicates that the number of Channels accessible by the controller is not reported.
05:04	<b>Selected Configuration (SELC):</b> This field indicates the Configuration Identifier selected by the most recent successful completion of the Capacity Management command Select Capacity Configuration operation. If a Select Capacity Configuration operation has not been completed, this field may indicate a non-zero value (i.e., a configuration was selected by default). If a Configuration Identifier is not selected, then this field shall be cleared to 0h.
15:06	Reserved
<b>Media Unit Status Descriptor List</b>	
NOTE 1	<b>Media Unit Status Descriptor 0:</b> This field contains the first Media Unit Status Descriptor (refer to Figure 255), if any.
NOTE 1	<b>Media Unit Status Descriptor 1:</b> This field contains the second Media Unit Status Descriptor, if any.
...	
NOTE 1	<b>Media Unit Status Descriptor NMU-1:</b> This field contains the last Media Unit Status Descriptor, if any.
Notes:	
1. Media Unit Status Descriptors may be different lengths.	

The Media Unit Status Descriptor is defined in Figure 255.



If the Selected Configuration field is cleared to 0h, then the following fields in each Media Unit Status Descriptor shall be cleared to 0h:

- a) Endurance Group Identifier (ENDGID);
- b) NVM Set Identifier (NVMSETID);
- c) Capacity Adjustment Factor; and
- d) Number of Channels (MUCS).

Channel Identifier values begin with 0h and increase sequentially. If the NVM subsystem supports multiple domains, then Channel Identifier values are unique within the specified domain. If the NVM subsystem does not support multiple domains, then Channel Identifier values are unique within the NVM subsystem.

In the Media Unit Status Descriptor, Channel Identifiers are listed in ascending order by value, and each Channel Identifier shall appear only once.

**Figure 255: Media Unit Status Descriptor**

Bytes	Description
<b>Header</b>	
01:00	<b>Media Unit Identifier (MUID):</b> This field indicates the identifier of the Media Unit.
03:02	<b>Domain Identifier (DID):</b> This field indicates the identifier of the domain that contains this Media Unit. Refer to section 3.2.5.3. A value of 0h indicates that a Domain Identifier is not reported. If multiple domains are not supported, then this field shall be cleared to 0h.
05:04	<b>Endurance Group Identifier (ENDGID):</b> This field indicates the identifier of the Endurance Group that contains this Media Unit. Refer to section 3.2.3. The value shall be less than or equal to the value of the Endurance Group Identifier Maximum field (refer to Figure 312). A value of 0h indicates that this Media Unit is not part of an Endurance Group.
07:06	<b>NVM Set Identifier (NVMSETID):</b> This field indicates the identifier of the NVM Set that contains this Media Unit. Refer to section 3.2.2. This field shall indicate a value less than or equal to the value of the NVM Set Identifier Maximum field (refer to Figure 312). If the controller does not support NVM Sets, then this field shall be cleared to 0h.
09:08	<b>Capacity Adjustment Factor (CAF):</b> This field indicates the capacity adjustment factor (refer to section 8.1.4.1) for this Endurance Group. A value of FFFFh indicates that value and all higher values. A value of 0h indicates that the Capacity Adjustment Factor is not reported. All Media Unit Status Descriptors which indicate the same Endurance Group Identifier shall indicate the same value in their Capacity Adjustment Factor fields.
10	<b>Available Spare (AVSP):</b> Contains a normalized percentage (0 to 100%) of the remaining spare capacity available for the Media Unit. The relationship between this value and the value in the Available Spare field in the Endurance Group Information log page (refer to section 5.1.12.1.10) is outside the scope of this specification.
11	<b>Percentage Used (PUSED):</b> Contains a vendor specific estimate of the percentage of life used for the Media Unit based on the actual usage and the manufacturer's prediction of NVM life. A value of 100 indicates that the estimated endurance of the NVM in the Media Unit has been consumed, but may not indicate an NVM failure. The value is allowed to exceed 100. Percentages greater than 254 shall be represented as 255. This value shall be updated once per power-on hour when the controller is not in a sleep state. Refer to the JEDEC JESD218B-02 standard for SSD device life and endurance measurement techniques. The relationship between this value and the value in the Percentage Used field in the Endurance Group Information log page is outside the scope of this specification.
12	<b>Number of Channels (MUCS):</b> This field indicates the number of Channels attached to this Media Unit. If this field is cleared to 0h, then no Channel Identifiers are reported for this Media Unit.
13	<b>Channel Identifiers Offset (CIO):</b> This field indicates the offset of the Channel 0 Identifier field from the beginning of the Media Unit Status Descriptor. This field shall be a non-zero value and a multiple of 16.
CIO-1:14	Reserved

**Figure 255: Media Unit Status Descriptor**

Bytes	Description
<b>Channel Identifier List</b>	
CIO+1 : CIO	<b>Channel 0 Identifier:</b> This field contains the identifier for the first Channel attached to this Media Unit, if any.
CIO+3 : CIO+2	<b>Channel 1 Identifier:</b> This field contains the identifier for the second Channel attached to this Media Unit, if reported, if any.
...	...
(MUCS*2)+CIO-1 : (MUCS*2)+CIO-2	<b>Channel MUCS-1 Identifier:</b> This field contains the identifier for the last Channel attached to this Media Unit, if any.

**5.1.12.1.17 Supported Capacity Configuration List (Log Page Identifier 11h)**

This log page is used to provide a list of Supported Capacity Configuration Descriptors (refer to Figure 256). Each entry in the list defines a different configuration of Endurance Groups supported by the domain specified in the Log Specific Identifier field in Command Dword 11 of the Get Log Page command as defined in Figure 253.

If the NVM subsystem supports multiple domains, then the controller reports the Supported Capacity Configuration List log page for the domain specified in the Log Specific Identifier field (refer to Figure 198), if accessible. If the information is not accessible, then the log page is not available (refer to section 8.1.1.3). If the Log Specific Identifier field is cleared to 0h, then the specified domain is the domain containing the controller that is processing the command.

If the NVM subsystem supports multiple domains, then Capacity Configuration Identifier values are unique within the specified domain. If the NVM subsystem does not support multiple domains, then Capacity Configuration Identifier values are unique within the NVM subsystem.

In the Supported Capacity Configuration List (refer to Figure 256), Capacity Configuration Descriptors shall be listed in ascending order by Capacity Configuration Identifier, and each Capacity Configuration Identifier shall appear only once.

**Figure 256: Supported Capacity Configuration List Log Page**

Bytes	Description
<b>Header</b>	
0	<b>Number of Supported Capacity Configurations (SCCN):</b> This field indicates the number of Capacity Configuration Descriptors in the list. If this field is cleared to 0h, then no Capacity Configuration Descriptors are reported.
15:1	Reserved
<b>Capacity Configuration Descriptor List</b>	
NOTE 1	<b>Capacity Configuration 0 Descriptor:</b> This field indicates the first Capacity Configuration Descriptor (refer to Figure 257) in the list, if any.
NOTE 1	<b>Capacity Configuration 1 Descriptor:</b> This field indicates the second Capacity Configuration Descriptor in the list, if any.
...	...
NOTE 1	<b>Capacity Configuration SCCN-1 Descriptor:</b> This field indicates the last Capacity Configuration Descriptor in the list, if any.
Notes:	
1. Capacity Configuration Descriptors may be different lengths.	

The Capacity Configuration Descriptor (refer to Figure 257) indicates the details of a particular configuration of Endurance Groups and contains one Endurance Group Configuration Descriptor for each Endurance Group accessible by the controller processing the command.

In the Capacity Configuration Descriptor (refer to Figure 257), Endurance Group Configuration Descriptors shall be listed in ascending order by Endurance Group Identifier, and each Endurance Group Identifier shall appear only once.

**Figure 257: Capacity Configuration Descriptor**

Bytes	Description
<b>Header</b>	
1:0	<b>Capacity Configuration Identifier (CCID):</b> This field indicates the identifier for this Capacity Configuration.
3:2	<b>Domain Identifier (DID):</b> This field indicates the identifier of the domain (refer to section 3.2.5.3) containing the Endurance Group configurations described by this Capacity Configuration Descriptor. A value of 0h indicates that a Domain Identifier is not reported. If multiple domains are not supported, then this field shall be cleared to 0h.
5:4	<b>Number of Endurance Group Configuration Descriptors (EGCN):</b> This field indicates the number of Endurance Group Configuration Descriptors in the list. If this field is cleared to 0h, then no Endurance Group Configuration Descriptors are reported.
31:6	Reserved
<b>Endurance Group Configuration Descriptor List</b>	
NOTE 1	<b>Endurance Group Configuration 0 Descriptor:</b> This field indicates the first Endurance Group Configuration Descriptor (refer to Figure 258) in the list, if any.
NOTE 1	<b>Endurance Group Configuration 1 Descriptor:</b> This field indicates the second Endurance Group Configuration Descriptor in the list, if any.
...	
NOTE 1	<b>Endurance Group Configuration EGCN-1 Descriptor:</b> This field indicates the last Endurance Group Configuration Descriptor in the list, if any.
Notes:	
1. Endurance Group Configuration Descriptors may be different lengths.	

The Endurance Group Configuration Descriptor is defined in Figure 258.

In the Endurance Group Configuration Descriptor (refer to Figure 258), NVM Set Identifiers shall be listed in ascending order by value, and each NVM Set Identifier shall appear only once.

In the Endurance Group Configuration Descriptor, Channel Configuration Descriptors shall be listed in ascending order by Channel Identifier value, and each Channel Identifier shall appear only once.

**Figure 258: Endurance Group Configuration Descriptor**

Bytes	Description
<b>Header</b>	
1:0	<b>Endurance Group Identifier (ENDGID):</b> This field indicates the identifier of the Endurance Group (refer to section 3.2.3) described by this Endurance Group Configuration Descriptor. This field shall indicate a value greater than or equal to 1h and less than or equal to the value of the Endurance Group Identifier Maximum field (refer to Figure 312).
3:2	<b>Capacity Adjustment Factor (CADJF):</b> This field indicates the capacity adjustment factor (refer to section 8.1.4.1) for this Endurance Group. A value of FFFFh indicates that value and all higher values. A value of 0h indicates that the Capacity Adjustment Factor is not reported.
15:4	Reserved
31:16	<b>Total Endurance Group Capacity (TEGCAP):</b> This field indicates the total NVM capacity in this Endurance Group. The value is in bytes. If this field is cleared to 0h, then the NVM subsystem does not report the total NVM capacity in this Endurance Group.
47:32	<b>Spare Endurance Group Capacity (SEGCAP):</b> This field indicates the spare NVM capacity in this Endurance Group. The value is in bytes. If this field is cleared to 0h, then the NVM subsystem does not report the unallocated NVM capacity in this Endurance Group.

**Figure 258: Endurance Group Configuration Descriptor**

Bytes	Description
63:48	<p><b>Endurance Estimate (EE):</b> This field is an estimate of the total number of data bytes that may be written to the Endurance Group over the lifetime of the Endurance Group assuming a write amplification of 1 (i.e., no increase in the number of write operations performed by the device beyond the number of write operations requested by a host). This value is reported in billions (i.e., a value of 1h corresponds to 1,000,000,000 bytes written) and is rounded up (e.g., a value of 1h indicates the number of bytes written is from 1 to 1,000,000,000, 2h indicates the number of bytes written is from 1,000,000,001 to 2,000,000,000).</p> <p>A value of FFFFFFFF_FFFFFFFF_FFFFFFFF_FFFFFFFFh means that value and all higher values.</p> <p>A value of 0h indicates that the Endurance Estimate is not reported.</p> <p>The relationship between this value and the value in the Endurance Estimate field in the Endurance Group Information log page (refer to section 5.1.12.1.10) is outside the scope of this specification.</p>
79:64	Reserved
<b>NVM Set Information</b>	
81:80	<p><b>Number of NVM Sets (EGSETS):</b> This field indicates the number of NVM Set Identifiers in this Endurance Group Configuration Descriptor. A value of 0h indicates that no NVM Set Identifiers are reported for this Endurance Group.</p>
<b>NVM Set Identifier List</b>	
83:82	<p><b>NVM Set 0 Identifier:</b> This field indicates the identifier of the first NVM Set assigned to this Endurance Group, if reported. Refer to section 3.2.2.</p>
85:84	<p><b>NVM Set 1 Identifier:</b> This field indicates the identifier of the second NVM Set assigned to this Endurance Group, if reported.</p>
...	...
(EGSETS*2)+81 : (EGSETS*2)+80	<p><b>NVM Set EGSETS-1 Identifier:</b> This field indicates the identifier of the last NVM Set assigned to this Endurance Group, if reported.</p>
<b>Channel Configuration Information</b>	
(EGSETS*2)+83 : (EGSETS*2)+82	<p><b>Number of Channels (EGCHANS):</b> This field indicates the number of Channel Configuration Descriptors in this Endurance Group Configuration Descriptor. If this field is cleared to 0h, then no Channel Configuration Descriptors are reported for this Endurance Group.</p>
<b>Channel Configuration Descriptor List</b>	
NOTE 1	<p><b>Channel 0 Configuration Descriptor:</b> This field contains the Channel Configuration Descriptor (refer to Figure 259) for the first Channel in this Endurance Group, if any.</p>
NOTE 1	<p><b>Channel 1 Configuration Descriptor:</b> This field contains the Channel Configuration Descriptor for the second Channel in this Endurance Group, if any.</p>
...	...
NOTE 1	<p><b>Channel EGCHANS-1 Configuration Descriptor:</b> This field contains the Channel Configuration Descriptor for the last Channel in this Endurance Group, if any.</p>
Notes:	
1. Channel Configuration Descriptors may be different lengths.	

The Channel Configuration Descriptor (refer to Figure 259) lists the Media Units attached to a Channel. Media Unit Configuration Descriptors (refer to Figure 260) shall be listed in ascending order by Media Unit Identifier, and each Media Unit Identifier shall appear only once.

**Figure 259: Channel Configuration Descriptor**

Bytes	Description
<b>Header</b>	
1:0	<p><b>Channel Identifier (CHID):</b> This field indicates the identifier of this Channel. A value of FFFFh indicates that the Channel Identifier is not specified.</p>
3:2	<p><b>Number of Channel Media Units (CHMUS):</b> This field indicates the number of Media Units that are attached to this Channel. If this field is cleared to 0h, then no Media Unit Configuration Descriptors are reported for this Channel.</p>

**Figure 259: Channel Configuration Descriptor**

Bytes	Description
<b>Media Unit Configuration Descriptor List</b>	
NOTE 1	<b>Media Unit 0 Configuration Descriptor:</b> This field contains the Media Unit Configuration Descriptor (refer to Figure 260) for the first Media Unit attached to this Channel, if any.
NOTE 1	<b>Media Unit 1 Configuration Descriptor:</b> This field contains the Media Unit Configuration Descriptor for the second Media Unit attached to this Channel, if reported.
...	...
NOTE 1	<b>Media Unit CHMUS-1 Configuration Descriptor:</b> This field contains the Media Unit Configuration Descriptor for the last Media Unit attached to this Channel, if any.
Notes:	
1. Media Unit Configuration Descriptors may be different lengths.	

The Media Unit Configuration Descriptor is defined in Figure 260.

**Figure 260: Media Unit Configuration Descriptor**

Bytes	Description
1:0	<b>Media Unit Identifier (MUID):</b> This field indicates the identifier of this Media Unit.
5:2	Reserved
7:6	<b>Media Unit Descriptor Length (MUDL):</b> This field contains the length in bytes of the descriptor information that follows. The total length of the Media Unit Configuration Descriptor in bytes is the value in this field plus 8.  This field shall be cleared to 0h.

#### 5.1.12.1.18 Feature Identifiers Supported and Effects (Log Page Identifier 12h)

An NVM subsystem may support several interfaces for submitting a Get Log Page command such as an Admin Submission Queue, PCIe VDM Management Endpoint, or 2-Wire Management Endpoint (refer the NVM Express Management Interface Specification for details on Management Endpoints) and may have zero or more instances of each of those interfaces. The feature identifiers (FIDs) supported on each instance of each interface may be different. This log page describes the FIDs that are supported on the interface to which the Get Log Page command was submitted and the effects of those features on the state of the NVM subsystem. The log page is defined in Figure 261. Each Feature Identifier's effects are described in a FID Supported and Effects data structure defined in Figure 262.

If the UUID Selection Supported bit is set to '1' for the Get Log Page command in the Commands Supported and Effects log page (refer to section 5.1.12.1.6), then the log page data reflects the FIDs that are supported based on the value of the UUID Index field (refer to section 8.1.28).

If the Feature Identifiers Supported and Effects log page is supported and the Feature is supported, then the scope, as defined in Figure 385, shall be indicated in the FID Scope field (FSP) for that Feature (refer to Figure 262).

For controllers that implement I/O Queues, the features that the controller supports are dependent on the I/O Command Set that is based on:

- the I/O Command Set selected in CC.CSS, if CC.CSS is not set to 110b; and
- the Command Set Identifier (CSI) field in CDW 14, if CC.CSS is set to 110b.

If CC.CSS is set to 110b, I/O Command Sets that have not been enabled by the I/O Command Set Profile (FID 19h) (refer to section 5.1.25.1.17) are treated as unsupported.

**Figure 261: Feature Identifiers Effects Log Page**

Bytes	Description
03:00	<b>Feature Identifier Supported 0 (FIS0):</b> Contains the FID Supported and Effects data structure (refer to Figure 262) for FID 0h.

**Figure 261: Feature Identifiers Effects Log Page**

Bytes	Description
07:04	<b>Feature Identifier Supported 1 (FIS1):</b> Contains the FID Supported and Effects data structure (refer to Figure 262) for FID 1h.
...	...
1019:1016	<b>Feature Identifier Supported 254 (FIS254):</b> Contains the FID Supported and Effects data structure (refer to Figure 262) for FID FEh.
1023:1020	<b>Feature Identifier Supported 255 (FIS255):</b> Contains the FID Supported and Effects data structure (refer to Figure 262) for FID FFh.

The FID Supported and Effects data structure describes the effect of a Set Features command for the FID, including any optional features of the FID.

**Figure 262: FID Supported and Effects Data Structure**

Bits	Description																		
31:20	<b>FID Scope (FSP):</b> This field defines the scope for the associated feature identifier. If the value of this field is 0h, then no scope is reported. If this field is non-zero, then only one bit shall be set to '1'.																		
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>11:7</td> <td>Reserved</td> </tr> <tr> <td>6</td> <td><b>Controller Data Queue (CDQSCP):</b> If set to '1', then modifying the attributes of the feature identified by this FID may impact a Controller Data Queue (refer to 8.1.6). If cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact a Controller Data Queue.</td> </tr> <tr> <td>5</td> <td><b>NVM Subsystem Scope (NSSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact the whole NVM subsystem. If this bit is cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact the whole NVM subsystem.</td> </tr> <tr> <td>4</td> <td><b>Domain Scope (DSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact a single Domain. If this bit is cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact a single Domain.</td> </tr> <tr> <td>3</td> <td><b>Endurance Group Scope (EGSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact Endurance Groups. If this bit is cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact Endurance Groups.</td> </tr> <tr> <td>2</td> <td><b>NVM Set Scope (NSSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact NVM Sets. If this bit is cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact NVM Sets.</td> </tr> <tr> <td>1</td> <td><b>Controller Scope (CSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact the controller. If this bit is cleared to '0' and the FSP field is non-zero, then the feature identified by this FID does not have controller scope.</td> </tr> <tr> <td>0</td> <td><b>Namespace Scope (NSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact namespaces. If this bit is cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact namespaces.</td> </tr> </tbody> </table>	Bits	Description	11:7	Reserved	6	<b>Controller Data Queue (CDQSCP):</b> If set to '1', then modifying the attributes of the feature identified by this FID may impact a Controller Data Queue (refer to 8.1.6). If cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact a Controller Data Queue.	5	<b>NVM Subsystem Scope (NSSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact the whole NVM subsystem. If this bit is cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact the whole NVM subsystem.	4	<b>Domain Scope (DSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact a single Domain. If this bit is cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact a single Domain.	3	<b>Endurance Group Scope (EGSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact Endurance Groups. If this bit is cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact Endurance Groups.	2	<b>NVM Set Scope (NSSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact NVM Sets. If this bit is cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact NVM Sets.	1	<b>Controller Scope (CSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact the controller. If this bit is cleared to '0' and the FSP field is non-zero, then the feature identified by this FID does not have controller scope.	0	<b>Namespace Scope (NSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact namespaces. If this bit is cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact namespaces.
	Bits	Description																	
	11:7	Reserved																	
	6	<b>Controller Data Queue (CDQSCP):</b> If set to '1', then modifying the attributes of the feature identified by this FID may impact a Controller Data Queue (refer to 8.1.6). If cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact a Controller Data Queue.																	
	5	<b>NVM Subsystem Scope (NSSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact the whole NVM subsystem. If this bit is cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact the whole NVM subsystem.																	
	4	<b>Domain Scope (DSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact a single Domain. If this bit is cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact a single Domain.																	
	3	<b>Endurance Group Scope (EGSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact Endurance Groups. If this bit is cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact Endurance Groups.																	
	2	<b>NVM Set Scope (NSSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact NVM Sets. If this bit is cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact NVM Sets.																	
1	<b>Controller Scope (CSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact the controller. If this bit is cleared to '0' and the FSP field is non-zero, then the feature identified by this FID does not have controller scope.																		
0	<b>Namespace Scope (NSCPE):</b> If this bit is set to '1', then modifying the attributes of the feature identified by this FID may impact namespaces. If this bit is cleared to '0' and the FSP field is non-zero, then modifying the attributes of the feature identified by this FID does not impact namespaces.																		
19	<b>UUID Selection Supported (USS):</b> If this bit is set to '1', then the controller supports the selection of a UUID (refer to section 8.1.28) by a Get Features command or a Set Features command using this FID.  If this bit is cleared to '0', then the controller does not support the selection of a UUID by a Get Features command or a Set Features command using this FID.																		
18:05	Reserved																		

**Figure 262: FID Supported and Effects Data Structure**

Bits	Description
04	<p><b>Controller Capability Change (CCC):</b> If this bit is set to '1', then changing the attributes of the feature identified by this FID may change controller capabilities. Controller capability changes include a firmware update that changes the capabilities reported in the CAP property.</p> <p>If this bit is cleared to '0', then changing the attributes of the feature identified by this FID does not modify controller capabilities.</p>
03	<p><b>Namespace Inventory Change (NIC):</b> If this bit is set to '1', then changing the attributes of the feature identified by this FID may change the number of namespaces or capabilities for multiple namespaces. Namespace inventory changes include adding or removing namespaces.</p> <p>If this bit is cleared to '0', then changing the attributes of the feature identified by this FID does not modify the number of namespaces or capabilities for multiple namespaces.</p>
02	<p><b>Namespace Capability Change (NCC):</b> If this bit is set to '1', then changing the attributes of the feature identified by this FID may change the capabilities of a single namespace. Namespace capability changes include a logical format change.</p> <p>If this bit is cleared to '0', then changing the attributes of the feature identified by this FID does not modify any namespace capabilities for the specified namespace.</p>
01	<p><b>User Data Content Change (UDCC):</b> If this bit is set to '1', then changing the attributes of the feature identified by this FID may modify user data content in one or more namespaces.</p> <p>If this bit is cleared to '0', then changing the attributes of the feature identified by this FID does not modify user data content in any namespace.</p>
00	<p><b>FID Supported (FSUPP):</b> If this bit is set to '1', then this FID is supported by the controller.</p> <p>If this bit is cleared to '0', then this FID is not supported by the controller and all other fields in this structure shall be cleared to 0h.</p> <p>Refer to section 3.1.3.6 for the FID support requirements for each controller type.</p>

#### 5.1.12.1.19 NVMe-MI Commands Supported and Effects (Log Page Identifier 13h)

This log page describes the Management Interface Command Set commands (refer to the NVM Express Management Interface Specification) that the controller supports using the NVMe-MI Send and NVMe-MI Receive commands and the effects of those Management Interface Command Set commands on the state of the NVM subsystem. The log page is defined in Figure 263.

**Figure 263: NVMe-MI Commands Supported and Effects Log Page**

Bytes	Description
03:00	<b>Management Interface Command Supported 0 (MICS0):</b> Contains the NVMe-MI Commands Supported and Effects data structure (refer to Figure 264) for the Management Interface command with an opcode value of 0h.
07:04	<b>Management Interface Command Supported 1 (MICS1):</b> Contains the NVMe-MI Commands Supported and Effects data structure (refer to Figure 264) for the Management Interface command with an opcode value of 1h.
...	...
1019:1016	<b>Management Interface Command Supported 254 (MICS254):</b> Contains the NVMe-MI Commands Supported and Effects data structure (refer to Figure 264) for the Management Interface command with an opcode value of 254.
1023:1020	<b>Management Interface Command Supported 255 (MICS255):</b> Contains the NVMe-MI Commands Supported and Effects data structure (refer to Figure 264) for the Management Interface command with an opcode value of 255.
4095:1024	Reserved

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

Figure 264: NVMe-MI Commands Supported and Effects Data Structure

Bits	Description																
31:20	<b>Command Scope (CSP):</b> This field defines the scope for the associated NVMe-MI Send command that specifies the Management Interface command opcode for this data structure. If the value of this field is 0h, then no scope is reported. If this field is non-zero, then only one bit shall be set to '1'.																
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>11:6</td> <td>Reserved</td> </tr> <tr> <td>5</td> <td><b>NVM Subsystem Scope (NSSCPE):</b> If this bit is set to '1', then the command performs actions that may impact the whole NVM subsystem. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that do not impact the whole NVM subsystem.</td> </tr> <tr> <td>4</td> <td><b>Domain Scope (DSCPE):</b> If this bit is set to '1', then the command has Domain scope. If this bit is cleared to '0' and the CSP field is non-zero, then the command does not have Domain scope.</td> </tr> <tr> <td>3</td> <td><b>Endurance Group Scope (EGSCPE):</b> If this bit is set to '1', then the command has Endurance Group scope. If this bit is cleared to '0' and the CSP field is non-zero, then the command does not have Endurance Group scope.</td> </tr> <tr> <td>2</td> <td><b>NVM Set Scope (NSSCPE):</b> If this bit is set to '1', then the command has NVM Set scope. If this bit is cleared to '0' and the CSP field is non-zero, then the command does not have NVM Set scope.</td> </tr> <tr> <td>1</td> <td><b>Controller Scope (CSCPE):</b> If this bit is set to '1', then the command has controller scope. If this bit is cleared to '0' and the CSP field is non-zero, then the command does not have controller scope.</td> </tr> <tr> <td>0</td> <td><b>Namespace Scope (NSCPE):</b> If this bit is set to '1', then the command has namespace scope. If this bit is cleared to '0' and the CSP field is non-zero, then the command does not have namespace scope.</td> </tr> </tbody> </table>	Bits	Description	11:6	Reserved	5	<b>NVM Subsystem Scope (NSSCPE):</b> If this bit is set to '1', then the command performs actions that may impact the whole NVM subsystem. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that do not impact the whole NVM subsystem.	4	<b>Domain Scope (DSCPE):</b> If this bit is set to '1', then the command has Domain scope. If this bit is cleared to '0' and the CSP field is non-zero, then the command does not have Domain scope.	3	<b>Endurance Group Scope (EGSCPE):</b> If this bit is set to '1', then the command has Endurance Group scope. If this bit is cleared to '0' and the CSP field is non-zero, then the command does not have Endurance Group scope.	2	<b>NVM Set Scope (NSSCPE):</b> If this bit is set to '1', then the command has NVM Set scope. If this bit is cleared to '0' and the CSP field is non-zero, then the command does not have NVM Set scope.	1	<b>Controller Scope (CSCPE):</b> If this bit is set to '1', then the command has controller scope. If this bit is cleared to '0' and the CSP field is non-zero, then the command does not have controller scope.	0	<b>Namespace Scope (NSCPE):</b> If this bit is set to '1', then the command has namespace scope. If this bit is cleared to '0' and the CSP field is non-zero, then the command does not have namespace scope.
	Bits	Description															
	11:6	Reserved															
	5	<b>NVM Subsystem Scope (NSSCPE):</b> If this bit is set to '1', then the command performs actions that may impact the whole NVM subsystem. If this bit is cleared to '0' and the CSP field is non-zero, then the command performs actions that do not impact the whole NVM subsystem.															
	4	<b>Domain Scope (DSCPE):</b> If this bit is set to '1', then the command has Domain scope. If this bit is cleared to '0' and the CSP field is non-zero, then the command does not have Domain scope.															
	3	<b>Endurance Group Scope (EGSCPE):</b> If this bit is set to '1', then the command has Endurance Group scope. If this bit is cleared to '0' and the CSP field is non-zero, then the command does not have Endurance Group scope.															
	2	<b>NVM Set Scope (NSSCPE):</b> If this bit is set to '1', then the command has NVM Set scope. If this bit is cleared to '0' and the CSP field is non-zero, then the command does not have NVM Set scope.															
1	<b>Controller Scope (CSCPE):</b> If this bit is set to '1', then the command has controller scope. If this bit is cleared to '0' and the CSP field is non-zero, then the command does not have controller scope.																
0	<b>Namespace Scope (NSCPE):</b> If this bit is set to '1', then the command has namespace scope. If this bit is cleared to '0' and the CSP field is non-zero, then the command does not have namespace scope.																
19:05	Reserved																
04	<b>Controller Capability Change (CCC):</b> If this bit is set to '1', then this command may change controller capabilities. If this bit is cleared to '0', then this command does not modify controller capabilities. Controller capability changes include a firmware update that changes the capabilities reported in the CAP property.																
03	<b>Namespace Inventory Change (NIC):</b> If this bit is set to '1', then this command may change the number of namespaces or capabilities for multiple namespaces. If this bit is cleared to '0', then this command does not modify the number of namespaces or capabilities for multiple namespaces. Namespace inventory changes include adding or removing namespaces.																
02	<b>Namespace Capability Change (NCC):</b> If this bit is set to '1', then this command may change the capabilities of a single namespace. If this bit is cleared to '0', then this command does not modify any namespace capabilities for the specified namespace. Namespace capability changes include a logical format change.																
01	<b>User Data Content Change (UDCC):</b> If this bit is set to '1', then this command may modify user data content in one or more namespaces. If this bit is cleared to '0', then this command does not modify user data content in any namespace. User data content changes include a write operation.																
00	<b>Command Supported (CSUPP):</b> If this bit is set to '1', then this command is supported by the controller. If this bit is cleared to '0', then this command is not supported by the controller and all other fields in this structure shall be cleared to 0h.																

#### 5.1.12.1.20 Command and Feature Lockdown (Log Page Identifier 14h)

This log page is used to indicate which commands and Set Features Feature Identifiers are supported to be prohibited from execution using the Command and Feature Lockdown capability (refer to section 8.1.5) and which commands are currently prohibited if received on an Admin Submission Queue or received out-of-band on a Management Endpoint (refer to the NVM Express Management Interface Specification). This log page uses the Log Specific Parameter field in Command Dword 10 (refer to Figure 197) as defined in Figure 265. This log page may use the UUID Index field in the Get Log Page command to specify the scope and content of the list returned in the Command and Feature Identifier List field of this log page. The UUID Index field may be used if the Scope field is set to 2h, allowing returning of vendor specific Set Features Feature Identifier lockdown information.



**Figure 265: Command and Feature Lockdown Log Specific Parameter Field**

Bits	Description														
14	Reserved														
13:12	<b>Contents (CNTTS):</b> This field in combination with the Scope field specifies the contents of the Command and Feature Identifier List field in the log page.														
	<table border="1"> <thead> <tr> <th>Value</th> <th>Command and Feature Identifier List Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>List of command opcodes or Feature Identifiers specified by the Scope field that are able to be prohibited.</td> </tr> <tr> <td>01b</td> <td>List of command opcodes or Feature Identifiers specified by the Scope field that are currently prohibited if received on an Admin submission queue.</td> </tr> <tr> <td>10b</td> <td>List of command opcodes or Feature Identifiers specified by the Scope field that are currently prohibited if received out-of-band on a Management Endpoint.</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Command and Feature Identifier List Definition	00b	List of command opcodes or Feature Identifiers specified by the Scope field that are able to be prohibited.	01b	List of command opcodes or Feature Identifiers specified by the Scope field that are currently prohibited if received on an Admin submission queue.	10b	List of command opcodes or Feature Identifiers specified by the Scope field that are currently prohibited if received out-of-band on a Management Endpoint.	11b	Reserved				
	Value	Command and Feature Identifier List Definition													
	00b	List of command opcodes or Feature Identifiers specified by the Scope field that are able to be prohibited.													
	01b	List of command opcodes or Feature Identifiers specified by the Scope field that are currently prohibited if received on an Admin submission queue.													
10b	List of command opcodes or Feature Identifiers specified by the Scope field that are currently prohibited if received out-of-band on a Management Endpoint.														
11b	Reserved														
11:08	<b>Scope (SCP):</b> This field in combination with the Contents field specifies the contents of the Command and Feature Identifier List field in the log page.														
	<table border="1"> <thead> <tr> <th>Value</th> <th>Command and Feature Identifier List Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>List of Admin Command Set opcodes</td> </tr> <tr> <td>1h</td> <td>Reserved</td> </tr> <tr> <td>2h</td> <td>List of Feature Identifiers</td> </tr> <tr> <td>3h</td> <td>List of Management Interface Command Set opcodes (refer to the NVM Express Management Interface Specification)</td> </tr> <tr> <td>4h</td> <td>List of PCIe Command Set opcodes (refer to the NVM Express Management Interface Specification)</td> </tr> <tr> <td>5h to Fh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Command and Feature Identifier List Definition	0h	List of Admin Command Set opcodes	1h	Reserved	2h	List of Feature Identifiers	3h	List of Management Interface Command Set opcodes (refer to the NVM Express Management Interface Specification)	4h	List of PCIe Command Set opcodes (refer to the NVM Express Management Interface Specification)	5h to Fh	Reserved
	Value	Command and Feature Identifier List Definition													
	0h	List of Admin Command Set opcodes													
	1h	Reserved													
	2h	List of Feature Identifiers													
	3h	List of Management Interface Command Set opcodes (refer to the NVM Express Management Interface Specification)													
4h	List of PCIe Command Set opcodes (refer to the NVM Express Management Interface Specification)														
5h to Fh	Reserved														

If a UUID Index is specified in the Get Log Page command (refer to section 5.1.12) and the Scope field is set to 2h, then the controller should return vendor specific Set Features lockdown information based on that UUID index. If the Scope field is not set to 2h, then the controller ignores the UUID index field.

If a controller processes this command with the Contents field set to 10b and the NVM subsystem does not contain a Management Endpoint, then the command shall be aborted with a status code of Invalid Field in Command.

The log page returned is defined in Figure 266.

Figure 266: Command and Feature Lockdown Log Page

Bytes	Description																																			
0	<b>Command and Feature Identifier List Attributes (CFILA):</b> This field indicates the contents of the Command and Feature Identifier List field in the log page.																																			
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:6</td> <td>Reserved</td> </tr> <tr> <td rowspan="4">5:4</td> <td><b>Contents Selected (CS):</b> This field in combination with the Scope Selected field indicates the contents of the Command and Feature Identifier List field in the log page. The Content Selected field is specified by the contents of the Contents field in the Log Specific Parameter field of the Get Log Page command.</td> </tr> <tr> <td> <table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>List contains command opcodes or Set Features Feature Identifiers based on the Scope Selected field that are supported to be prohibited</td> </tr> <tr> <td>01b</td> <td>List contains command opcodes or Set Features Feature Identifiers based on the Scope Selected field that are currently prohibited if received on an Admin submission queue</td> </tr> <tr> <td>10b</td> <td>List contains command opcodes or Set Features Feature Identifiers based on the Scope field that are currently prohibited if received out-of-band on a Management Endpoint</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table> </td> </tr> <tr> <td>3:0</td> <td><b>Scope Selected (SS):</b> This field in combination with the Contents Selected field indicates what the Command and Feature Identifier List field contains in the log page. The Scope Selected field is specified by the contents of the Scope field in the Log Specific Parameter field of the Get Log Page command.</td> </tr> <tr> <td></td> <td> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>List contains Admin Command Set opcodes</td> </tr> <tr> <td>1h</td> <td>Reserved</td> </tr> <tr> <td>2h</td> <td>List contains Set Features Feature Identifiers</td> </tr> <tr> <td>3h</td> <td>List contains Management Interface Command Set opcodes</td> </tr> <tr> <td>4h</td> <td>List contains PCIe Command Set opcodes</td> </tr> <tr> <td>5h to Fh</td> <td>Reserved</td> </tr> </tbody> </table> </td> </tr> </tbody> </table>	Bits	Description	7:6	Reserved	5:4	<b>Contents Selected (CS):</b> This field in combination with the Scope Selected field indicates the contents of the Command and Feature Identifier List field in the log page. The Content Selected field is specified by the contents of the Contents field in the Log Specific Parameter field of the Get Log Page command.	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>List contains command opcodes or Set Features Feature Identifiers based on the Scope Selected field that are supported to be prohibited</td> </tr> <tr> <td>01b</td> <td>List contains command opcodes or Set Features Feature Identifiers based on the Scope Selected field that are currently prohibited if received on an Admin submission queue</td> </tr> <tr> <td>10b</td> <td>List contains command opcodes or Set Features Feature Identifiers based on the Scope field that are currently prohibited if received out-of-band on a Management Endpoint</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Description	00b	List contains command opcodes or Set Features Feature Identifiers based on the Scope Selected field that are supported to be prohibited	01b	List contains command opcodes or Set Features Feature Identifiers based on the Scope Selected field that are currently prohibited if received on an Admin submission queue	10b	List contains command opcodes or Set Features Feature Identifiers based on the Scope field that are currently prohibited if received out-of-band on a Management Endpoint	11b	Reserved	3:0	<b>Scope Selected (SS):</b> This field in combination with the Contents Selected field indicates what the Command and Feature Identifier List field contains in the log page. The Scope Selected field is specified by the contents of the Scope field in the Log Specific Parameter field of the Get Log Page command.		<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>List contains Admin Command Set opcodes</td> </tr> <tr> <td>1h</td> <td>Reserved</td> </tr> <tr> <td>2h</td> <td>List contains Set Features Feature Identifiers</td> </tr> <tr> <td>3h</td> <td>List contains Management Interface Command Set opcodes</td> </tr> <tr> <td>4h</td> <td>List contains PCIe Command Set opcodes</td> </tr> <tr> <td>5h to Fh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	List contains Admin Command Set opcodes	1h	Reserved	2h	List contains Set Features Feature Identifiers	3h	List contains Management Interface Command Set opcodes	4h	List contains PCIe Command Set opcodes	5h to Fh	Reserved
	Bits	Description																																		
	7:6	Reserved																																		
	5:4	<b>Contents Selected (CS):</b> This field in combination with the Scope Selected field indicates the contents of the Command and Feature Identifier List field in the log page. The Content Selected field is specified by the contents of the Contents field in the Log Specific Parameter field of the Get Log Page command.																																		
<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>List contains command opcodes or Set Features Feature Identifiers based on the Scope Selected field that are supported to be prohibited</td> </tr> <tr> <td>01b</td> <td>List contains command opcodes or Set Features Feature Identifiers based on the Scope Selected field that are currently prohibited if received on an Admin submission queue</td> </tr> <tr> <td>10b</td> <td>List contains command opcodes or Set Features Feature Identifiers based on the Scope field that are currently prohibited if received out-of-band on a Management Endpoint</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>		Value	Description	00b	List contains command opcodes or Set Features Feature Identifiers based on the Scope Selected field that are supported to be prohibited	01b	List contains command opcodes or Set Features Feature Identifiers based on the Scope Selected field that are currently prohibited if received on an Admin submission queue	10b	List contains command opcodes or Set Features Feature Identifiers based on the Scope field that are currently prohibited if received out-of-band on a Management Endpoint	11b	Reserved																									
Value		Description																																		
00b		List contains command opcodes or Set Features Feature Identifiers based on the Scope Selected field that are supported to be prohibited																																		
01b	List contains command opcodes or Set Features Feature Identifiers based on the Scope Selected field that are currently prohibited if received on an Admin submission queue																																			
10b	List contains command opcodes or Set Features Feature Identifiers based on the Scope field that are currently prohibited if received out-of-band on a Management Endpoint																																			
11b	Reserved																																			
3:0	<b>Scope Selected (SS):</b> This field in combination with the Contents Selected field indicates what the Command and Feature Identifier List field contains in the log page. The Scope Selected field is specified by the contents of the Scope field in the Log Specific Parameter field of the Get Log Page command.																																			
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>List contains Admin Command Set opcodes</td> </tr> <tr> <td>1h</td> <td>Reserved</td> </tr> <tr> <td>2h</td> <td>List contains Set Features Feature Identifiers</td> </tr> <tr> <td>3h</td> <td>List contains Management Interface Command Set opcodes</td> </tr> <tr> <td>4h</td> <td>List contains PCIe Command Set opcodes</td> </tr> <tr> <td>5h to Fh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	List contains Admin Command Set opcodes	1h	Reserved	2h	List contains Set Features Feature Identifiers	3h	List contains Management Interface Command Set opcodes	4h	List contains PCIe Command Set opcodes	5h to Fh	Reserved																					
Value	Definition																																			
0h	List contains Admin Command Set opcodes																																			
1h	Reserved																																			
2h	List contains Set Features Feature Identifiers																																			
3h	List contains Management Interface Command Set opcodes																																			
4h	List contains PCIe Command Set opcodes																																			
5h to Fh	Reserved																																			
2:1	Reserved																																			
3	<b>Length (LNGTH):</b> This field indicates the length in bytes (n) of the Command and Feature Identifier List field that follow in the log page. If the Command and Feature Identifier List field contains no coded values, then this field shall be cleared to 0h.																																			
n+3:4	<b>Command and Feature Identifier List (CFIL):</b> The contents of this field are dependent on the setting of the Contents Selected field and Scope Selected field. This field contains a list of coded values identified by the Scope Selected field and the Content Selected field. The list shall be in order from lowest numerical value to highest numerical value.																																			
511:n+4	Reserved																																			

#### 5.1.12.1.21 Boot Partition (Log Page Identifier 15h)

The Boot Partition log page provides read only access to the Boot Partition (refer to section 8.1.3) accessible by this controller through the BPRSEL register (refer to section 3.1.4.14).

This log page consists of a header describing the Boot Partition and Boot Partition data as defined by Figure 268. The Boot Partition Identifier bit in the Log Specific Parameter field determines the Boot Partition.

A host reading this log page has no effects on the BPINFO (refer to section 3.1.4.13), BPRSEL, and BPMBL (refer to section 3.1.4.15) registers.

The Log Specific Parameter field in Command Dword 10 (refer to Figure 197) for this log page is defined in Figure 267.

**Figure 267: Boot Partition Log Specific Parameter Field**

Bits	Description
14:09	Reserved
08	<b>Boot Partition Identifier (BPID):</b> This bit specifies the Boot Partition identifier for the Boot Partition to return.

**Figure 268: Boot Partition Log Page**

Bytes	Description								
<b>Boot Partition Header</b>									
00	<b>Log Page Identifier (LID):</b> This field shall be set to 15h.								
03:01	Reserved								
07:04	<b>Boot Partition Information (BPINFO):</b> Contains defines the characteristics of Boot Partitions.								
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>31</td> <td><b>Active Boot Partition ID (ABPID):</b> This bit indicates the identifier of the active Boot Partition.</td> </tr> <tr> <td>30:15</td> <td>Reserved</td> </tr> <tr> <td>14:00</td> <td><b>Boot Partition Size (BPSZ):</b> This field defines the size of the Boot Partition Data field in multiples of 128 KiB.</td> </tr> </tbody> </table>	Bits	Description	31	<b>Active Boot Partition ID (ABPID):</b> This bit indicates the identifier of the active Boot Partition.	30:15	Reserved	14:00	<b>Boot Partition Size (BPSZ):</b> This field defines the size of the Boot Partition Data field in multiples of 128 KiB.
	Bits	Description							
	31	<b>Active Boot Partition ID (ABPID):</b> This bit indicates the identifier of the active Boot Partition.							
30:15	Reserved								
14:00	<b>Boot Partition Size (BPSZ):</b> This field defines the size of the Boot Partition Data field in multiples of 128 KiB.								
15:08	Reserved								
<b>Boot Partition Data</b>									
BPSZ*128 KiB + 15:16	<b>Boot Partition Data (BPD):</b> Contains the contents of the specified Boot Partition.								

#### 5.1.12.1.22 Rotational Media Information Log (Log Page Identifier 16h)

This log page provides rotational media information (refer to section 8.1.23) for Endurance Groups that store data on rotational media. The information provided is retained across power cycles and resets.

The Endurance Group Identifier is specified in the Log Specific Identifier field in Command Dword 11 of the Get Log Page command as defined in Figure 217.

If the NVM subsystem does not contain any Endurance Groups that store data on rotational media, then the Rotational Media Information Log should not be supported.

**Figure 269: Rotational Media Information Log Page**

Bytes	Description										
1:0	<b>Endurance Group Identifier (ENDGID):</b> The Endurance Group Identifier specified by the Get Log Page command.										
3:2	<b>Number of Actuators (NUMA):</b> Contains the number of actuators in this Endurance Group.										
5:4	<b>Nominal Rotational Speed (NRS):</b>										
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0000h</td> <td>Not reported</td> </tr> <tr> <td>0001h</td> <td>This value is prohibited to maintain backward compatibility with other standards. This value shall not be used.</td> </tr> <tr> <td>FFFFh</td> <td>Reserved</td> </tr> <tr> <td>All other values</td> <td>Nominal rotational speed in revolutions per minute while the current Power State is 0 (refer to section 5.1.25.1.2).</td> </tr> </tbody> </table>	Value	Definition	0000h	Not reported	0001h	This value is prohibited to maintain backward compatibility with other standards. This value shall not be used.	FFFFh	Reserved	All other values	Nominal rotational speed in revolutions per minute while the current Power State is 0 (refer to section 5.1.25.1.2).
	Value	Definition									
	0000h	Not reported									
0001h	This value is prohibited to maintain backward compatibility with other standards. This value shall not be used.										
FFFFh	Reserved										
All other values	Nominal rotational speed in revolutions per minute while the current Power State is 0 (refer to section 5.1.25.1.2).										
7:6	Reserved										
11:8	<b>Spinup Count (SPINC):</b> Contains the total number of successful spinup events for this Endurance Group over the lifetime of the Endurance Group. If the Spinup Count is less than FFFFFFFFh, then the controller shall increment this count by one for each successful spinup event.  A successful spinup event occurs when the controller power state transitions from a non-operational power state to an operational power state.										

**Figure 269: Rotational Media Information Log Page**

Bytes	Description
15:12	<p><b>Failed Spinup Count (FSPINC):</b> Contains the total number of failed spinup events for this Endurance Group over the lifetime of the Endurance Group. If the Failed Spinup Count is less than FFFFFFFFh, then the controller shall increment this count by one for each failed spinup event.</p> <p>A failed spinup event occurs when the controller fails an attempt to transition from a non-operational power state to an operational power state.</p>
19:16	<p><b>Load Count (LDC):</b> Contains the total number successful actuator load events for this Endurance Group over the lifetime of the Endurance Group. If the Load Count is less than FFFFFFFFh, then the controller shall increment this count by one for each successful actuator load event.</p> <p>A successful actuator load event occurs if an actuator transitions from a non-operational state to an operational state.</p>
23:20	<p><b>Failed Load Count (FLDC):</b> Contains the number of failed actuator load events for this Endurance Group over the lifetime of the Endurance Group. If the Failed Load Count is less than FFFFFFFFh, then the controller shall increment this count by one for each failed actuator load event.</p> <p>A failed actuator load event occurs if an actuator fails an attempt to transition from a non-operational state to an operational state.</p>
511:24	Reserved

#### 5.1.12.1.23 Dispersed Namespace Participating NVM Subsystems (Log Page Identifier 17h)

This log page is used to provide a list of NQNs for all participating NVM subsystems which contain controllers that are able to provide access to the dispersed namespace for the specified NSID. A typical Dispersed Namespace implementation uses Fabrics based NVM subsystems.

If an NSID of 0h or FFFFFFFFh is specified by the host in the Get Log Page command, then the controller shall abort that command with a status code of Invalid Namespace or Format as described in Figure 92. If the NSID specified by the host in the Get Log Page command is associated with a namespace that is not a dispersed namespace (i.e., the Dispersed Namespace (DISNS) bit is cleared to '0' in the Namespace Multi-path I/O and Namespace Sharing Capabilities (NMIC) field of the I/O Command Set Independent Identify Namespace data structure (refer to Figure 319) or the Identify Namespace data structure associated with that namespace), then the controller shall abort that command with a status code of Invalid Field in Command as described in section 5.1.12.

The method used by the NVM subsystem containing the controller processing the command to discover the NQNs of separate participating NVM subsystems which contain controllers that are able to provide access to the same dispersed namespace is outside the scope of this specification.

The log page returned is defined in Figure 270. Each entry in the Participating NVM Subsystems List shall contain an NQN that matches the NQN contained in the NVM Subsystem NVMe Qualified Name (SUBNQN) field of that participating NVM subsystem's Identify Controller data structure (refer to Figure 312). The Participating NVM Subsystem Entry 0 field (i.e., first entry in the Participating NVM Subsystems List) shall contain the NQN of the NVM subsystem containing the controller processing the command.

**Figure 270: Dispersed Namespace Participating NVM Subsystems Log Page**

Bytes	Description
<b>Header</b>	
07:00	<p><b>Generation Counter (GENCTR):</b> This field contains a value that is incremented each time the participating NVM subsystem information for the specified NSID changes (i.e., each time participating NVM subsystems are added or removed for the specified NSID) since the last time the host read this log page. If the value of this field is FFFFFFFF_FFFFFFFFh, then the field shall be cleared to 0h when incremented (i.e., rolls over to 0h).</p>
15:08	<p><b>Number of Participating NVM Subsystems (NUMPSUB):</b> This field indicates the number of participating NVM subsystems contained in the log page.</p>
255:16	Reserved
<b>Participating NVM Subsystems List</b>	

**Figure 270: Dispersed Namespace Participating NVM Subsystems Log Page**

Bytes	Description
511:256	<b>Participating NVM Subsystem Entry 0:</b> This field contains the NQN of the NVM subsystem containing the controller processing the command.
767:512	<b>Participating NVM Subsystem Entry 1:</b> This field contains the NQN of a participating NVM subsystem that contains controllers that are able to provide access to the dispersed namespace.
...	...
$((N + 2) * 256) - 1$ : $(N + 1) * 256$	<b>Participating NVM Subsystem Entry N:</b> This field contains the NQN of a participating NVM subsystem that contains controllers that are able to provide access to the dispersed namespace.

**5.1.12.1.24 Management Address List (Log Page Identifier 18h)**

This log consists of a Management Address List (refer to Figure 271) of up to eight Management Address Descriptors (refer to Figure 272). Management addresses are described in section 8.1.14.

**Figure 271: Management Address List – Log Page**

Bytes	Description
511:00	<b>Management Address Descriptor 0 (MAD0):</b> This field contains the first Management Address Descriptor (refer to Figure 272), if any. If the Management Address Type field is set to FFh, then subsequent Management Address Descriptors are reserved.
1023:512	<b>Management Address Descriptor 1 (MAD1):</b> This field contains the second Management Address Descriptor (refer to Figure 272), if any. If the Management Address Type field is set to FFh, then subsequent Management Address Descriptors are reserved.
...	...
4095:3584	<b>Management Address Descriptor 7 (MAD7):</b> This field contains the last Management Address Descriptor (refer to Figure 272), if any.

Figure 272 describes the Management Address Descriptor.

**Figure 272: Management Address Descriptor**

Bytes	Description												
00	<b>Management Address Type (MAT):</b> This field describes the contents of the Management Address field:												
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>The Management Address field is reserved.</td> </tr> <tr> <td>1h</td> <td>The Management Address field contains the address of a management agent in the NVM subsystem (e.g., in an Ethernet-attached SSD).</td> </tr> <tr> <td>2h</td> <td>The Management Address field contains the address of a fabric interface manager (e.g., a Service URI for the Simple Network Management Protocol (SNMPv3) agent as defined in RFC 4088).</td> </tr> <tr> <td>FFh</td> <td>The Management Address field is reserved and all subsequent Management Address Descriptors, if any, are reserved.</td> </tr> <tr> <td>Other values</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	The Management Address field is reserved.	1h	The Management Address field contains the address of a management agent in the NVM subsystem (e.g., in an Ethernet-attached SSD).	2h	The Management Address field contains the address of a fabric interface manager (e.g., a Service URI for the Simple Network Management Protocol (SNMPv3) agent as defined in RFC 4088).	FFh	The Management Address field is reserved and all subsequent Management Address Descriptors, if any, are reserved.	Other values	Reserved
	Value	Definition											
	0h	The Management Address field is reserved.											
	1h	The Management Address field contains the address of a management agent in the NVM subsystem (e.g., in an Ethernet-attached SSD).											
2h	The Management Address field contains the address of a fabric interface manager (e.g., a Service URI for the Simple Network Management Protocol (SNMPv3) agent as defined in RFC 4088).												
FFh	The Management Address field is reserved and all subsequent Management Address Descriptors, if any, are reserved.												
Other values	Reserved												
03:01	Reserved												
511:04	<b>Management Address (MADRS):</b> A uniform resource indicator (URI; refer to RFC 3986) as a UTF-8 null-terminated string, containing the address of a management entity of the type indicated by the MAT field.												

Because the list is able to contain from zero to eight Management Address Descriptors with a MAT field less than FFh, a host scans the list starting with Management Address Descriptor 0, and ending with either Management Address Descriptor 7 or a Management Address Descriptor having a MAT field set to FFh.

### 5.1.12.1.25 Reachability Groups (Log Identifier 1Ah)

This log page consists of a header describing the log page and a list of descriptors containing the Reachability Groups (refer to section 8.1.19) that contain namespaces attached to the controller processing this command that all have the same reachability attributes.

If the Index Offset Supported bit is cleared to '0' in the LID Support and Effects data structure for this log page (refer to Figure 204), then:

- if the RGO bit is cleared to '0' in Command Dword 10, then the LPOL field in Command Dword 12 and the LPOU field in Command Dword 13 of the Get Log Page command should be cleared to 0h.

If the Index Offset Supported bit is set to '1' in the LID Supported and Effects data structure for this log page (refer to Figure 204), then:

- for indexes greater than 0, the entry data structure that is indexed is a Reachability Group descriptor (e.g., specifying an index offset of 2 returns this log page starting at the second descriptor (i.e., Reachability Group Descriptor 1)); and
- for index 0, the data structure starting from the beginning of the log page is returned.

If the host performs multiple Get Log Page commands to read this log page (e.g., using the LPOL field or the LPOU field), then the host should re-read the header of the log page and ensure that the Change Count field in the log page matches the original value read. If it does not match, then the data captured is not consistent and this log page should be re-read.

The Log Specific Parameter field in Command Dword 10 (refer to Figure 197) for this log page is defined in Figure 273.

**Figure 273: Reachability Groups Log Specific Parameter Field**

Bits	Description
14:09	Reserved
08	<b>Return Groups Only (RGO):</b> If this bit is set to '1', then the controller shall return Reachability Group Descriptors with the Number of NSID Values field in each Reachability Group Descriptor cleared to 0h (i.e., no Namespace Identifiers are returned). If this bit is cleared to '0', then the controller shall return Reachability Group Descriptors that contain the Namespace Identifiers of attached namespaces that are members of the Reachability Group described by that Reachability Group Descriptor and the Number of NSID Values field set to the number of Namespace Identifier values in that Reachability Group Descriptor.

**Figure 274: Reachability Groups Log Page**

Bytes	Description
<b>Header</b>	
07:00	<b>Change Count (CHNGC):</b> This field contains a 64-bit incrementing Reachability Groups log page change count, indicating an identifier for this set of reachability groups information. The count starts at 0h following a Controller Level Reset and is incremented each time the contents of the log page change. If the value of this field is FFFFFFFF_FFFFFFFFh, then the field shall be set to 0h when incremented (i.e., rolls over to 0h).
09:08	<b>Number of Reachability Group Descriptors (NRGD):</b> This field indicates the number of Reachability Group Descriptors available in the log page. The log page shall contain one Reachability Group Descriptor for each Reachability Group that contains namespaces that are attached to the controller.  If, for a Reachability Group, there are no namespaces attached to the controller processing the command, then no Reachability Group Descriptor is returned for that Reachability Group (i.e., a Reachability Group Descriptor is returned only if that Reachability Group contains namespaces that are attached to the controller processing the command).  If no namespaces are attached to the controller, then the log page does not contain any Reachability Group Descriptors and this field is cleared to 0h.
15:10	Reserved

**Figure 274: Reachability Groups Log Page**

Bytes	Description
<b>Reachability Group Descriptor List</b>	
n:16	<b>Reachability Group Descriptor 0:</b> The first Reachability Group Descriptor (refer to Figure 275), if any.
m:n+1	<b>Reachability Group Descriptor 1:</b> The second Reachability Group Descriptor (refer to Figure 275), if any.
...	...
x:y	<b>Reachability Group Descriptor NRGD-1:</b> The last Reachability Group Descriptor (refer to Figure 275), if any.

The format of the Reachability Group Descriptor is defined in Figure 275. Namespace Identifiers shall be listed in ascending NSID order.

**Figure 275: Reachability Group Descriptor format**

Bytes	Description
<b>Header</b>	
03:00	<b>Reachability Group ID (RGID):</b> The Reachability Group Identifier associated with namespaces in the Reachability Group (refer to section 8.1.19) described by this Reachability Group Descriptor.
07:04	<b>Number of NSID Values (NNID):</b> This field indicates the number of Namespace Identifier values in this Reachability Group Descriptor. If the RGO bit is set to '1', then this field shall be cleared to 0h.
15:08	<b>Change Count (CHNGC):</b> This field contains a 64-bit incrementing count, indicating an identifier for the information contained in this Reachability Group Descriptor. A value of 0h indicates that the controller does not report a Change Count for this Reachability Group Descriptor. If a Change Count is reported, then the count starts at 1h following a Controller Level Reset and is incremented each time the data in this Reachability Group Descriptor changes. If the value of this field is FFFFFFFF_FFFFFFFFh, then the field shall be set to 1h when incremented (i.e., rolls over to 1h). If this field contains 0h, the host should examine this Reachability Group Descriptor for any changes and not use this field as an indicator that a change has occurred.
31:16	Reserved
<b>Namespace Identifier List</b>	
35:32	<b>Namespace Identifier 0:</b> The Namespace Identifier of the first attached namespace, if any, that is a member of this Reachability Group.
39:36	<b>Namespace Identifier 1:</b> The Namespace Identifier of the second attached namespace, if any, that is a member of this Reachability Group.
...	...
(((NNID-1)*4) + 35): (((NNID-1)*4) + 32)	<b>Namespace Identifier NNID-1:</b> The Namespace Identifier of the NNID attached namespace, if any, that is a member of this Reachability Group.

#### 5.1.12.1.26 Reachability Associations (Log Identifier 1Bh)

This log page consists of a header describing the log page and a list of descriptors containing the Reachability Associations (refer to section 8.1.19), for the controller processing the Get Log Page command requesting the Reachability Associations log page, that contain Reachability Groups (refer 5.1.12.1.25) that all have the same reachability attributes.

If the Index Offset Supported bit is cleared to '0' in the LID Support and Effects data structure for this log page (refer to Figure 204), then:

- if the RAO bit is cleared to '0' in Command Dword 10, then the LPOL field in Command Dword 12 and the LPOU field in Command Dword 13 of the Get Log Page command should be cleared to 0h.

If the Index Offset Supported bit is set to '1' in the LID Supported and Effects data structure for this log page (refer to Figure 204), then:

- for indexes greater than 0, the entry data structure that is indexed is a Reachability Association descriptor (e.g., specifying an index offset of 2 returns this log page starting at the second descriptor (i.e., Reachability Association Descriptor 1)).

If the host performs multiple Get Log Page commands to read this log page (e.g., using the LPOL field or the LPOU field), then the host should re-read the header of the log page and ensure that the Change Count field in this log page matches the original value read. If it does not match, then the data captured is not consistent and this log page should be re-read.

The Log Specific Parameter field in Command Dword 10 (refer to Figure 197) for this log page is defined in Figure 276.

**Figure 276: Reachability Associations Log Specific Parameter Field**

Bits	Description
14:09	Reserved
08	<b>Return Associations Only (RAO):</b> If this bit is set to '1', then the controller shall return Reachability Association Descriptors with the Number of RGID Values field in each Reachability Association Descriptor cleared to 0h (i.e., no Reachability Group Identifiers are returned). If this bit is cleared to '0', then the controller shall return Reachability Association Descriptors that contain the RGIDs of reachability groups that are members of the Reachability Association described by that Reachability Association Descriptor and the Number of RGID Values field set to the number of Reachability Group Identifier values in that Reachability Association Descriptor.

**Figure 277: Reachability Associations Log Page**

Bytes	Description
<b>Header</b>	
07:00	<b>Change Count (CHNGC):</b> This field contains a 64-bit incrementing Reachability Associations log change count, indicating an identifier for this set of reachability associations information. The count starts at 0h following a Controller Level Reset and is incremented each time the contents of the log page change. If the value of this field is FFFFFFFF_FFFFFFFFh, then the field shall be set to 0h when incremented (i.e., rolls over to 0h).
09:08	<b>Number of Reachability Association Descriptors (NRAD):</b> This field indicates the number of Reachability Association Descriptors available in the log page. The log page shall contain one Reachability Association Descriptor for each Reachability Association that contains a Reachability Group that contains namespaces that are attached to the controller.  If, for a Reachability Association, there are no Reachability Groups that contains namespaces that are attached to the controller processing the command, then no Reachability Association Descriptor shall be returned for that Reachability Association (i.e., a Reachability Association Descriptor is returned only if that Reachability Association contains Reachability Groups associated with the controller processing the command).  If no namespaces are attached to the controller, then the log page does not contain any Reachability Association Descriptors and this field shall be cleared to 0h.
15:10	Reserved
<b>Reachability Association Descriptor List</b>	
n:16	<b>Reachability Association Descriptor 0:</b> This is the first Reachability Association Descriptor (Figure 278), if any.
m:n+1	<b>Reachability Association Descriptor 1:</b> This is the second Reachability Association Descriptor (Figure 278), if any.
...	...
x:y	<b>Reachability Association Descriptor NRAD-1:</b> This is the last Reachability Association Descriptor (Figure 278), if any.

The format of the Reachability Association Descriptor is defined in Figure 278. Reachability Group Identifiers shall be listed in ascending order.



**Figure 278: Reachability Association Descriptor format**

Bytes	Description															
<b>Header</b>																
03:00	<b>Reachability Association ID (RASID):</b> The Reachability Association Identifier associated with Reachability Groups in the Reachability Association (refer to section 8.1.19) described by this Reachability Association Descriptor.															
07:04	<b>Number of RGID Values (NRID):</b> This field indicates the number of Reachability Group Identifier values in this Reachability Association Descriptor.  If the RAO bit is set to '1', then this field shall be cleared to 0h.															
15:08	<b>Change Count (CHNGC):</b> This field contains a 64-bit incrementing count, indicating an identifier for the information contained in this Reachability Association Descriptor. A value of 0h indicates that the controller does not report a Change Count for this Reachability Association Descriptor. If a Change Count is reported, then the count starts at 1h following a Controller Level Reset and is incremented each time the data in this Reachability Association Descriptor changes. If the value of this field is FFFFFFFF_FFFFFFFFh, then the field shall be set to 1h when incremented (i.e., rolls over to 1h).  If this field contains 0h, the host should examine this Reachability Group Descriptor for any changes and not use this field as an indicator that a change has occurred.															
16	<b>Reachability Association Characteristics (RAC):</b> This field indicates the Access characteristics for all Reachability Groups in this Reachability Association Descriptor for operations between the namespaces represented by the Reachability Groups.															
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> <th>Reference</th> </tr> </thead> <tbody> <tr> <td>01h</td> <td>Reachable with no performance characteristics specified</td> <td>8.1.19</td> </tr> <tr> <td>02h</td> <td>Fast copy operations are supported</td> <td>The Fast copy operations section in the NVM Command Set Specification</td> </tr> <tr> <td>03h</td> <td>Fast copy operations are not supported</td> <td>The Fast copy operations section in the NVM Command Set Specification</td> </tr> <tr> <td>All other values</td> <td colspan="2" style="text-align: center;">Reserved</td> </tr> </tbody> </table>	Value	Definition	Reference	01h	Reachable with no performance characteristics specified	8.1.19	02h	Fast copy operations are supported	The Fast copy operations section in the NVM Command Set Specification	03h	Fast copy operations are not supported	The Fast copy operations section in the NVM Command Set Specification	All other values	Reserved	
	Value	Definition	Reference													
	01h	Reachable with no performance characteristics specified	8.1.19													
	02h	Fast copy operations are supported	The Fast copy operations section in the NVM Command Set Specification													
03h	Fast copy operations are not supported	The Fast copy operations section in the NVM Command Set Specification														
All other values	Reserved															
31:17	Reserved															
<b>Reachability Group Identifier List</b>																
35:32	<b>Reachability Group Identifier 0:</b> The Reachability Group Identifier of the first Reachability Group, if any, that is a member of this Reachability Association.															
39:36	<b>Reachability Group Identifier 1:</b> The Reachability Group Identifier of the second Reachability Group, if any, that is a member of this Reachability Association.															
...	...															
(((NRID-1)*4)+35): (((NRID-1)*4)+32)	<b>Reachability Group Identifier NRID-1:</b> The Reachability Group Identifier of the last Reachability Group, if any, that is a member of this Reachability Association.															

**5.1.12.1.27 Changed Allocated Namespace List (Log Identifier 1Ch)**

This log page is used to describe changes to allocated namespaces (refer to section 1.5.6), since the last time that this log page was read, that:

- a) have changed information in their Identify Namespace data structures (refer to section 1.5.49);
- b) were previously unattached to the controller and have since been attached to the controller;
- c) were previously attached to the controller and have since been unattached from the controller;
- d) were created; and
- e) were deleted.

The log page contains a Namespace List with up to 1,024 entries. If more than 1,024 namespaces have changed attributes since the last time the log page was read, the first entry in the log page shall be set to FFFFFFFFh and the remainder of the list shall be zero filled.

### 5.1.12.1.28 Flexible Data Placement (FDP) Configurations (Log Page Identifier 20h)

The FDP Configurations log page (refer to Figure 279) identifies a list of static FDP configurations that are allowed to be applied to the specified Endurance Group. The Endurance Group Identifier in the Log Specific Identifier field as defined in Figure 217 specifies the Endurance Group. The creation of an Endurance Group or the enablement of an FDP configuration in an Endurance Group within the NVM subsystem may affect which FDP configurations are currently allowed (refer to the FDP Configuration Valid bit in Figure 280). Refer to section 8.1.10 for the usage of this log page by the host.

**Figure 279: FDP Configurations Log Page**

Bytes	Description
<b>Header</b>	
01:00	<b>Number of FDP Configurations (NUMFDPC):</b> This field indicates the number of FDP Configuration Descriptors contained in this log page. This is a 0's based number.
02	<b>Version (VER):</b> This field indicates the version of the log page that includes the header and the FDP Configuration Descriptor List. This field shall be cleared to 0h.
03	Reserved
07:04	<b>Size (SIZE):</b> This field identifies the size of this log page in bytes.
15:08	Reserved
<b>FDP Configuration Descriptor List</b>	
Variable Sized: 16	<b>FDP Configuration Descriptor 0:</b> The FDP Configuration Descriptor (refer to Figure 280) of the first configuration.
Variable Sized	<b>FDP Configuration Descriptor 1:</b> The FDP Configuration Descriptor (refer to Figure 280) of the second configuration, if any.
...	...
Variable Sized	<b>FDP Configuration Descriptor NUMFDPC-1:</b> The FDP Configuration Descriptor (refer to Figure 280) of the last configuration, if any.

Figure 280 defines the format of an FDP Configuration Descriptor. The Reclaim Unit Handles in the Reclaim Unit Handle list shall be listed in ascending order of Reclaim Unit Handle Identifier.

**Figure 280: FDP Configuration Descriptor**

Bytes	Description	
01:00	<b>Descriptor Size (DSZE):</b> This field indicates the size in bytes of this FDP Configuration Descriptor.	
02	<b>FDP Attributes (FDPA):</b> This field identifies attributes of this FDP configuration.	
	<b>Bits</b>	<b>Description</b>
	7	<b>FDP Configuration Valid (FDPCV):</b> This bit is set to '1' to indicate that this FDP configuration is valid. This bit is cleared to '0' to indicate that this entry is not currently available and the host should ignore this FDP configuration.
	6:5	Reserved
4	<b>FDP Volatile Write Cache (FDPVWC):</b> If this bit is set to '1', then a volatile write cache is present for this FDP configuration. If this bit is cleared to '0', then a volatile write cache is not present for this FDP configuration. Refer to section 5.1.25.1.4.  If the controller reports one or more FDP configurations with this bit set to '1', then the Volatile Write Cache Present (VWCP) bit in the Identify Controller data structure (refer to Figure 312) shall be set to '1'. This results in the VWCP bit being set to '1' even though there is no volatile write cache in the controller to support hosts not aware of the Flexible Data Placement capability (refer to section 8.1.10).  Refer to section 7.2 to determine if a volatile write cache is present if a namespace exists in an Endurance Group that has Flexible Data Placement enabled.	
3:0	<b>Reclaim Group Identifier Format (RGIF):</b> This field identifies the number of most-significant bits in the Placement Identifier that contain the Reclaim Group Identifier (refer to Figure 282 and Figure 283). If the NRG field in this data structure is set to 1h (i.e., there is a single Reclaim Group), then this field may be cleared to 0h. If the NRG field is greater than 1h, then this field shall be a non-zero value.	

**Figure 280: FDP Configuration Descriptor**

Bytes	Description
03	<b>Vendor Specific Size (VSS):</b> This field identifies the size in bytes of the Vendor Specific field.
07:04	<b>Number of Reclaim Groups (NRG):</b> This field indicates the number of Reclaim Groups in this FDP configuration. This field shall be a non-zero value.
09:08	<b>Number of Reclaim Unit Handles (NRUH):</b> This field identifies the number of Reclaim Unit Handle Descriptors in the Reclaim Unit Handle List. This field shall be a non-zero value.
11:10	<b>Max Placement Identifiers (MAXPIDS):</b> This field indicates the maximum value allowed in the Number of Placement Identifiers (NPID) field (refer to Figure 566) in an I/O Management Send command for the Reclaim Unit Handle Update Operation (refer to section 7.4.1.1) for this FDP configuration. This is a 0's based number.  The value of this field shall be less than the product of the NRG field and the NRUH field.
15:12	<b>Number of Namespaces Supported (NNS):</b> This field indicates the number of namespaces allowed to be created in the Endurance Group if Flexible Data Placement is enabled utilizing this FDP configuration.  If the MNAN field is set to a value greater than 0h (refer to Figure 312), then this field shall be less than or equal to the value in the MNAN field. If the MNAN field is cleared to 0h, then this field shall be less than or equal to the value in the NN field (refer to Figure 312).
23:16	<b>Reclaim Unit Nominal Size (RUNS):</b> This field indicates the nominal size, in bytes, of each Reclaim Unit of non-volatile storage within each Reclaim Group.
27:24	<b>Estimated Reclaim Unit Time Limit (ERUTL):</b> This field indicates the estimated maximum time in seconds that a Reclaim Unit is allowed to be referenced by a Reclaim Unit Handle (refer to section 3.2.4 and section 8.1.10) before the controller is allowed to modify the Reclaim Unit Handle to reference a different Reclaim Unit. If this field is cleared to 0h, then the Estimated Reclaim Unit Time Limit is not reported.
63:28	Reserved
<b>Reclaim Unit Handle Descriptor List</b>	
67:64	<b>Reclaim Unit Handle Descriptor 0:</b> This field contains the Reclaim Unit Handle Descriptor (refer to Figure 281) for Reclaim Unit Handle 0 (refer to Figure 70).
71:68	<b>Reclaim Unit Handle Descriptor 1:</b> This field contains the Reclaim Unit Handle Descriptor (refer to Figure 281) for Reclaim Unit Handle 1 (refer to Figure 70), if any.
...	...
$((NRUH-1)*4)+67$ : $((NRUH-1)*4)+64$	<b>Reclaim Unit Handle Descriptor NRUH-1:</b> This field contains the Reclaim Unit Handle Descriptor (refer to Figure 281) for Reclaim Unit Handle NRUH-1 (refer to Reclaim Unit Handle <i>N</i> -1 in Figure 70), if any.
$(NRUH*4)+63+VSS$ : $(NRUH*4)+64$	<b>Vendor Specific (VS):</b> This field is valid if the VSS field is greater than 0h.
$(NRUH*4)+64+VSS$	<b>Pad (PAD):</b> Padding to align the size of this descriptor to the next 8-byte boundary, if necessary. This field shall be cleared to 0h.

**Figure 281: Reclaim Unit Handle Descriptor**

Bytes	Description						
00	<p><b>Reclaim Unit Handle Type (RUHT):</b> This field indicates the Reclaim Unit Handle type of the Reclaim Unit Handle.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Reserved</td> </tr> <tr> <td>1h</td> <td><b>Initially Isolated:</b> The user data from a write command that utilizes this type of Reclaim Unit Handle is originally isolated in the referenced Reclaim Unit from other user data in write commands that utilize a different Reclaim Unit Handle in the same Reclaim Group. If the controller moves the user data due to vendor specific operations (i.e., garbage collection), then the controller is allowed to move that user data to a Reclaim Unit in the same Reclaim Group that contains other user data moved by the controller that was written by the host utilizing any Reclaim Unit Handle of the same type. Refer to section 8.1.10.</td> </tr> </tbody> </table>	Value	Definition	0h	Reserved	1h	<b>Initially Isolated:</b> The user data from a write command that utilizes this type of Reclaim Unit Handle is originally isolated in the referenced Reclaim Unit from other user data in write commands that utilize a different Reclaim Unit Handle in the same Reclaim Group. If the controller moves the user data due to vendor specific operations (i.e., garbage collection), then the controller is allowed to move that user data to a Reclaim Unit in the same Reclaim Group that contains other user data moved by the controller that was written by the host utilizing any Reclaim Unit Handle of the same type. Refer to section 8.1.10.
Value	Definition						
0h	Reserved						
1h	<b>Initially Isolated:</b> The user data from a write command that utilizes this type of Reclaim Unit Handle is originally isolated in the referenced Reclaim Unit from other user data in write commands that utilize a different Reclaim Unit Handle in the same Reclaim Group. If the controller moves the user data due to vendor specific operations (i.e., garbage collection), then the controller is allowed to move that user data to a Reclaim Unit in the same Reclaim Group that contains other user data moved by the controller that was written by the host utilizing any Reclaim Unit Handle of the same type. Refer to section 8.1.10.						

**Figure 281: Reclaim Unit Handle Descriptor**

Bytes	Description
	2h <b>Persistently Isolated:</b> The user data from a write command that utilizes this type of Reclaim Unit Handle is originally isolated in the referenced Reclaim Unit from other user data in write commands that utilize a different Reclaim Unit Handle. If the controller moves the user data due to vendor specific operations (i.e., garbage collection), then the controller shall move that user data to a Reclaim Unit in the same Reclaim Group that only contains user data that was written by the host utilizing the same Reclaim Unit Handle. Refer to section 8.1.10.
	3h to BFh Reserved
	C0h to FFh Vendor Specific
03:01	Reserved

The format of the Placement Identifier used in the Data Placement Directive (refer to section 8.1.8.4) for a specific FDP configuration is dependent on the number of Reclaim Groups supported by the FDP configuration in use and the number of bits allocated for the Reclaim Group Identifier field. If the NRG field (refer to Figure 280) is set to 1h indicating that there is only one Reclaim Group and zero bits are allocated to the Reclaim Group Identifier (i.e., the RGIF field (refer to Figure 280) is cleared to 0h), then the format of the Placement Identifier is defined in Figure 282. If one or more bits are allocated to the Reclaim Group Identifier (i.e., the RGIF field is non-zero), then the format of the Placement Identifier is defined in Figure 283.

**Figure 282: Placement Identifier Format without Reclaim Group Identifier**

Bits	Description
15:00	<b>Placement Handle (PHNDL):</b> This field specifies a Placement Handle associated with the namespace that is used by the controller to determine the Reclaim Unit Handle.

**Figure 283: Placement Identifier Format with a non-zero RGIF**

Bits	Description
15:(16-RGIF)	<b>Reclaim Group Identifier (RGID):</b> This field specifies the Reclaim Group used by the write command.  If the NRG field (refer to Figure 280) is set to 1h (i.e., there is only one Reclaim Group), then this field shall be ignored by the controller.
(15-RGIF):00	<b>Placement Handle (PHNDL):</b> This field specifies a Placement Handle associated with the namespace that is used by the controller to determine the Reclaim Unit Handle.

#### 5.1.12.1.29 Reclaim Unit Handle Usage (Log Page Identifier 21h)

The Reclaim Unit Handle Usage log page (refer to Figure 284) is used to provide information about the Reclaim Unit Handles associated with the Placement Handles of the namespaces in the specified Endurance Group. The Endurance Group is specified by the Endurance Group Identifier field in the Log Specific Identifier field as defined in Figure 217).

If Flexible Data Placement is enabled in the Endurance Group specified by the Endurance Group Identifier field and a Get Log Page command specifies the Reclaim Unit Handle Usage log page, then the NSID field in that command is reserved.

If Flexible Data Placement is disabled in the specified Endurance Group and the Get Log Page command specifying this log page, then the controller shall abort the command with a status code of FDP Disabled.

**Figure 284: Reclaim Unit Handle Usage Log Page**

Bytes	Description
<b>Header</b>	
01:00	<b>Number of Reclaim Unit Handles (NRUH):</b> This field identifies the number of Reclaim Unit Handle Usage Descriptors in the Reclaim Unit Handle Usage Descriptor List. This field shall be a non-zero value.  This field shall be set to the value of the NRUH field in the current FDP configuration (refer to Figure 280) for the specified Endurance Group.
07:02	Reserved
<b>Reclaim Unit Handle Usage Descriptor List</b>	
15:08	<b>Reclaim Unit Handle Usage Descriptor 0:</b> Contains the Reclaim Unit Handle Usage Descriptor (refer to Figure 285) for Reclaim Unit Handle 0.
23:16	<b>Reclaim Unit Handle Usage Descriptor 1:</b> Contains the Reclaim Unit Handle Usage Descriptor (refer to Figure 285) for Reclaim Unit Handle 1.
...	
(NRUH-1)*8+15: (NRUH-1)*8+8	<b>Reclaim Unit Handle Usage Descriptor NRUH-1:</b> Contains the Reclaim Unit Handle Usage Descriptor (refer to Figure 285) for Reclaim Unit Handle NRUH-1.

The Reclaim Unit Handle Usage Descriptor is defined in Figure 285.

**Figure 285: Reclaim Unit Handle Usage Descriptor**

Bytes	Description	
00	<b>Reclaim Unit Handle Attributes (RUHA):</b> This field identifies attributes associated with the Reclaim Unit Handle Usage.	
	<b>Value</b>	<b>Definition</b>
	0h	Not used by a namespace.
	1h	<b>Host Specified:</b> The Reclaim Unit Handle Identifier is used by a namespace created with a Namespace Management command explicitly requesting the usage of this Reclaim Unit Handle (i.e., specifying a Placement Handle List).
	2h	<b>Controller Specified:</b> The Reclaim Unit Handle Identifier is used by a namespace created with a Namespace Management command requesting the controller to select the one and only Reclaim Unit Handle (i.e., not specifying a Placement Handle List).
	3h to FFh	Reserved
07:01	Reserved	

In the list of Reclaim Unit Handle Usage Descriptors, no more than one entry shall have the Reclaim Unit Handle Attributes field set to Controller Specified (i.e., 2h).

#### 5.1.12.1.30 Flexible Data Placement (FDP) Statistics (Log Page Identifier 22h)

The FDP Statistics log page (refer to Figure 286) is used to provide information about the FDP configuration over the life of the FDP configuration in an Endurance Group. The Endurance Group is specified by the Endurance Group Identifier field in the Log Specific Identifier field as defined in Figure 217).

If Flexible Data Placement is enabled in the specified Endurance Group and the Get Log Page command specifies this log page, then the NSID field in that command is reserved.

If Flexible Data Placement is disabled in the specified Endurance Group and the Get Log Page command specifies this log page, then the controller shall abort the command with a status code of FDP Disabled.

**Figure 286: FDP Statistics Log Page**

Bytes	Description
15:00	<p><b>Host Bytes with Metadata Written (HBMW):</b> Contains the total number of bytes of user data the host has written to the specified Endurance Group as part of processing I/O commands as defined in the appropriate I/O Command Set specification. This value does not include controller writes due to internal operations such as garbage collection.</p> <p>This field shall not wrap once the value FFFFFFFF_FFFFFFFF_FFFFFFFF_FFFFFFFFh is reached.</p> <p>If the Flexible Data Placement Feature value is modified by a Set Features command, then this field shall be cleared to 0h.</p>
31:16	<p><b>Media Bytes with Metadata Written (MBMW):</b> Contains the total number of bytes of user data that have been written to the specified Endurance Group including both host and controller writes (e.g., garbage collection and background media scan operations) as part of processing as defined in the appropriate I/O Command Set specification.</p> <p>This field shall not wrap once the value FFFFFFFF_FFFFFFFF_FFFFFFFF_FFFFFFFFh is reached.</p> <p>If the Flexible Data Placement Feature value is modified by a Set Features command, then this field shall be cleared to 0h.</p>
47:32	<p><b>Media Bytes Erased (MBE):</b> Contains the total number of bytes of data that have been erased in the specified Endurance Group.</p> <p>This field shall not wrap once the value FFFFFFFF_FFFFFFFF_FFFFFFFF_FFFFFFFFh is reached.</p> <p>If the Flexible Data Placement Feature value is modified by a Set Features command, then this field shall be cleared to 0h.</p>
63:48	Reserved

The values reported in this log page are not cleared to 0h by a firmware update.

The values reported in this log page include operations on all namespaces that existed in the specified Endurance Group since the Flexible Data Placement was last enabled in that Endurance Group (refer to section 5.1.25.1.20).

#### 5.1.12.1.31 Flexible Data Placement (FDP) Events (Log Page Identifier 23h)

The FDP Events log page is used to provide information about events affecting Reclaim Units and media usage in an Endurance Group that has Flexible Data Placement enabled. The log page is 4 KiB bytes in size. The Endurance Group is specified by the Endurance Group Identifier in the Log Specific Identifier field as defined in Figure 217.

FDP events are associated with Reclaim Unit Handles. An FDP event shall only be reported if the FDP event occurs and that FDP event is enabled for that Reclaim Unit Handle (refer to section 5.1.25.1.21).

If:

- an FDP event occurs;
- that event is enabled; and
- the number of events recorded is the maximum supported,

then the oldest event shall be discarded.

The FDP Event Type bit (refer to Figure 287) specifies whether the controller reports host events or controller events in the log page. The controller shall not report both host events and controller events in the log page.

If Flexible Data Placement is enabled in the specified Endurance Group and a Get Log Page command specifies this log page, then the NSID field in that command is reserved.

If Flexible Data Placement is disabled in the specified Endurance Group, then a Get Log Page command that specifies this log page shall be aborted with a status code of FDP Disabled.

**Figure 287: Command Dword 10 – Log Specific Field**

Bits	Description
14:09	Reserved
08	<b>FDP Event Type (FDPET):</b> This bit specifies the type of events to be reported in the log page. If this bit is set to '1', then the controller shall report host events. If this bit is cleared to '0', then the controller shall report controller events.

**Figure 288: FDP Events Log Page**

Bytes	Description
<b>Header</b>	
03:00	<b>Number of FDP Events (NUMFDPE):</b> This field indicates the number of FDP events (refer to Figure 289) contained in this log.  If Flexible Data Placement transitions from disabled to enabled in an Endurance Group, then this field shall be cleared to 0h for that Endurance Group.
63:04	Reserved
<b>FDP Event List</b>	
127:64	<b>FDP Event 1:</b> The oldest FDP event, if any.
...	...
(NUMFDPC*64)+63: (NUMFDPC*64)	<b>FDP Event NUMFDPC:</b> The most recent FDP event, if any.

The FDP events shall be listed in ascending order of occurrence. Because the Timestamp feature may or may not have been set by the host after each Controller Level Reset and because the Event Timestamp field is the time at which the event was created, the FDP events may or may not be in order of ascending value of the Event Timestamp field.

The number of FDP events reported in the log page is equal to the value of the Number of FDP Events field. The values of the bytes in this log page which follow the FDP Event List is implementation specific.

The format of the FDP event is described in Figure 289.

**Figure 289: FDP Event**

Bytes	Description				
00	<b>Event Type (ETYP):</b> This field indicates the type of event:				
	<b>Value</b>	<b>O/M<sup>1</sup></b>	<b>ETS<sup>2</sup></b>	<b>Definition</b>	<b>Reference</b>
	<b>Host Events</b>				
	0h	O	No	Reclaim Unit Not Fully Written To Capacity	5.1.12.1.31.1.1
	1h	O	No	Reclaim Unit Time Limit Exceeded	5.1.12.1.31.1.2
	2h	O	No	Controller Level Reset Modified Reclaim Unit Handles	5.1.12.1.31.1.3
	3h	O	No	Invalid Placement Identifier	5.1.12.1.31.1.4
	4h to 6Fh			Reserved	
	70h to 7Fh	O		Vendor Specific Host Events	
	<b>Controller Events</b>				
	80h	O	Yes	Media Reallocated	5.1.12.1.31.2.1
	81h	O	No	Implicitly Modified Reclaim Unit Handle	5.1.12.1.31.2.2
	82h to EFh			Reserved	
	F0h to FFh	O		Vendor Specific Controller Events	
	Notes:				
	1. O/M definition: O = Optional, M = Mandatory.				
	2. Event Type Specific field: Yes = the field is utilized, No = the field is not utilized.				

Figure 289: FDP Event

Bytes	Description										
01	<b>FDP Event Flags (FDPEF):</b>										
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>07:03</td> <td>Reserved</td> </tr> <tr> <td>02</td> <td><b>Location Valid (LV):</b> If this bit is set to '1', then the Reclaim Group Identifier field and the Reclaim Unit Handle Identifier field contain a reported value. If this bit is cleared to '0', then the Reclaim Group Identifier field and the Reclaim Unit Handle Identifier field do not contain a reported value.</td> </tr> <tr> <td>01</td> <td><b>NSID Valid (NSIDV):</b> If this bit is set to '1', then the NSID field contains a reported value. If this bit is cleared to '0', then the NSID field does not contain a reported value.</td> </tr> <tr> <td>00</td> <td><b>Placement Identifier Valid (PIV):</b> If this bit is set to '1', then the Placement Identifier field contains a reported value. If this bit is cleared to '0', then the Placement Identifier field does not contain a reported value.</td> </tr> </tbody> </table>	Bits	Description	07:03	Reserved	02	<b>Location Valid (LV):</b> If this bit is set to '1', then the Reclaim Group Identifier field and the Reclaim Unit Handle Identifier field contain a reported value. If this bit is cleared to '0', then the Reclaim Group Identifier field and the Reclaim Unit Handle Identifier field do not contain a reported value.	01	<b>NSID Valid (NSIDV):</b> If this bit is set to '1', then the NSID field contains a reported value. If this bit is cleared to '0', then the NSID field does not contain a reported value.	00	<b>Placement Identifier Valid (PIV):</b> If this bit is set to '1', then the Placement Identifier field contains a reported value. If this bit is cleared to '0', then the Placement Identifier field does not contain a reported value.
	Bits	Description									
	07:03	Reserved									
	02	<b>Location Valid (LV):</b> If this bit is set to '1', then the Reclaim Group Identifier field and the Reclaim Unit Handle Identifier field contain a reported value. If this bit is cleared to '0', then the Reclaim Group Identifier field and the Reclaim Unit Handle Identifier field do not contain a reported value.									
01	<b>NSID Valid (NSIDV):</b> If this bit is set to '1', then the NSID field contains a reported value. If this bit is cleared to '0', then the NSID field does not contain a reported value.										
00	<b>Placement Identifier Valid (PIV):</b> If this bit is set to '1', then the Placement Identifier field contains a reported value. If this bit is cleared to '0', then the Placement Identifier field does not contain a reported value.										
03:02	<b>Placement Identifier (PID):</b> This field indicates the Placement Identifier associated with the FDP event. This field is reserved if the PIV bit is cleared to '0'. If the FDP event is an Invalid Placement Identifier event (i.e., Event Type 03h), then this field is Placement Identifier from the write command submitted by the host.										
11:04	<b>Event Timestamp (ETMSP):</b> This field indicates a timestamp indicating the current value of the Timestamp feature (refer to section 5.1.25.1.7) when this FDP event occurred. The format of this field is as defined in Figure 396).										
15:12	<b>Namespace Identifier (NSID):</b> This field indicates the identifier of the namespace associated with the FDP event. If the NSIDV bit is cleared to '0', then this field shall be cleared to 0h and the host should ignore this field.										
31:16	<b>Event Type Specific (ETSP):</b> Each Event Type indicates the use of this field and the format of this field.										
33:32	<b>Reclaim Group Identifier (RGICD):</b> This field indicates the Reclaim Group associated with the event. If the LV bit is cleared to '0', then this field shall be cleared to 0h and the host should ignore this field. If the FDP event is an Invalid Placement Identifier event (i.e., Event Type 03h), then this field indicates the Reclaim Group selected by the controller.										
35:34	<b>Reclaim Unit Handle Identifier (RUHID):</b> This field indicates the Reclaim Unit Handle associated with the FDP event. If the LV bit is cleared to '0', then this field shall be cleared to 0h and the host should ignore this field. If the FDP event is an Invalid Placement Identifier event (i.e., Event Type 03h), then this field indicates the Reclaim Unit Handle selected by the controller.										
39:36	Reserved										
<b>Vendor Specific Information</b>											
63:40	<b>Vendor Specific (VS):</b> This field may be used by any Event Type (i.e., this field is not only for Vendor specific Event Types).										

If the controller is not able to report the namespace identifier associated with the event, then the NSIDV bit shall be cleared to '0'.

If the controller is not able to report the specific Reclaim Unit Handle, then:

- the PIV bit shall be cleared to '0'; and
- the LV bit shall be cleared to '0'.

### 5.1.12.1.31.1 Host Events

#### 5.1.12.1.31.1.1 Reclaim Unit Not Fully Written To Capacity (Event Type 0h)

This FDP event indicates that the Reclaim Unit Handle reported in the Placement Identifier (refer to Figure 282 and Figure 283) was modified to reference a different Reclaim Unit due to an I/O Management Send command issued by the host prior to the previously referenced Reclaim Unit being written to capacity.

#### 5.1.12.1.31.1.2 Reclaim Unit Time Limit Exceeded (Event Type 1h)



This FDP event indicates that the capacity of a Reclaim Unit was not completely written within the time period defined in the Estimated Reclaim Unit Time Limit field (refer to Figure 280) due to the controller modifying the Reclaim Unit Handle to reference a different Reclaim Unit.

#### **5.1.12.1.31.1.3 Controller Level Reset Modified Reclaim Unit Handles (Event Type 2h)**

This FDP event indicates that one or more Reclaim Unit Handles were modified to reference a different Reclaim Unit due to a Controller Level Reset.

In the FDP event, the NSID field is reserved and the Placement Identifier field is reserved.

#### **5.1.12.1.31.1.4 Invalid Placement Identifier (Event Type 3h)**

This FDP event indicates that a write command specified an invalid Placement Identifier (refer to Figure 282 and Figure 283) due to an invalid Placement Handle or an invalid Reclaim Group Identifier.

### **5.1.12.1.31.2 Controller Events**

#### **5.1.12.1.31.2.1 Media Reallocated (Event Type 80h)**

This FDP event indicates that user data contained in a Reclaim Unit was written by the host specifying a Reclaim Unit Handle with a Reclaim Unit Handle type of Initially Isolated and was moved by the controller to a different Reclaim Unit (e.g., controller performed garbage collection).

The Event Type Specific field is I/O Command Set specific. Refer to the applicable I/O Command Set specification to determine if this log page is supported and the definition of the Event Type Specific field for this FDP event.

#### **5.1.12.1.31.2.2 Implicitly Modified Reclaim Unit Handle (Event Type 81h)**

This FDP event indicates that the controller modified a Reclaim Unit Handle to reference a different Reclaim Unit where that modification was not due to any host action (e.g., a write command that has a size greater than the remaining capacity of the initial Reclaim Unit being written).

### **5.1.12.1.32 Reservation Notification (Log Page Identifier 80h)**

The Reservation Notification log page reports one log page from a time ordered queue of Reservation Notification log pages, if available. A new Reservation Notification log page is created and added to the end of the queue of reservation notifications whenever an unmasked reservation notification occurs on any namespace that is attached to the controller. The Get Log Page command:

- returns a data buffer containing a log page corresponding to the oldest log page in the reservation notification queue (i.e., the log page containing the lowest Log Page Count field; accounting for wrapping); and
- removes that Reservation Notification log page from the queue.

If there are no available Reservation Notification log page entries when a Get Log Page command is issued, then an empty log page (i.e., all fields in the log page cleared to 0h) shall be returned.

If the controller is unable to store a reservation notification in the Reservation Notification log page due to the size of the queue, that reservation notification is lost. If a reservation notification is lost, then the controller shall increment the Log Page Count field of the last reservation notification in the queue (i.e., the Log Page Count field in the last reservation notification in the queue shall contain the value associated with the most recent reservation notification that has been lost).

The format of the log page is defined in Figure 290.

**Figure 290: Reservation Notification Log Page**

Bytes	Description												
07:00	<p><b>Log Page Count (LPC):</b> This is a 64-bit incrementing Reservation Notification log page count, indicating a unique identifier (modulo 64 bit) for this notification. The count starts at 0h following a Controller Level Reset and is incremented for every event that causes a reservation notification regardless of whether that notification is added to the queue. If the value of this field is FFFFFFFF_FFFFFFFFh, then the field is set to 1h when incremented (i.e., rolls over to 1h) and a new log page is created.</p> <p>If there are no Reservation Notification log pages to return (i.e., the queue of Reservation Notification log pages is empty), then this field shall return the value 0h. Subsequent reservation notifications continue incrementing this unique identifier from the last non-zero value (i.e., the value that identified the previous Reservation Notification log page). A value of 0h indicates the log page is empty.</p>												
08	<p><b>Reservation Notification Log Page Type (RNLPT):</b> This field indicates the Reservation Notification type described by this log page.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td><b>Empty Log Page:</b> Get Log Page command was processed when no unread Reservation Notification log pages were available. All the fields of an empty log page shall have a value of 0h.</td> </tr> <tr> <td>1h</td> <td><b>Registration Preempted</b></td> </tr> <tr> <td>2h</td> <td><b>Reservation Released</b></td> </tr> <tr> <td>3h</td> <td><b>Reservation Preempted</b></td> </tr> <tr> <td>4h to FFh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	<b>Empty Log Page:</b> Get Log Page command was processed when no unread Reservation Notification log pages were available. All the fields of an empty log page shall have a value of 0h.	1h	<b>Registration Preempted</b>	2h	<b>Reservation Released</b>	3h	<b>Reservation Preempted</b>	4h to FFh	Reserved
Value	Definition												
0h	<b>Empty Log Page:</b> Get Log Page command was processed when no unread Reservation Notification log pages were available. All the fields of an empty log page shall have a value of 0h.												
1h	<b>Registration Preempted</b>												
2h	<b>Reservation Released</b>												
3h	<b>Reservation Preempted</b>												
4h to FFh	Reserved												
09	<p><b>Number of Available Log Pages (NALP):</b> This field indicates the number of additional available Reservation Notification log pages (i.e., the number of unread log pages not counting this one). If there are more than 255 additional available log pages, then a value of 255 is returned. A value of 0h indicates that there are no additional available log pages.</p>												
11:10	Reserved												
15:12	<p><b>Namespace ID (NSID):</b> This field indicates the namespace ID of the namespace associated with the Reservation Notification described by this log page.</p>												
63:16	Reserved												

#### 5.1.12.1.33 Sanitize Status (Log Page Identifier 81h)

The Sanitize Status log page is used to report sanitize operation time estimates and information about the most recent sanitize operation (refer to section 8.1.24). The Get Log Page command returns a data buffer containing a log page formatted as defined in Figure 291. This log page shall be retained across power cycles and resets. This log page shall contain valid data whenever CSTS.RDY is set to '1'.

If the Sanitize Capabilities (SANICAP) field in the Identify Controller data structure is not cleared to 0h (i.e., the Sanitize command is supported), then this log page shall be supported. If the Sanitize Capabilities field in the Identify Controller data structure is cleared to 0h, then this log page is reserved.

**Figure 291: Sanitize Status Log Page**

Bytes	Description
01:00	<p><b>Sanitize Progress (SPROG):</b> This field indicates the fraction complete of the:</p> <ul style="list-style-type: none"> <li>sanitize processing state (i.e., the Restricted Processing state or the Unrestricted Processing state); or</li> <li>Post-Verification Deallocation state, if the Post-Verification Deallocation state is entered as part of the sanitize operation.</li> </ul> <p>The value is the numerator of the fraction complete that has 65,536 (10000h) as its denominator. This value shall be set to FFFFh if the Sanitize Operation Status (SOS) field is set to a value other than 010b (i.e., Sanitizing) or if the sanitization target is in the Media Verification state. Refer to section 8.1.24.3 for the effects of state changes that change the value of this field.</p>

03:02	<p><b>Sanitize Status (SSTAT):</b> This field indicates the status associated with the most recent sanitize operation.</p>																
	<b>Bits</b>	<b>Description</b>															
	15:10	Reserved															
	09	<p><b>Media Verification Canceled (MVCNCLD):</b> This bit indicates whether the Media Verification state is canceled.</p> <p>If the most recent sanitize operation was started by a Sanitize command with the EMVS bit set to '1' and during the Restricted Processing state, the Unrestricted Processing state, or the Media Verification state:</p> <ul style="list-style-type: none"> <li>a) a change in the composition of the NVM subsystem prevents media verification; or</li> <li>b) a Controller Level Reset of any controller in the NVM subsystem occurs that is caused by: <ul style="list-style-type: none"> <li>• a transport-specific reset type (refer to the applicable NVMe Transport specification); or</li> <li>• an NVM Subsystem Reset,</li> </ul> </li> </ul> <p>then this bit is set to '1'. Otherwise, this bit is cleared to '0'.</p> <p>Refer to the Sanitize Operation State Machine defined in section 8.1.24.3 for conditions that affect the setting of this bit.</p>															
	08	<p><b>Global Data Erased (GDE):</b> If no user data has been written to the NVM subsystem and no Persistent Memory Region in the NVM subsystem has been enabled:</p> <ul style="list-style-type: none"> <li>a) since being manufactured and the NVM subsystem has never been sanitized; or</li> <li>b) since the most recent successful sanitize operation,</li> </ul> <p>then this bit shall be set to '1'. Otherwise, this bit shall be cleared to '0'.</p>															
	07:03	<p><b>Overwrite Passes Completed (OPC):</b> This field shall indicate the number of completed passes if the most recent sanitize operation was an Overwrite. This field shall be cleared to 0h if the most recent sanitize operation was not an Overwrite.</p>															
02:00	<p><b>Sanitize Operation Status (SOS):</b> This field indicates the status of the most recent sanitize operation as shown below. Sanitize states are described in section 8.1.24.3.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Value</th> <th style="text-align: center;">State</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">000b</td> <td><b>Sanitize Never Started:</b> If a sanitize operation has never been started in the NVM subsystem, then this value shall be reported. Otherwise, this value shall not be reported.</td> </tr> <tr> <td style="text-align: center;">001b</td> <td><b>Sanitized:</b> If: <ul style="list-style-type: none"> <li>a) the most recent sanitize operation completed successfully; and</li> <li>b) the Sanitize Operation State Machine is in the Idle state,</li> </ul> then this value shall be reported. Otherwise, this value shall not be reported.</td> </tr> <tr> <td style="text-align: center;">010b</td> <td><b>Sanitizing:</b> If a sanitize operation is currently in progress (i.e., in the Restricted Processing state, the Unrestricted Processing state, the Media Verification state, or the Post-Verification Deallocation state), then this value shall be reported. Otherwise, this value shall not be reported.</td> </tr> <tr> <td style="text-align: center;">011b</td> <td><b>Sanitize Failed:</b> If the most recent sanitize operation failed, then this value shall be reported. Otherwise, this value shall not be reported.</td> </tr> <tr> <td style="text-align: center;">100b</td> <td>This value shall be reported when the Sanitize Operation State Machine is in the Restricted Failure state or the Unrestricted Failure state.</td> </tr> <tr> <td style="text-align: center;">100b</td> <td>This value is able to be reported when the Sanitize Operation State Machine is in the Idle state.</td> </tr> <tr> <td style="text-align: center;">100b</td> <td><b>Sanitized Unexpected Deallocate:</b> If:</td> </tr> </tbody> </table>	Value	State	000b	<b>Sanitize Never Started:</b> If a sanitize operation has never been started in the NVM subsystem, then this value shall be reported. Otherwise, this value shall not be reported.	001b	<b>Sanitized:</b> If: <ul style="list-style-type: none"> <li>a) the most recent sanitize operation completed successfully; and</li> <li>b) the Sanitize Operation State Machine is in the Idle state,</li> </ul> then this value shall be reported. Otherwise, this value shall not be reported.	010b	<b>Sanitizing:</b> If a sanitize operation is currently in progress (i.e., in the Restricted Processing state, the Unrestricted Processing state, the Media Verification state, or the Post-Verification Deallocation state), then this value shall be reported. Otherwise, this value shall not be reported.	011b	<b>Sanitize Failed:</b> If the most recent sanitize operation failed, then this value shall be reported. Otherwise, this value shall not be reported.	100b	This value shall be reported when the Sanitize Operation State Machine is in the Restricted Failure state or the Unrestricted Failure state.	100b	This value is able to be reported when the Sanitize Operation State Machine is in the Idle state.	100b	<b>Sanitized Unexpected Deallocate:</b> If:
Value	State																
000b	<b>Sanitize Never Started:</b> If a sanitize operation has never been started in the NVM subsystem, then this value shall be reported. Otherwise, this value shall not be reported.																
001b	<b>Sanitized:</b> If: <ul style="list-style-type: none"> <li>a) the most recent sanitize operation completed successfully; and</li> <li>b) the Sanitize Operation State Machine is in the Idle state,</li> </ul> then this value shall be reported. Otherwise, this value shall not be reported.																
010b	<b>Sanitizing:</b> If a sanitize operation is currently in progress (i.e., in the Restricted Processing state, the Unrestricted Processing state, the Media Verification state, or the Post-Verification Deallocation state), then this value shall be reported. Otherwise, this value shall not be reported.																
011b	<b>Sanitize Failed:</b> If the most recent sanitize operation failed, then this value shall be reported. Otherwise, this value shall not be reported.																
100b	This value shall be reported when the Sanitize Operation State Machine is in the Restricted Failure state or the Unrestricted Failure state.																
100b	This value is able to be reported when the Sanitize Operation State Machine is in the Idle state.																
100b	<b>Sanitized Unexpected Deallocate:</b> If:																

Figure 291: Sanitize Status Log Page

Bytes	Description
	<p>a) the most recent sanitize operation for which No-Deallocate After Sanitize (refer to section 5.1.22) was requested has completed successfully with deallocation of all media allocated for user data (refer to section 5.1.25.1.15); and</p> <p>b) the Sanitize Operation State Machine is in the Idle state, then this value shall be reported. Otherwise, this value shall not be reported.</p> <p>101b to 111b Reserved</p>
07:04	<p><b>Sanitize Command Dword 10 Information (SCDW10):</b> This field shall indicate the value of the Command Dword 10 field of the Sanitize command that started the sanitize operation whose status is reported in the SSTAT field. Refer to Figure 372.</p>
11:08	<p><b>Estimated Time For Overwrite (ETO):</b> This field shall indicate the number of seconds required to complete sanitize processing (i.e., the time difference between entering and exiting the Restricted Processing state or the Unrestricted Processing state) of an Overwrite sanitize operation with 16 passes in the background (refer to section 5.1.22) when the No-Deallocate Modifies Media After Sanitize field (refer to Figure 312) is set to a value other than 10b.</p> <p>A value of 0h indicates that the sanitize processing is expected to be completed in the background when the Sanitize command that started that operation is completed. A value of FFFFFFFFh indicates that no time period is reported.</p>
15:12	<p><b>Estimated Time For Block Erase (ETBE):</b> This field shall indicate the number of seconds required to complete sanitize processing (i.e., the time difference between entering and exiting the Restricted Processing state or the Unrestricted Processing state) of a Block Erase sanitize operation in the background (refer to section 5.1.22) when the No-Deallocate Modifies Media After Sanitize field (refer to Figure 312) is set to a value other than 10b.</p> <p>A value of 0h indicates that the sanitize processing is expected to be completed in the background when the Sanitize command that started that operation is completed. A value of FFFFFFFFh indicates that no time period is reported.</p>
19:16	<p><b>Estimated Time For Crypto Erase (ETCE):</b> This field shall indicate the number of seconds required to complete sanitize processing (i.e., the time difference between entering and exiting the Restricted Processing state or the Unrestricted Processing state) of a Crypto Erase sanitize operation in the background (refer to section 5.1.22) when the No-Deallocate Modifies Media After Sanitize field (refer to Figure 312) is set to a value other than 10b.</p> <p>A value of 0h indicates that the sanitize processing is expected to be completed in the background when the Sanitize command that started that operation is completed. A value of FFFFFFFFh indicates that no time period is reported.</p>
23:20	<p><b>Estimated Time For Overwrite With No-Deallocate Media Modification (ETODMM):</b> This field shall indicate the number of seconds required to complete sanitize processing (i.e., the time difference between entering and exiting the Restricted Processing state or the Unrestricted Processing state) of an Overwrite sanitize operation, including associated additional media modification, in the background (refer to section 5.1.22) when:</p> <ul style="list-style-type: none"> <li>a) the No-Deallocate After Sanitize bit was set to '1' in the Sanitize command that requested the Overwrite sanitize operation; and</li> <li>b) the No-Deallocate Modifies Media After Sanitize field (refer to Figure 312) is set to 10b.</li> </ul> <p>A value of 0h indicates that the sanitize processing is expected to be completed in the background when the Sanitize command that started that operation is completed. A value of FFFFFFFFh indicates that no time period is reported.</p>

Figure 291: Sanitize Status Log Page

Bytes	Description																																	
27:24	<p><b>Estimated Time For Block Erase With No-Deallocate Media Modification (ETBENMM):</b> This field shall indicate the number of seconds required to complete sanitize processing (i.e., the time difference between entering and exiting the Restricted Processing state or the Unrestricted Processing state) of a Block Erase sanitize operation, including associated additional media modification, in the background (refer to section 5.1.22) when:</p> <ul style="list-style-type: none"> <li>a) the No-Deallocate After Sanitize bit was set to '1' in the Sanitize command that requested the Block Erase sanitize operation; and</li> <li>b) the No-Deallocate Modifies Media After Sanitize field (refer to Figure 312) is set to 10b.</li> </ul> <p>A value of 0h indicates that the sanitize processing is expected to be completed in the background when the Sanitize command that started that operation is completed. A value of FFFFFFFFh indicates that no time period is reported.</p>																																	
31:28	<p><b>Estimated Time For Crypto Erase With No-Deallocate Media Modification (ETCENMM):</b> This field shall indicate the number of seconds required to complete sanitize processing (i.e., the time difference between entering and exiting the Restricted Processing state or the Unrestricted Processing state) of a Crypto Erase sanitize operation, including associated additional media modification, in the background (refer to section 5.1.22) when:</p> <ul style="list-style-type: none"> <li>a) the No-Deallocate After Sanitize bit was set to '1' in the Sanitize command that requested the Crypto Erase sanitize operation; and</li> <li>b) the No-Deallocate Modifies Media After Sanitize field (refer to Figure 312) is set to 10b.</li> </ul> <p>A value of 0h indicates that the sanitize processing is expected to be completed in the background when the Sanitize command that started that operation is completed. A value of FFFFFFFFh indicates that no time period is reported.</p>																																	
35:32	<p><b>Estimated Time For Post-Verification Deallocation State (ETPVDS):</b> This field shall indicate the number of seconds required to deallocate all media allocated for user data after exiting the Media Verification state (i.e., the time difference between entering and exiting the Post-Verification Deallocation state), if that state is entered as part of the sanitize operation. A value of FFFFFFFFh indicates that no time period is reported.</p>																																	
36	<p><b>Sanitize State Information (SSI):</b> This field indicates additional information about the state of sanitization.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:4</td> <td> <p><b>Failure State (FAILS):</b> If the SOS field is set to 011b (i.e., Sanitize Failed), then this field shall indicate the state of the Sanitize Operation State Machine (refer to Figure 648) in which the failure occurred.</p> <p>The values of this field are sanitize states, defined in the description of the SANS field.</p> <p>If the VERS bit is cleared to '0', then this field shall be cleared to 0h.</p> <p>If the SOS field is set to a value other than 011b (i.e., Sanitize Failed), then this field shall be cleared to 0h.</p> </td> </tr> <tr> <td>3:0</td> <td> <p><b>Sanitize State (SANS):</b> This field shall indicate the current state of the Sanitize Operation State Machine (refer to Figure 648).</p> <p>If the VERS bit is cleared to '0', then this field shall be cleared to 0h.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th>Value</th> <th>State</th> <th>Reference</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Idle</td> <td>8.1.24.3.1</td> </tr> <tr> <td>1h</td> <td>Restricted Processing</td> <td>8.1.24.3.2</td> </tr> <tr> <td>2h</td> <td>Restricted Failure</td> <td>8.1.24.3.3</td> </tr> <tr> <td>3h</td> <td>Unrestricted Processing</td> <td>8.1.24.3.4</td> </tr> <tr> <td>4h</td> <td>Unrestricted Failure</td> <td>8.1.24.3.5</td> </tr> <tr> <td>5h</td> <td>Media Verification</td> <td>8.1.24.3.6</td> </tr> <tr> <td>6h</td> <td>Post-Verification Deallocation</td> <td>8.1.24.3.7</td> </tr> <tr> <td>All other values</td> <td>Reserved</td> <td></td> </tr> </tbody> </table> </td> </tr> </tbody> </table>	Bits	Description	7:4	<p><b>Failure State (FAILS):</b> If the SOS field is set to 011b (i.e., Sanitize Failed), then this field shall indicate the state of the Sanitize Operation State Machine (refer to Figure 648) in which the failure occurred.</p> <p>The values of this field are sanitize states, defined in the description of the SANS field.</p> <p>If the VERS bit is cleared to '0', then this field shall be cleared to 0h.</p> <p>If the SOS field is set to a value other than 011b (i.e., Sanitize Failed), then this field shall be cleared to 0h.</p>	3:0	<p><b>Sanitize State (SANS):</b> This field shall indicate the current state of the Sanitize Operation State Machine (refer to Figure 648).</p> <p>If the VERS bit is cleared to '0', then this field shall be cleared to 0h.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th>Value</th> <th>State</th> <th>Reference</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Idle</td> <td>8.1.24.3.1</td> </tr> <tr> <td>1h</td> <td>Restricted Processing</td> <td>8.1.24.3.2</td> </tr> <tr> <td>2h</td> <td>Restricted Failure</td> <td>8.1.24.3.3</td> </tr> <tr> <td>3h</td> <td>Unrestricted Processing</td> <td>8.1.24.3.4</td> </tr> <tr> <td>4h</td> <td>Unrestricted Failure</td> <td>8.1.24.3.5</td> </tr> <tr> <td>5h</td> <td>Media Verification</td> <td>8.1.24.3.6</td> </tr> <tr> <td>6h</td> <td>Post-Verification Deallocation</td> <td>8.1.24.3.7</td> </tr> <tr> <td>All other values</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Value	State	Reference	0h	Idle	8.1.24.3.1	1h	Restricted Processing	8.1.24.3.2	2h	Restricted Failure	8.1.24.3.3	3h	Unrestricted Processing	8.1.24.3.4	4h	Unrestricted Failure	8.1.24.3.5	5h	Media Verification	8.1.24.3.6	6h	Post-Verification Deallocation	8.1.24.3.7	All other values	Reserved	
	Bits	Description																																
	7:4	<p><b>Failure State (FAILS):</b> If the SOS field is set to 011b (i.e., Sanitize Failed), then this field shall indicate the state of the Sanitize Operation State Machine (refer to Figure 648) in which the failure occurred.</p> <p>The values of this field are sanitize states, defined in the description of the SANS field.</p> <p>If the VERS bit is cleared to '0', then this field shall be cleared to 0h.</p> <p>If the SOS field is set to a value other than 011b (i.e., Sanitize Failed), then this field shall be cleared to 0h.</p>																																
3:0	<p><b>Sanitize State (SANS):</b> This field shall indicate the current state of the Sanitize Operation State Machine (refer to Figure 648).</p> <p>If the VERS bit is cleared to '0', then this field shall be cleared to 0h.</p> <table border="1" style="width: 100%;"> <thead> <tr> <th>Value</th> <th>State</th> <th>Reference</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Idle</td> <td>8.1.24.3.1</td> </tr> <tr> <td>1h</td> <td>Restricted Processing</td> <td>8.1.24.3.2</td> </tr> <tr> <td>2h</td> <td>Restricted Failure</td> <td>8.1.24.3.3</td> </tr> <tr> <td>3h</td> <td>Unrestricted Processing</td> <td>8.1.24.3.4</td> </tr> <tr> <td>4h</td> <td>Unrestricted Failure</td> <td>8.1.24.3.5</td> </tr> <tr> <td>5h</td> <td>Media Verification</td> <td>8.1.24.3.6</td> </tr> <tr> <td>6h</td> <td>Post-Verification Deallocation</td> <td>8.1.24.3.7</td> </tr> <tr> <td>All other values</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Value	State	Reference	0h	Idle	8.1.24.3.1	1h	Restricted Processing	8.1.24.3.2	2h	Restricted Failure	8.1.24.3.3	3h	Unrestricted Processing	8.1.24.3.4	4h	Unrestricted Failure	8.1.24.3.5	5h	Media Verification	8.1.24.3.6	6h	Post-Verification Deallocation	8.1.24.3.7	All other values	Reserved							
Value	State	Reference																																
0h	Idle	8.1.24.3.1																																
1h	Restricted Processing	8.1.24.3.2																																
2h	Restricted Failure	8.1.24.3.3																																
3h	Unrestricted Processing	8.1.24.3.4																																
4h	Unrestricted Failure	8.1.24.3.5																																
5h	Media Verification	8.1.24.3.6																																
6h	Post-Verification Deallocation	8.1.24.3.7																																
All other values	Reserved																																	
511:37	Reserved																																	

### 5.1.12.2 Memory-Based Log Specific Information (PCIe)

There are no log pages that are specific to the Memory-based transport model.

### 5.1.12.3 Message-Based Log Specific Information (Fabrics)

This section describes log pages that are specific to the Message-based transport model.

#### 5.1.12.3.1 Discovery (Log Page Identifier 70h)

The Discovery log page shall only be supported by Discovery controllers. The Discovery log page shall not be supported by controllers that expose namespaces for NVMe over PCIe or NVMe over Fabrics. The Discovery log page provides an inventory of NVM subsystems with which a host may attempt to form an association. Depending on the parameters specified in a Get Log Page command, the Discovery log page may contain entries for:

- all known NVM subsystems; or
- a subset of entries specific to the host or Discovery controller requesting the log page.

The Discovery log page is persistent across power cycles.

Figure 292 specifies the format for the LID Specific Parameter field in the Supported Log Pages log page (refer to section 5.1.12.1.1) for the Discovery log page.

**Figure 292: Discovery Log Page – LID Specific Parameter Field**

Bits	Description
15:3	Reserved
2	<b>All NVM Subsystem Entries Supported (ALLSUBES):</b> If this bit is set to '1', then the Discovery controller supports returning records for all NVM subsystem ports that are registered without filtering the list of returned records based upon the requesting HOSTNQN. If this bit is cleared to '0', then the Discovery controller does not support returning records for all NVM subsystem ports that are registered.
1	<b>Port Local Entries Only Supported (PLEOS):</b> If this bit is set to '1', then the Discovery controller supports returning only port local Discovery Log Page Entries or port local Extended Discovery Log Page Entries. If this bit is cleared to '0', then the Discovery controller does not support returning only port local Discovery Log Page Entries or port local Extended Discovery Log Page Entries. If the Discovery controller is not a DDC, then this bit should be ignored.
0	<b>Extended Discovery Log Page Entry Supported (EXTDLPE):</b> If this bit is set to '1', then the Discovery controller supports returning Extended Discovery Log Page Entries. If this bit is cleared to '0', then the Discovery controller does not support returning Extended Discovery Log Page Entries. If the Discovery controller is a CDC, then that CDC shall support returning Extended Discovery Log Page Entries.

If a Discovery controller supports returning Extended Discovery Log Page Entries (i.e., the Extended Discovery Log Page Entries Supported (EXTDLPE) bit is set to '1' in the LID Specific Parameter field as defined in Figure 292) and that Discovery controller processes a Get Log Page command for this log page with the Extended Discovery Log Page Entries (EXTDLPE) bit set to '1' in the Log Specific Parameter (LSP) field (refer to Figure 293), then each entry returned by that Discovery controller may be either a Discovery Log Page Entry or an Extended Discovery Log Page Entry. Extended Discovery Log Page Entries may contain zero or more extended attributes. If a Discovery controller returns records that include Extended Discovery Log Page Entries, then that Discovery controller shall set the Extended (EXTEND) bit to '1' in the Discovery Log Page Flags (DLPF) field. A Centralized Discovery controller (CDC) shall support returning Extended Discovery Log Page Entries.

If a Discovery controller supports returning only port local Discovery Log Page Entries or port local Extended Discovery Log Page Entries (i.e., the Port Local Entries Only Supported (PLEOS) bit is set to '1' in the LID Specific Parameter field as defined in Figure 292) and that Discovery controller processes a Get Log Page command for this log page with the Port Local Entries Only (PLEO) bit set to '1' in the LSP field (refer to Figure 293), then that Discovery controller shall return records for only NVM subsystem ports that are presented through the same NVM subsystem port that received the Get Log Page command. If a Discovery controller returns records for only NVM subsystem ports that are presented through the same NVM subsystem port that received the Get Log Page command, then that Discovery controller shall set the Port

Local (PORTLCL) bit to '1' in the DLPF field. Only a Direct Discovery controller (DDC) may support returning only port local Discovery Log Page Entries or port local Extended Discovery Log Page Entries.

If a Discovery controller supports returning records for all NVM subsystem ports that are registered (i.e., the All NVM Subsystems Entries Supported (ALLSUBES) bit is set to '1' in the LID Specific Parameter field as defined in Figure 292) and that Discovery controller processes a Get Log Page command for this log page with the All NVM Subsystem Entries (ALLSUBE) bit set to '1' in the LSP field (refer to Figure 293), then that Discovery controller shall return records for all NVM subsystem ports that are registered. The list of records returned by that Discovery controller shall not be filtered based upon the requesting HOSTNQN. If a Discovery controller returns records for all NVM subsystem ports that are registered, then that Discovery controller shall set the All Subsystems (ALLSUB) bit to '1' in the DLPF field. A Discovery controller may require authentication before reporting the ALLSUBES bit being set to '1' or may require configuration outside the scope of this specification.

The Log Page Offset may be used to retrieve specific records. If the Discovery controller supports an index offset for this log page (i.e., the Index Offset Supported (IOS) bit is set to '1' in the Supported Log Pages log page (refer to section 5.1.12.1.1) for this log page), then the index values specified in the Log Page Offset Lower (LPOL) field (refer to Figure 199) and Log Page Offset Upper (LPOU) field (refer to Figure 200) in the Get Log Page command shall be used to index into the header and list of entries within this log page (e.g., specifying an index offset of 0 returns this log page starting at the header, specifying an index offset of 1 returns this log page starting at the offset of Entry 0, etc.). The number of records is returned in the header of the log page. The format for a Discovery Log Page Entry is defined in Figure 294. The format for an Extended Discovery Log Page Entry is defined in Figure 295. The format for an extended attribute is defined in Figure 499. The format for the Discovery log page is defined in Figure 296.

A single Get Log Page command used to read the Discovery log page shall be atomic. If the Discovery log page is read using multiple Get Log Page commands, then the reader should validate the Generation Counter field to ensure that there has not been a change in the contents of the data. The Discovery log page contents should be processed in the following sequence:

1. read the Discovery log page contents in order (i.e., with increasing Log Page Offset values);
2. re-read the Generation Counter field after the entire log page is transferred; and
3. if the Generation Counter field does not match the original value read, then discard the log page read, as the entries may be inconsistent, and restart the sequence.; or
4. if the Generation Counter field does match the original value read, then process the returned data.

If the log page contents change during this command sequence, then the controller may return a status code of Discover Restart.

Every record indicates via the SUBTYPE field if that record is describing the current Discovery Service, referring to another Discovery Service, or if the record indicates an NVM subsystem composed of controllers that may expose namespaces. A referral to another Discovery Service (i.e., SUBTYPE field set to 01h) is a mechanism to find additional Discovery subsystems. An NVM subsystem entry (i.e., SUBTYPE field set to 02h) is a mechanism to find NVM subsystems that contain controllers that may expose namespaces. An entry that describes the current Discovery Service (i.e., SUBTYPE field set to 03h) is a mechanism to find additional access information (e.g., other NVM subsystem ports) for the current Discovery service. Hosts or Discovery controllers are not required to process referral entries deeper than eight levels.

For entries that describe the current Discovery Service, the DUPRETINFO bit set to '1' in the Entry Flags field indicates those ports that return duplicate information. A host need not retrieve the Discovery Log Page from all NVM subsystem ports described by a Discovery Log Page Entry that contains the DUPRETINFO bit set to '1' (refer to Figure 294), as each of those ports return the same information. A host may retrieve the Discovery Log Page from multiple ports with the DUPRETINFO bit set to '1' to act as redundant access to the same information. A host may choose to enable the Discovery Log Page Change AEN on one of the connections associated with an entry with the DUPRETINFO bit set to '1'. If the Discovery Log Page Change AEN is enabled on multiple ports represented by entries with the DUPRETINFO bit set to '1', then the host may process a Discovery Log Page Change AEN for one connection as applying to all ports represented by entries that also have the DUPRETINFO bit set to '1'.

If an NVM subsystem supports the dynamic controller model, then all entries for that NVM subsystem shall have the Controller ID field set to FFFFh. For a particular NVM subsystem port and NVMe Transport address in an NVM subsystem, there shall be no more than one entry with the Controller ID field set to:

- FFFFh if that NVM subsystem supports the dynamic controller model; or
- FFFEh if that NVM subsystem supports the static controller model.

**Figure 293: Discovery Log Specific Parameter Field**

Bits	Description
14:11	Reserved
10	<b>All NVM Subsystem Entries (ALLSUBE):</b> If this bit is set to '1', then the Discovery controller shall return records for all NVM subsystem ports that are registered without filtering the list of returned records based upon the requesting HOSTNQN. If this bit is cleared to '0', then this bit has no effect.
09	<b>Port Local Entries Only (PLEO):</b> If this bit is set to '1', then the Discovery controller shall return records for only NVM subsystem ports that are presented through the same NVM subsystem port that received the Get Log Page command. If this bit is cleared to '0', then this bit has no effect.
08	<b>Extended Discovery Log Page Entries (EXTDLPE):</b> If this bit is set to '1', then each record returned by the Discovery controller may be either a Discovery Log Page Entry or an Extended Discovery Log Page Entry. If this bit is cleared to '0', then this bit has no effect.

**Figure 294: Discovery Log Page Entry Data Structure**

Bytes	Description																		
00	<b>Transport Type (TRTYPE):</b> Specifies the NVMe Transport type.																		
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00</td> <td>Reserved</td> </tr> <tr> <td>01</td> <td>RDMA Transport (refer to the NVMe RDMA Transport specification)</td> </tr> <tr> <td>02</td> <td>Fibre Channel Transport (refer to INCITS 556)</td> </tr> <tr> <td>03</td> <td>TCP Transport (refer to the NVMe TCP Transport specification)</td> </tr> <tr> <td>04 to 253</td> <td>Reserved</td> </tr> <tr> <td>254</td> <td>Intra-host Transport (i.e., loopback) (NOTE: This is a reserved value for use by host software.)</td> </tr> <tr> <td>255</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00	Reserved	01	RDMA Transport (refer to the NVMe RDMA Transport specification)	02	Fibre Channel Transport (refer to INCITS 556)	03	TCP Transport (refer to the NVMe TCP Transport specification)	04 to 253	Reserved	254	Intra-host Transport (i.e., loopback) (NOTE: This is a reserved value for use by host software.)	255	Reserved		
	Value	Definition																	
	00	Reserved																	
	01	RDMA Transport (refer to the NVMe RDMA Transport specification)																	
	02	Fibre Channel Transport (refer to INCITS 556)																	
	03	TCP Transport (refer to the NVMe TCP Transport specification)																	
	04 to 253	Reserved																	
254	Intra-host Transport (i.e., loopback) (NOTE: This is a reserved value for use by host software.)																		
255	Reserved																		
01	<b>Address Family (ADRFAM):</b> Specifies the address family.																		
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00</td> <td>Reserved</td> </tr> <tr> <td>01</td> <td><b>AF_INET:</b> IPv4 address family. IPv4address format syntax specified in section 3.2.2 of IETF RFC 3986.</td> </tr> <tr> <td>02</td> <td><b>AF_INET6:</b> IPv6 address family. IPv6address format syntax specified in section 3.2.2 of IETF RFC 3986.</td> </tr> <tr> <td>03</td> <td><b>AF_IB:</b> InfiniBand address family.</td> </tr> <tr> <td>04</td> <td>Fibre Channel address family.</td> </tr> <tr> <td>05 to 253</td> <td>Reserved</td> </tr> <tr> <td>254</td> <td>Intra-host Transport (i.e., loopback) (NOTE: This is a reserved value for use by host software.)</td> </tr> <tr> <td>255</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00	Reserved	01	<b>AF_INET:</b> IPv4 address family. IPv4address format syntax specified in section 3.2.2 of IETF RFC 3986.	02	<b>AF_INET6:</b> IPv6 address family. IPv6address format syntax specified in section 3.2.2 of IETF RFC 3986.	03	<b>AF_IB:</b> InfiniBand address family.	04	Fibre Channel address family.	05 to 253	Reserved	254	Intra-host Transport (i.e., loopback) (NOTE: This is a reserved value for use by host software.)	255	Reserved
	Value	Definition																	
	00	Reserved																	
	01	<b>AF_INET:</b> IPv4 address family. IPv4address format syntax specified in section 3.2.2 of IETF RFC 3986.																	
	02	<b>AF_INET6:</b> IPv6 address family. IPv6address format syntax specified in section 3.2.2 of IETF RFC 3986.																	
	03	<b>AF_IB:</b> InfiniBand address family.																	
	04	Fibre Channel address family.																	
05 to 253	Reserved																		
254	Intra-host Transport (i.e., loopback) (NOTE: This is a reserved value for use by host software.)																		
255	Reserved																		



**Figure 294: Discovery Log Page Entry Data Structure**

Bytes	Description																																
02	<p><b>Subsystem Type (SUBTYPE):</b> Specifies the type of the NVM subsystem that is indicated in this entry.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00</td> <td>Reserved</td> </tr> <tr> <td>01</td> <td><b>Referral:</b> The entry describes a referral to another Discovery Service composed of Discovery controllers that provide additional discovery records. Multiple Referral entries may be reported for each Discovery Service (e.g., if that Discovery Service has multiple NVM subsystem ports or supports multiple protocols).</td> </tr> <tr> <td>02</td> <td><b>NVM Subsystem:</b> The entry describes an NVM subsystem whose controllers may have attached namespaces (i.e., an NVM subsystem that is not composed of Discovery controllers). Multiple NVM Subsystem entries may be reported for each NVM subsystem if that NVM subsystem has multiple NVM subsystem ports.</td> </tr> <tr> <td>03</td> <td><b>Current Discovery Subsystem:</b> The entry describes this Discovery subsystem (i.e., the Discovery Service that contains the controller processing the Get Log Page command). Multiple Current Discovery Subsystem entries may be reported for this Discovery subsystem if the current Discovery subsystem has multiple NVM subsystem ports.</td> </tr> <tr> <td>04 to 255</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00	Reserved	01	<b>Referral:</b> The entry describes a referral to another Discovery Service composed of Discovery controllers that provide additional discovery records. Multiple Referral entries may be reported for each Discovery Service (e.g., if that Discovery Service has multiple NVM subsystem ports or supports multiple protocols).	02	<b>NVM Subsystem:</b> The entry describes an NVM subsystem whose controllers may have attached namespaces (i.e., an NVM subsystem that is not composed of Discovery controllers). Multiple NVM Subsystem entries may be reported for each NVM subsystem if that NVM subsystem has multiple NVM subsystem ports.	03	<b>Current Discovery Subsystem:</b> The entry describes this Discovery subsystem (i.e., the Discovery Service that contains the controller processing the Get Log Page command). Multiple Current Discovery Subsystem entries may be reported for this Discovery subsystem if the current Discovery subsystem has multiple NVM subsystem ports.	04 to 255	Reserved																				
	Value	Definition																															
	00	Reserved																															
	01	<b>Referral:</b> The entry describes a referral to another Discovery Service composed of Discovery controllers that provide additional discovery records. Multiple Referral entries may be reported for each Discovery Service (e.g., if that Discovery Service has multiple NVM subsystem ports or supports multiple protocols).																															
	02	<b>NVM Subsystem:</b> The entry describes an NVM subsystem whose controllers may have attached namespaces (i.e., an NVM subsystem that is not composed of Discovery controllers). Multiple NVM Subsystem entries may be reported for each NVM subsystem if that NVM subsystem has multiple NVM subsystem ports.																															
03	<b>Current Discovery Subsystem:</b> The entry describes this Discovery subsystem (i.e., the Discovery Service that contains the controller processing the Get Log Page command). Multiple Current Discovery Subsystem entries may be reported for this Discovery subsystem if the current Discovery subsystem has multiple NVM subsystem ports.																																
04 to 255	Reserved																																
03	<p><b>Transport Requirements (TREQ):</b> Indicates requirements for the NVMe Transport.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:6</td> <td>Reserved</td> </tr> <tr> <td>5:4</td> <td> <p><b>Transport Authentication and Secure Channel (TASC):</b> Indicates whether an authentication transaction (refer to section 8.3.4.2) or an authentication transaction concatenated to a secure channel establishment (refer to section 8.3.4.3) is required upon completion of the Connect command. These bits do not override the AUTHREQ bits in the Connect response; these bits apply only when the AUTHREQ bits specify no requirements (refer to Figure 548).</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Not specified</td> </tr> <tr> <td>01b</td> <td>Authentication required</td> </tr> <tr> <td>10b</td> <td>Authentication concatenated to secure channel establishment required</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table> </td> </tr> <tr> <td>3</td> <td> <p><b>Zero Host ID Support (ZHIDS):</b> If this bit is set to '1', then the controller supports a Host Identifier value of 0h in a Connect command. If this bit is cleared to '0', then the controller does not support a Host Identifier value of 0h in a Connect command.</p> </td> </tr> <tr> <td>2</td> <td> <p><b>SQ Flow Control Disable (SQFCD):</b> If this bit is set to '1', then the controller is capable of disabling SQ flow control. A controller that is capable of disabling SQ flow control may accept or reject a host request to disable SQ flow control. If this bit is cleared to '0', then the controller requires use of SQ flow control.</p> </td> </tr> <tr> <td>1:0</td> <td> <p><b>Transport Secure Channel (TSC):</b> Indicates whether connections shall be made over a fabric secure channel (refer to section 8.3.4.1).</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Not specified</td> </tr> <tr> <td>01b</td> <td>Required</td> </tr> <tr> <td>10b</td> <td>Not required (i.e., not supported or supported but not required)</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table> </td> </tr> </tbody> </table>	Bits	Description	7:6	Reserved	5:4	<p><b>Transport Authentication and Secure Channel (TASC):</b> Indicates whether an authentication transaction (refer to section 8.3.4.2) or an authentication transaction concatenated to a secure channel establishment (refer to section 8.3.4.3) is required upon completion of the Connect command. These bits do not override the AUTHREQ bits in the Connect response; these bits apply only when the AUTHREQ bits specify no requirements (refer to Figure 548).</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Not specified</td> </tr> <tr> <td>01b</td> <td>Authentication required</td> </tr> <tr> <td>10b</td> <td>Authentication concatenated to secure channel establishment required</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	Not specified	01b	Authentication required	10b	Authentication concatenated to secure channel establishment required	11b	Reserved	3	<p><b>Zero Host ID Support (ZHIDS):</b> If this bit is set to '1', then the controller supports a Host Identifier value of 0h in a Connect command. If this bit is cleared to '0', then the controller does not support a Host Identifier value of 0h in a Connect command.</p>	2	<p><b>SQ Flow Control Disable (SQFCD):</b> If this bit is set to '1', then the controller is capable of disabling SQ flow control. A controller that is capable of disabling SQ flow control may accept or reject a host request to disable SQ flow control. If this bit is cleared to '0', then the controller requires use of SQ flow control.</p>	1:0	<p><b>Transport Secure Channel (TSC):</b> Indicates whether connections shall be made over a fabric secure channel (refer to section 8.3.4.1).</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Not specified</td> </tr> <tr> <td>01b</td> <td>Required</td> </tr> <tr> <td>10b</td> <td>Not required (i.e., not supported or supported but not required)</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	Not specified	01b	Required	10b	Not required (i.e., not supported or supported but not required)	11b	Reserved
	Bits	Description																															
	7:6	Reserved																															
	5:4	<p><b>Transport Authentication and Secure Channel (TASC):</b> Indicates whether an authentication transaction (refer to section 8.3.4.2) or an authentication transaction concatenated to a secure channel establishment (refer to section 8.3.4.3) is required upon completion of the Connect command. These bits do not override the AUTHREQ bits in the Connect response; these bits apply only when the AUTHREQ bits specify no requirements (refer to Figure 548).</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Not specified</td> </tr> <tr> <td>01b</td> <td>Authentication required</td> </tr> <tr> <td>10b</td> <td>Authentication concatenated to secure channel establishment required</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	Not specified	01b	Authentication required	10b	Authentication concatenated to secure channel establishment required	11b	Reserved																					
	Value	Definition																															
00b	Not specified																																
01b	Authentication required																																
10b	Authentication concatenated to secure channel establishment required																																
11b	Reserved																																
3	<p><b>Zero Host ID Support (ZHIDS):</b> If this bit is set to '1', then the controller supports a Host Identifier value of 0h in a Connect command. If this bit is cleared to '0', then the controller does not support a Host Identifier value of 0h in a Connect command.</p>																																
2	<p><b>SQ Flow Control Disable (SQFCD):</b> If this bit is set to '1', then the controller is capable of disabling SQ flow control. A controller that is capable of disabling SQ flow control may accept or reject a host request to disable SQ flow control. If this bit is cleared to '0', then the controller requires use of SQ flow control.</p>																																
1:0	<p><b>Transport Secure Channel (TSC):</b> Indicates whether connections shall be made over a fabric secure channel (refer to section 8.3.4.1).</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Not specified</td> </tr> <tr> <td>01b</td> <td>Required</td> </tr> <tr> <td>10b</td> <td>Not required (i.e., not supported or supported but not required)</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	Not specified	01b	Required	10b	Not required (i.e., not supported or supported but not required)	11b	Reserved																						
Value	Definition																																
00b	Not specified																																
01b	Required																																
10b	Not required (i.e., not supported or supported but not required)																																
11b	Reserved																																
05:04	<p><b>Port ID (PORTID):</b> Specifies a particular NVM subsystem port. Different NVMe Transports or address families may utilize the same Port ID value (e.g., a Port ID may support both iWARP and RoCE).</p>																																

Figure 294: Discovery Log Page Entry Data Structure

Bytes	Description										
07:06	<b>Controller ID (CNTLID):</b> Specifies the controller ID. If the NVM subsystem uses a dynamic controller model, then this field shall be set to FFFFh. If the NVM subsystem uses a static controller model, then this field may be set to a specific controller ID (values 0h to FFEFh are valid). If the NVM subsystem uses a static controller model and the value indicated is FFFEh, then the host should remember the Controller ID returned as part of the Fabrics Connect command in order to re-establish an association in the future with the same controller.										
09:08	<b>Admin Max SQ Size (ASQSZ):</b> Specifies the maximum size of an Admin Submission Queue. This applies to all controllers in the NVM subsystem. The value shall be a minimum of 32 entries.										
11:10	<b>Entry Flags (EFLAGS):</b> This field Indicates additional information related to the current entry.										
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:3</td> <td>Reserved</td> </tr> <tr> <td>2</td> <td><b>No CDC Connectivity (NCC):</b> If this bit is set to '1', then no DDC that describes this entry is currently connected to the CDC. If this bit is cleared to '0', then at least one DDC that describes this entry is currently connected to the CDC. If the Discovery controller returning this log page is not a CDC, then this bit shall be cleared to '0' and should be ignored by the host.</td> </tr> <tr> <td>1</td> <td><b>Explicit Persistent Connection Support for Discovery (EPCSD):</b> If this bit is set to '1', then Explicit Persistent Connections are supported for the Discovery controller described by this entry. If this bit is cleared to '0', then support for Explicit Persistent Connections is not reported.  For entries with the SUBTYPE field set to 2h, this bit shall be cleared to '0'.</td> </tr> <tr> <td>0</td> <td><b>Duplicate Returned Information (DUPRETINFO):</b> If this bit is set to '1', then using the content of this entry to access this Discovery Service returns the same information that is returned by using the content of other entries in this log page that also have this bit set to '1'. If this bit is cleared to '0', then using the content of this entry to access this Discovery Service may or may not return different information than is returned by using the content of any other entry in this log page.  For entries with the SUBTYPE field set to a value other than 3h, this bit shall be cleared to '0'.</td> </tr> </tbody> </table>	Bits	Description	15:3	Reserved	2	<b>No CDC Connectivity (NCC):</b> If this bit is set to '1', then no DDC that describes this entry is currently connected to the CDC. If this bit is cleared to '0', then at least one DDC that describes this entry is currently connected to the CDC. If the Discovery controller returning this log page is not a CDC, then this bit shall be cleared to '0' and should be ignored by the host.	1	<b>Explicit Persistent Connection Support for Discovery (EPCSD):</b> If this bit is set to '1', then Explicit Persistent Connections are supported for the Discovery controller described by this entry. If this bit is cleared to '0', then support for Explicit Persistent Connections is not reported.  For entries with the SUBTYPE field set to 2h, this bit shall be cleared to '0'.	0	<b>Duplicate Returned Information (DUPRETINFO):</b> If this bit is set to '1', then using the content of this entry to access this Discovery Service returns the same information that is returned by using the content of other entries in this log page that also have this bit set to '1'. If this bit is cleared to '0', then using the content of this entry to access this Discovery Service may or may not return different information than is returned by using the content of any other entry in this log page.  For entries with the SUBTYPE field set to a value other than 3h, this bit shall be cleared to '0'.
	Bits	Description									
	15:3	Reserved									
2	<b>No CDC Connectivity (NCC):</b> If this bit is set to '1', then no DDC that describes this entry is currently connected to the CDC. If this bit is cleared to '0', then at least one DDC that describes this entry is currently connected to the CDC. If the Discovery controller returning this log page is not a CDC, then this bit shall be cleared to '0' and should be ignored by the host.										
1	<b>Explicit Persistent Connection Support for Discovery (EPCSD):</b> If this bit is set to '1', then Explicit Persistent Connections are supported for the Discovery controller described by this entry. If this bit is cleared to '0', then support for Explicit Persistent Connections is not reported.  For entries with the SUBTYPE field set to 2h, this bit shall be cleared to '0'.										
0	<b>Duplicate Returned Information (DUPRETINFO):</b> If this bit is set to '1', then using the content of this entry to access this Discovery Service returns the same information that is returned by using the content of other entries in this log page that also have this bit set to '1'. If this bit is cleared to '0', then using the content of this entry to access this Discovery Service may or may not return different information than is returned by using the content of any other entry in this log page.  For entries with the SUBTYPE field set to a value other than 3h, this bit shall be cleared to '0'.										
31:12	Reserved										
63:32	<b>Transport Service Identifier (TRSVCID):</b> Specifies the NVMe Transport service identifier as an ASCII string. The NVMe Transport service identifier is specified by the associated NVMe Transport binding specification.										
255:64	Reserved										
511:256	<b>NVM Subsystem Qualified Name (SUBNQN):</b> NVMe Qualified Name (NQN) that uniquely identifies the NVM subsystem. Refer to section 4.7.  For a Discovery subsystem, if that Discovery subsystem has a unique Discovery Service NQN (i.e., the NVM Subsystem NVMe Qualified Name (SUBNQN) field in that Discovery subsystem's Identify Controller data structure contains a unique Discovery Service NQN value), then the value returned shall be that unique Discovery Service NQN. If the Discovery subsystem does not have a unique Discovery Service NQN, then the value returned shall be the well-known Discovery Service NQN (i.e., nqn.2014-08.org.nvmexpress.discovery).										
767:512	<b>Transport Address (TRADDR):</b> Specifies the address of the NVM subsystem that may be used for a Connect command as an ASCII string. The Address Family field describes the reference for parsing this field. Refer to section 1.4.2 for ASCII string requirements. For the definition of this field, refer to the appropriate NVMe Transport binding specification.										
1023:768	<b>Transport Specific Address Subtype (TSAS):</b> Specifies NVMe Transport specific information about the address. For the definition of this field, refer to the appropriate NVMe Transport binding specification.										

**Figure 295: Extended Discovery Log Page Entry**

Bytes	Description
<b>Header</b>	
00	<b>Transport Type (TRTYPE):</b> This field indicates the transport type as defined in the Transport Type (TRTYPE) field in Figure 294.
01	<b>Address Family (ADRFAM):</b> This field indicates the address family as defined in the Address Family (ADRFAM) field in Figure 294.
02	<b>Subsystem Type (SUBTYPE):</b> This field indicates the type of the NVM subsystem that is indicated in this entry as defined in the Subsystem Type (SUBTYPE) field in Figure 294.
03	<b>Transport Requirements (TREQ):</b> This field indicates requirements for the NVMe Transport as defined in the Transport Requirements (TREQ) field in Figure 294.
05:04	<b>Port ID (PORTID):</b> This field indicates a particular NVM subsystem port as defined in the Port ID (PORTID) field in Figure 294.
07:06	<b>Controller ID (CNTLID):</b> This field indicates the controller ID as defined in the Controller ID (CNTLID) field in Figure 294.
09:08	<b>Admin Max SQ Size (ASQSZ):</b> This field indicates the maximum size of an Admin Submission Queue as defined in the Admin Max SQ Size (ASQSZ) field in Figure 294.
11:10	<b>Entry Flags (EFLAGS):</b> This field indicates additional information related to the current entry as defined in the Entry Flags (EFLAGS) field in Figure 294.
31:12	Reserved
63:32	<b>Transport Service Identifier (TRSVCID):</b> This field indicates the NVMe Transport service identifier as an ASCII string as defined in the Transport Service Identifier (TRSVCID) field in Figure 294.
255:64	Reserved
511:256	<b>NVM Subsystem Qualified Name (NQN):</b> This field indicates the NVMe Qualified Name (NQN) that uniquely identifies the NVM subsystem as defined in the NVM Subsystem Qualified Name (SUBNQN) field in Figure 294.
767:512	<b>Transport Address (TRADDR):</b> This field indicates the address of the NVM subsystem that may be used for a Connect command as an ASCII string as defined in the Transport Address (TRADDR) field in Figure 294.
1023:768	<b>Transport Specific Address Subtype (TSAS):</b> This field indicates NVMe Transport specific information about the address as defined in the Transport Specific Address Subtype (TSAS) field in Figure 294.
1027:1024	<b>Total Entry Length (TEL):</b> This field indicates the length in bytes of the entire Extended Discovery Log Page Entry.
1029:1028	<b>Number of Extended Attributes (NUMEXAT):</b> This field indicates the number of extended attributes contained in the Extended Discovery Log Page Entry.
1031:1030	Reserved
<b>Extended Attribute List</b>	
((EXATLEN - 1) + 4) + 1032: 1032	<b>Extended Attribute 0:</b> This field contains the first extended attribute as defined in Figure 499 (if present), where EXATLEN is the size specified in the Extended Attribute Length (EXATLEN) field of the extended attribute.
...	...
TEL - 1: TEL - (EXATLEN + 4)	<b>Extended Attribute N:</b> This field contains the Nth extended attribute as defined in Figure 499 (if present), where EXATLEN is the size specified in the Extended Attribute Length (EXATLEN) field of the extended attribute and TEL is the size specified in the Total Entry Length (TEL) field.

Figure 296: Discovery Log Page

Bytes	Description	
<b>Header</b>		
07:00	<b>Generation Counter (GENCTR):</b> Indicates the version of the discovery information, starting at a value of 0h. For each change in the Discovery log page, this counter is incremented by one. If the value of this field is FFFFFFFF_FFFFFFFFh, then the field shall be cleared to 0h when incremented (i.e., rolls over to 0h).	
15:08	<b>Number of Records (NUMREC):</b> Indicates the number of records contained in the log page.	
17:16	<b>Record Format (RECFMT):</b> Specifies the format of the Discovery log page. If a new format is defined, this value is incremented by one. The format of the record specified in this definition shall be 0h.	
18	<b>Discovery Log Page Flags (DLPF):</b> This field indicates additional information related to the Discovery log page.	
	<b>Bits</b>	
	7:3	Reserved
	2	<b>All Subsystems (ALLSUB):</b> If this bit is set to '1', then the records contained in the log page are for all NVM subsystem ports. If this bit is cleared to '0', then the records contained in the log page may or may not be for all NVM subsystem ports.
	1	<b>Port Local (PORTLCL):</b> If this bit is set to '1', then the records contained in the log page are only for NVM subsystem ports that are presented through the same NVM subsystem port that is returning this log. If this bit is cleared to '0', then the records contained in the log page may be for NVM subsystem ports that are presented through any NVM subsystem port on that NVM subsystem.
0	<b>Extended (EXTEND):</b> If this bit is set to '1', then the records contained in the log page may include Extended Discovery Log Page Entries. If this bit is cleared to '0', then the records contained in the log page do not include Extended Discovery Log Page Entries.	
19	Reserved	
23:20	<b>Total Discovery Log Page Length (TDLPL):</b> This field indicates the length in bytes of the entire Discovery log page. If the value in this field is cleared to 0h, then this field is not reported.	
1023:24	Reserved	
<b>Discovery Log Page Entry List</b>		
(TEL - 1) + 1024: 1024	<b>Entry 0:</b> Contains either the first Discovery Log Page Entry as defined in Figure 294 or the first Extended Discovery Log Page Entry as defined in Figure 295. TEL is the size indicated in the Total Entry Length (TEL) field of the Extended Discovery Log Page Entry. For Discovery Log Page Entries, TEL shall be 1,024 bytes.	
...	...	
TDLPL - 1: TDLPL - TEL	<b>Entry N:</b> Contains either the Nth Discovery Log Page Entry as defined in Figure 294 (if present) or the Nth Extended Discovery Log Page Entry as defined in Figure 295 (if present). TEL is the size indicated in the Total Entry Length (TEL) field of the Extended Discovery Log Page Entry and TDLPL is the size indicated in the Total Discovery Log Page Length (TDLPL) field. For Discovery Log Page Entries, TEL shall be 1,024 bytes. If the TDLPL field is reserved (i.e., this log page does not contain any Extended Discovery Log Page Entries), then the byte position of the Nth Discovery Log Page Entry shall be: $((N + 2) * 1024) - 1 : ((N + 1) * 1024)$ .	

### 5.1.12.3.2 Host Discovery (Log Page Identifier 71h)

The Host Discovery log page shall only be supported by Discovery controllers. The Host Discovery log page shall not be supported by controllers that expose namespaces for NVMe over PCIe or NVMe over Fabrics. The Host Discovery log page provides an inventory of hosts that have registered discovery information with the Discovery controller. A host or a Discovery controller may obtain the host inventory from this log page. Depending on the parameters specified in a Get Log Page command, the Host Discovery log page may contain entries for:

- all known hosts; or

- a subset of entries specific to the host or Discovery controller requesting the log page.

Each Host Extended Discovery Log Page Entry shall contain at least one extended attribute containing a Host Identifier.

Figure 297 specifies the format for the LID Specific Parameter field in the Supported Log Pages log page (refer to section 5.1.12.1.1) for the Host Discovery log page.

**Figure 297: Host Discovery – LID Specific Parameter Field**

Bits	Description
15:1	Reserved
0	<b>All Host Entries Supported (ALLHOSTES):</b> If this bit is set to '1', then the Discovery controller supports returning records for all hosts that are registered without filtering the list of returned records based upon the requesting HOSTNQN. If this bit is cleared to '0', then the Discovery controller does not support returning records for all hosts that are registered.

If a Discovery controller supports returning records for all hosts that are registered (i.e., the All Host Entries Supported (ALLHOSTES) bit is set to '1' in the LID Specific Parameter field as defined in Figure 297) and that Discovery controller processes a Get Log Page command for this log page with the All Host Entries (ALLHOSTES) bit set to '1' in the Log Specific Parameter field (refer to Figure 298), then that Discovery controller shall return records for all hosts that are registered. The list of records returned by the Discovery controller shall not be filtered based upon the requesting HOSTNQN. If a Discovery controller returns records for all hosts that are registered, then that Discovery controller shall set the All Hosts (ALLHOST) bit to '1' in the Host Discovery Log Page Flags (HDLPF) field. A Discovery controller may require authentication before reporting the ALLHOSTES bit being set to '1' or may require configuration outside the scope of this specification.

The Log Page Offset may be used to retrieve specific records. If the Discovery controller supports an index offset for this log page (i.e., the Index Offset Supported (IOS) bit is set to '1' in the Supported Log Pages log page (refer to section 5.1.12.1.1) for this log page), then the index values specified in the Log Page Offset Lower (LPOL) field (refer to Figure 199) and Log Page Offset Upper (LPOU) field (refer to Figure 200) in the Get Log Page command shall be used to index into the header and list of entries within this log page (e.g., specifying an index offset of 0 returns this log page starting at the header, specifying an index offset of 1 returns this log page starting at the offset of Entry 0, etc.). The number of records is returned in the header of the log page. The format for a Host Extended Discovery Log Page Entry is defined in Figure 299. The format for an extended attribute is defined in Figure 499. The format for the Host Discovery log page is defined in Figure 300.

A single Get Log Page command used to read the Host Discovery log page shall be atomic. If the Host Discovery log page is read using multiple Get Log Page commands, then the reader should validate the Generation Counter field to ensure that there has not been a change in the contents of the data. The Host Discovery log page contents should be processed in the following sequence:

1. read the Host Discovery log page contents in order (i.e., with increasing Log Page Offset values);
2. re-read the Generation Counter field after the entire log page is transferred; and
3. if the Generation Counter field does not match the original value read, then discard the log page read, as the entries may be inconsistent, and restart the sequence; or
4. if the Generation Counter field does match the original value read, then process the returned data.

If the log page contents change during this command sequence, then the Discovery controller may return a status code of Discover Restart.

**Figure 298: Host Discovery Log Specific Parameter Field**

Bits	Description
14:9	Reserved
08	<b>All Host Entries (ALLHOSTE):</b> If this bit is set to '1', then the Discovery controller shall return records for all hosts that are registered without filtering the list of returned records based upon the requesting HOSTNQN. If this bit is cleared to '0', then this bit has no effect.

Figure 299: Host Extended Discovery Log Page Entry

Bytes	Description	
<b>Header</b>		
00	<b>Transport Type (TRTYPE):</b> This field indicates the transport type as defined in the Transport Type (TRTYPE) field in Figure 294.	
01	<b>Address Family (ADRFAM):</b> This field indicates the address family as defined in the Address Family (ADRFAM) field in Figure 294.	
09:02	Reserved	
11:10	<b>Entry Flags (EFLAGS):</b> This field indicates additional information related to the current entry.	
	<b>Bits</b> <b>Description</b>	
	15:3	Reserved
	2	<b>No CDC Connectivity (NCC):</b> If this bit is set to '1', then the host that describes this entry is not currently connected to the CDC. If this bit is cleared to '0', then the host that describes this entry is currently connected to the CDC. If the Discovery controller returning this log page is not a CDC, then this bit shall be cleared to '0' and should be ignored by the host.
1:0	Reserved	
255:12	Reserved	
511:256	<b>Host NVMe Qualified Name (HOSTNQN):</b> This field indicates the NVMe Qualified Name (NQN) that uniquely identifies the host. Refer to section 4.7 for the formatting requirements for NQNs.	
767:512	<b>Transport Address (TRADDR):</b> This field indicates the address of the host that may be used for a Connect command as an ASCII string. The Address Family field describes the reference for parsing this field. Refer to section 1.4.2 for ASCII string requirements. For the definition of this field, refer to the appropriate NVMe Transport binding specification.	
1023:768	<b>Transport Specific Address Subtype (TSAS):</b> This field indicates NVMe Transport specific information about the address as defined in the Transport Specific Address Subtype (TSAS) field in Figure 294.	
1027:1024	<b>Total Entry Length (TEL):</b> This field indicates the length in bytes of the entire Host Extended Discovery Log Page Entry.	
1029:1028	<b>Number of Extended Attributes (NUMEXAT):</b> This field indicates the number of extended attributes contained in the Host Extended Discovery Log Page Entry. This field shall be non-zero (i.e., the Host Extended Discovery Log Page Entry contains at least one extended attribute containing a Host Identifier as described in this section).	
1031:1030	Reserved	
<b>Extended Attributes List</b>		
((EXATLEN - 1) + 4) + 1032:1032	<b>Extended Attribute 0:</b> This field contains the first extended attribute as defined in Figure 499, where EXATLEN is the size specified in the Extended Attribute Length (EXATLEN) field of the extended attribute.	
...	...	
TEL - 1:TEL - (EXATLEN + 4)	<b>Extended Attribute N:</b> This field contains the Nth extended attribute as defined in Figure 499 (if present), where EXATLEN is the size specified in the Extended Attribute Length (EXATLEN) field of the extended attribute and TEL is the size specified in the Total Entry Length (TEL) field.	

Figure 300: Host Discovery Log Page

Bytes	Description
<b>Header</b>	
07:00	<b>Generation Counter (GENCTR):</b> This field indicates the version of the discovery information, starting at a value of 0h. For each change in the Host Discovery log page, this counter is incremented by one. If the value of this field is FFFFFFFF_FFFFFFFFh, then the field shall be cleared to 0h when incremented (i.e., rolls over to 0h).
15:08	<b>Number of Records (NUMREC):</b> Indicates the number of records contained in the log page.

**Figure 300: Host Discovery Log Page**

Bytes	Description						
17:16	<b>Record Format (RECFMT):</b> This field indicates the format of the Host Discovery log page. If a new format is defined, this value is incremented by one. The format of the record specified in this definition shall be 0h.						
18	<p><b>Host Discovery Log Page Flags (HDLPF):</b> This field indicates additional information related to the Host Discovery log page.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td><b>All Hosts (ALLHOST):</b> If this bit is set to '1', then the records contained in the log page are for all hosts. If this bit is cleared to '0', then the records contained in the log page may or may not be for all hosts.</td> </tr> </tbody> </table>	Bits	Description	7:1	Reserved	0	<b>All Hosts (ALLHOST):</b> If this bit is set to '1', then the records contained in the log page are for all hosts. If this bit is cleared to '0', then the records contained in the log page may or may not be for all hosts.
Bits	Description						
7:1	Reserved						
0	<b>All Hosts (ALLHOST):</b> If this bit is set to '1', then the records contained in the log page are for all hosts. If this bit is cleared to '0', then the records contained in the log page may or may not be for all hosts.						
19	Reserved						
23:20	<b>Total Host Discovery Log Page Length (THDLPL):</b> This field indicates the length in bytes of the entire Host Discovery log page.						
1023:24	Reserved						
Host Extended Discovery Log Page Entry List							
(TEL - 1) + 1024: 1024	<b>Host Extended Discovery Log Page Entry 0:</b> This field contains the first Host Extended Discovery Log Page Entry as defined in Figure 299. TEL is the size indicated in the Total Entry Length (TEL) field of the Host Extended Discovery Log Page Entry.						
...	...						
THDLPL - 1: THDLPL – TEL	<b>Host Extended Discovery Log Page Entry N:</b> This field contains the Nth Host Extended Discovery Log Page Entry as defined in Figure 299 (if present). TEL is the size indicated in the Total Entry Length (TEL) field of the Host Extended Discovery Log Page Entry and THDLPL is the size indicated in the Total Host Discovery Log Page Length (THDLPL) field.						

**5.1.12.3.3 AVE Discovery Log Page (Log Page Identifier 72h)**

The format of the AVE Discovery Log Page is shown in Figure 301.

**Figure 301: Get Log Page – AVE Discovery Log Page**

Bytes	Description
Header	
07:00	<b>Generation Counter (GENCTR):</b> This field indicates the version of the discovery information, starting at a value of 0h. For each change in the AVE Discovery log page, this field shall be incremented by one. If the value of this field is FFFFFFFF_FFFFFFFFh, then the field shall be cleared to 0h when incremented (i.e., rolls over to 0h).
15:08	<b>Number of Records (NUMREC):</b> Indicates the number of records contained in the log page.
17:16	<b>Record Format (RECFMT):</b> This field indicates the format of the AVE Discovery log page. If a new format is defined, this value is incremented by one. The format of the record specified in this definition shall be 0h.
19:18	Reserved
23:20	<b>Total AVE Discovery Log Page Length (TADLPL):</b> This field indicates the length in bytes of the entire AVE Discovery log page.
1023:24	Reserved
AVE Discovery Log Page Entry List	
TEL + 1023:1024	<b>AVE Discovery Log Page Entry 0:</b> This field contains the first AVE Discovery Log Page Entry as defined in Figure 302. TEL is the size indicated in the Total Entry Length (TEL) field of the AVE Discovery Log Page Entry.
...	...
TADLPL - 1: TADLPL – TEL	<b>AVE Discovery Log Page Entry NUMREC-1:</b> This field contains the NUMREC-1 AVE Discovery Log Page Entry as defined in Figure 302 (if present). TEL is the size indicated in the Total Entry Length (TEL) field of the AVE Discovery Log Page Entry and TADLPL is the size indicated in the Total AVE Log Page Length (TADLPL) field.

The format of the AVE Discovery Log Page Entry is shown in Figure 302.

**Figure 302: Get Log Page – AVE Discovery Log Page Entry**

Bytes	Description
<b>Header</b>	
03:00	<b>Total Entry Length (TEL):</b> This field indicates the length in bytes of the AVE Discovery Log Page Entry.
227:04	<b>AVE NQN (AVENQN):</b> This field indicates the NQN (represented as a null-terminated string, NULL padded as necessary to the 224-byte maximum length) of the AVE.
228	<b>Number of AVE Transport Records (NUMATR):</b> This field indicates the number of subsequent AVE transport records.
231:229	Reserved
<b>AVE Transport Record List</b>	
251:232	<b>AVE Transport Record 1:</b> The first AVE Transport Record (refer to Figure 303), if any.
271:252	<b>AVE Transport Record 2:</b> The second AVE Transport Record (refer to Figure 303), if any.
...	...
(NUMATR*20)+231: (NUMATR*20)+212	<b>AVE Transport Record NUMATR:</b> The last AVE Transport Record (refer to Figure 303), if any.

The format of the AVE Transport Record is shown in Figure 303.

**Figure 303: AVE Transport Record**

Bytes	Description										
00	<b>AVE Address Family (AVEADRFAM):</b> This field identifies the IP address family. This field shall be set to one of the following values: <ul style="list-style-type: none"> <li>• 1h: IPv4 address family; or</li> <li>• 2h: IPv6 address family.</li> </ul>										
01	Reserved										
03:02	<b>AVE Transport Service Identifier (AVETR SVCID):</b> This field identifies the TCP port.										
19:04	<b>AVE Transport Address (AVETRADDR):</b> This field identifies the IP address. An IPv6 address is encoded in binary as follows: <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Bytes</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:00</td> <td><b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.</td> </tr> </tbody> </table> <p>An IPv4 address is encoded in binary as follows:</p> <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Bytes</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>03:00</td> <td><b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.</td> </tr> <tr> <td>15:04</td> <td>Reserved</td> </tr> </tbody> </table>	Bytes	Description	15:00	<b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.	Bytes	Description	03:00	<b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.	15:04	Reserved
	Bytes	Description									
	15:00	<b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.									
	Bytes	Description									
	03:00	<b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.									
15:04	Reserved										

#### 5.1.12.3.4 Pull Model DDC Request Log Page (Log Page Identifier 73h)

The format of the Pull Model DDC Request log page is shown in Figure 304.



**Figure 304: Pull Model DDC Request Log Page**

Bytes	Description														
<b>Header</b>															
00	<p><b>Operation Request Identifier (ORI):</b> This field indicates the operation that a pull model DDC is requesting a CDC to perform. Defined values are:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Reserved</td> </tr> <tr> <td>1</td> <td>Get Active ZoneGroup (GAZ) operation</td> </tr> <tr> <td>2</td> <td>Add or Replace Active ZoneGroup (AAZ) operation</td> </tr> <tr> <td>3</td> <td>Remove Active ZoneGroup (RAZ) operation</td> </tr> <tr> <td>4</td> <td>Discovery Log Page Request operation</td> </tr> <tr> <td>All others</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0	Reserved	1	Get Active ZoneGroup (GAZ) operation	2	Add or Replace Active ZoneGroup (AAZ) operation	3	Remove Active ZoneGroup (RAZ) operation	4	Discovery Log Page Request operation	All others	Reserved
Value	Definition														
0	Reserved														
1	Get Active ZoneGroup (GAZ) operation														
2	Add or Replace Active ZoneGroup (AAZ) operation														
3	Remove Active ZoneGroup (RAZ) operation														
4	Discovery Log Page Request operation														
All others	Reserved														
03:01	Reserved														
07:04	<b>Total Pull Model DDC Request Log Page Length (TPDRPL):</b> This field indicates the length in bytes of the entire Pull Model DDC Request log page.														
TPDRPL-1:08	<b>Operation Specific Parameters (OSP)</b>														

### 5.1.12.4 Command Completion

Upon completion of the Get Log Page command, the controller posts a completion queue entry to the Admin Completion Queue. Get Log Page command specific status values are defined in Figure 305.

**Figure 305: Get Log Page – Command Specific Status Values**

Value	Description
9h	<b>Invalid Log Page:</b> The log page indicated is invalid or not supported. This error condition is also returned if a reserved log page is requested. Controllers compliant with NVM Express Base Specification revision 2.0 and earlier may return Invalid Field in Command for this condition.
29h	<b>I/O Command Set Not Supported:</b> The specified I/O Command Set is not supported by the controller.

### 5.1.13 Identify command

#### 5.1.13.1 Identify command overview

The Identify command returns a data buffer that describes information about the NVM subsystem, the domain, the controller or the namespace(s). The data structure is 4,096 bytes in size.

The Identify command uses the Data Pointer, Command Dword 10, Command Dword 11, and Command Dword 14 fields. All other command specific fields are reserved.

**Figure 306: Identify – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field. If using PRPs, this field shall not be a pointer to a PRP List as the data buffer may not cross more than one page boundary.

**Figure 307: Identify – Command Dword 10**

Bits	Description
31:16	<p><b>Controller Identifier (CNTID):</b> This field specifies the controller identifier used as part of some Identify operations. Whether the CNTID field is used for a particular Identify operation is indicated in Figure 310. If this field is not used as part of the Identify operation, then:</p> <ul style="list-style-type: none"> <li>host software shall clear this field to 0h for backwards compatibility (0h is a valid controller identifier); and</li> <li>the controller shall ignore this field.</li> </ul> <p>Controllers that support the Namespace Management capability (refer to section 8.1.15) shall support this field.</p>
15:08	Reserved
07:00	<b>Controller or Namespace Structure (CNS):</b> This field specifies the information to be returned to the host. Refer to Figure 310.

**Figure 308: Identify – Command Dword 11**

Bits	Description
31:24	<p><b>Command Set Identifier (CSI):</b> This field is CNS value specific. This field specifies the I/O Command Set to be used by the command for CNS values that require a Command Set Identifier. Refer to Figure 310 for Identify command CNS values that use this field. This field shall be cleared to 0h for Identify operations with CNS values that do not use this field.</p> <p>Values for this field are defined by Figure 311.</p>
23:16	Reserved
15:00	<b>CNS Specific Identifier (CNSSID):</b> This field is dependent on the specified CNS value and if not defined by that CNS value, then this field is reserved.

If the controller supports selection of a UUID by the Identify command (refer to section 8.1.28), then Command Dword 14 is used to specify a UUID Index value (refer to Figure 309).

**Figure 309: Identify – Command Dword 14**

Bits	Description
31:07	Reserved
06:00	<b>UUID Index (UIDX):</b> Refer to Figure 658.

The data structure returned is based on the Controller or Namespace Structure (CNS) field as shown in Figure 310. If there are fewer entries to return for the data structure indicated based on CNS value, then the unused portion of the returned data is zero filled. If a controller does not support the specified CNS value, then the controller shall abort the command with a status code of Invalid Field in Command.

When issuing the Identify command, if the specified namespace is not associated with an I/O Command Set that supports the specified Identify CNS value (refer to Figure 310), then the controller shall abort the command with a status code of Invalid I/O Command Set.

Note: The CNS field was specified as a one bit field in revision 1.0 and is a two bit field in revision 1.1. Host software should only issue CNS values defined in revision 1.0 to controllers compliant with revision 1.0. Host software should only issue CNS values defined in revision 1.1 to controllers compliant with revision 1.1. The results of issuing other CNS values to controllers compliant with revision 1.0 or revision 1.1, respectively, are indeterminate.

The Identify Controller data structure, Identify Namespace data structure, and the I/O Command Set specific Identify Namespace data structure include several unique identifiers. The format and layout of these unique identifiers is described in section 4.7.1.

Section 5.1.13.2 describes Identify data structures that are common to all transport models. Section 5.1.13.3 describes Identify data structures that are specific to the Memory-based transport model. Section 5.1.13.4 describes Identify data structures that are specific to the Message-based transport model.

**Figure 310: Identify – CNS Values**

CNS Value	O/M <sup>1</sup>	Definition	NSID <sup>2</sup>	CNTID <sup>3</sup>	CSI <sup>4</sup>	Reference Section
<b>Active Namespace Management</b>						
00h	M <sup>11</sup>	Identify Namespace data structure for the specified NSID or the namespace capabilities for the NVM Command Set. <sup>7</sup>	Y	N	N <sup>8</sup>	NVM Command Set Specification
01h	M	Identify Controller data structure for the controller processing the command. <sup>7</sup>	N	N	N	5.1.13.2.1
02h	M	Active Namespace ID list.	Y	N	N	5.1.13.2.2
03h	M	Namespace Identification Descriptor list for the specified NSID.	Y	N	N	5.1.13.2.3
04h	O	An NVM Set List (refer to Figure 317) is returned to the host for up to 31 NVM Sets. The list contains entries for NVM Set identifiers greater than or equal to the value specified in the NVM Set Identifier (CDW11.NVMSETID) field.	N	N	N	5.1.13.2.4
05h	M	I/O Command Set specific Identify Namespace data structure for the specified NSID for the I/O Command Set specified in the CSI field. <sup>7</sup>	Y	N	Y	5.1.13.2.5
06h	M	I/O Command Set specific Identify Controller data structure for the controller processing the command. <sup>7</sup>	N	N	Y	5.1.13.2.6
07h	M	Active Namespace ID list associated with the specified I/O Command Set.	Y	N	Y	5.1.13.2.7
08h	M	I/O Command Set Independent Identify Namespace data structure.	Y	N	N	5.1.13.2.8
09h	O	Identify Namespace data structure for the specified Format Index containing the namespace capabilities for the NVM Command Set. <sup>7</sup>	N	N	Y	NVM Command Set Specification
0Ah	O	I/O Command Set specific Identify Namespace data structure for the specified Format Index containing the namespace capabilities for the I/O Command Set specified in the CSI field. <sup>7</sup>	N	N	Y	I/O Command Set Specification
0Bh to 0Fh		Reserved				
<b>Controller and Namespace Management</b>						
10h	O <sup>5</sup>	Allocated Namespace ID list.	Y	N	N	5.1.13.2.9
11h	O <sup>5, 11</sup>	Identify Namespace data structure for the specified allocated NSID.	Y	N	N <sup>8</sup>	5.1.13.2.10
12h	O <sup>5</sup>	Controller List of controllers attached to the specified NSID.	Y	Y	N	5.1.13.2.11
13h	O <sup>5</sup>	Controller List of controllers that exist in the NVM subsystem.	N	Y	N	5.1.13.2.12
14h	O <sup>6</sup>	Primary Controller Capabilities data structure for the specified primary controller.	N	Y	N	5.1.13.3.1
15h	O <sup>6</sup>	Secondary Controller list of controllers associated with the primary controller processing the command.	N	Y	N	5.1.13.3.2
16h	O <sup>11</sup>	A Namespace Granularity List (refer to the NVM Command Set Specification) is returned to the host for up to sixteen Namespace Granularity Entries.	N	N	N <sup>8</sup>	5.1.13.2.13
17h	O	A UUID List (refer to Figure 320) is returned to the host.	N	N	N	5.1.13.2.14

Figure 310: Identify – CNS Values

CNS Value	O/M <sup>1</sup>	Definition	NSID <sup>2</sup>	CNTID <sup>3</sup>	CSI <sup>4</sup>	Reference Section
18h	O <sup>10</sup>	Domain List	N	N	N	5.1.13.2.15
19h	O <sup>9</sup>	Endurance Group List	N	N	N	5.1.13.2.16
1Ah	O <sup>5</sup>	I/O Command Set specific Allocated Namespace ID list	Y	N	Y	5.1.13.2.17
1Bh	O <sup>5</sup>	I/O Command Set specific Identify Namespace data structure for the specified allocated NSID.	Y	N	Y	5.1.13.2.18
1Ch	O	I/O Command Set data structure	N	Y	N	5.1.13.2.19
1Dh	O	Get Underlying Namespace List <sup>12</sup>	Y	Y	N	5.1.13.4.1
1Eh	O	Get Ports List	N	N	N	5.1.13.4.2
1Fh	O <sup>13</sup>	I/O Command Set Independent Identify Namespace data structure for the specified allocated NSID.	Y	N	N	5.1.13.2.20
20h	O	Supported Controller State Formats	N	N	N	5.1.13.2.21
<b>Future Definition</b>						
21h to FFh		Reserved				
Notes:						
1. O/M definition: O = Optional, M = Mandatory.						
2. The NSID field is used: Y = Yes, N = No.						
3. The CDW10.CNTID field is used: Y = Yes, N = No.						
4. The CDW11.CSI field is used: Y = Yes, N = No.						
5. Mandatory for controllers that support the Namespace Management capability (refer to section 8.1.15).						
6. Mandatory for controllers that support Virtualization Enhancements (refer to section 8.2.6).						
7. Selection of a UUID may be supported (refer to section 8.1.28).						
8. This Identify data structure applies to namespaces that are associated with command sets that specify logical blocks (e.g., Command Set Identifier 0h or Command Set Identifier 2h).						
9. Mandatory for controllers that support Variable Capacity Management (refer to section 8.1.4.3).						
10. Mandatory for controllers that support Capacity Management (refer to section 8.1.4) in an NVM subsystem that supports multiple domains (refer to section 3.2.5).						
11. Only applicable for the NVM Command Set and I/O Command Sets based on the NVM Command Set. Prohibited for all other I/O Command Sets.						
12. Support for this CNS value is prohibited in NVM subsystems that use a Memory-Based Transport Model (e.g., the PCIe transport) for any controller.						
13. For controllers compliant with NVM Express Base Specification, Revision TBD and later, mandatory if the Namespace Management capability is supported.						

The Command Set Identifier values are defined in Figure 311.

Figure 311: Command Set Identifiers

Command Set Identifier Value	Definition	Reference
00h	NVM Command Set	Refer to the NVM Command Set Specification
01h	Key Value Command Set	Refer to the Key Value Command Set Specification
02h	Zoned Namespace Command Set	Refer to the Zoned Namespace Command Set Specification
03h	Subsystem Local Memory Command Set	Refer to the Subsystem Local Memory Command Set Specification
04h	Computational Programs Command Set	Refer to the Computational Programs Command Set Specification
05h to 2Fh	Reserved	
30h to 3Fh	Vendor specific	
40h to FFh	Reserved	

### 5.1.13.2 Common Identify Data Structures

#### 5.1.13.2.1 Identify Controller Data Structure (CNS 01h)

The Identify Controller data structure (refer to Figure 312) is returned to the host for the controller processing the command.

**Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent**

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description												
<b>Controller Capabilities and Features</b>																
01:00	M	M	R	<b>PCI Vendor ID (VID):</b> Contains the company vendor identifier that is assigned by the PCI SIG. This is the same value as reported in the ID register in the PCI Header section of the NVMe over PCIe Transport Specification.												
03:02	M	M	R	<b>PCI Subsystem Vendor ID (SSVID):</b> Contains the company vendor identifier that is assigned by the PCI SIG for the subsystem. This is the same value as reported in the SS register in the PCI Header section of the NVMe over PCIe Transport Specification.												
23:04	M	M	O <sup>5</sup>	<b>Serial Number (SN):</b> Contains the serial number for the NVM subsystem that is assigned by the vendor as an ASCII string. Refer to section 1.4.2 for ASCII string requirements. For a Discovery subsystem, this field should not be used to construct a unique identifier. For subsystems that are not Discovery subsystems, refer to section 4.7.1 for unique identifier requirements.												
63:24	M	M	O <sup>5</sup>	<b>Model Number (MN):</b> Contains the model number for the NVM subsystem that is assigned by the vendor as an ASCII string. Refer to section 1.4.2 for ASCII string requirements. For a Discovery subsystem, this field should not be used to construct a unique identifier. For subsystems that are not Discovery subsystems, refer to section 4.7.1 for unique identifier requirements.												
71:64	M	M	M	<b>Firmware Revision (FR):</b> Contains the currently active firmware revision, as an ASCII string, for the domain of which this controller is a part. This is the same revision information that may be retrieved with the Get Log Page command, refer to section 5.1.12.1.4.												
72	M	M	R	<b>Recommended Arbitration Burst (RAB):</b> This is the recommended Arbitration Burst size. The value is in commands and is reported as a power of two (2 <sup>n</sup> ). This is the same units as the Arbitration Burst size. Refer to section 3.4.4.												
75:73	M	M	R	<b>IEEE OUI Identifier (IEEE):</b> Contains the Organization Unique Identifier (OUI) for the controller vendor. The OUI shall be a valid IEEE/RAC assigned identifier that may be registered at <a href="http://standards.ieee.org/develop/regauth/oui/public.html">http://standards.ieee.org/develop/regauth/oui/public.html</a> .												
76	O	O	R	<b>Controller Multi-Path I/O and Namespace Sharing Capabilities (CMIC):</b> This field specifies multi-path I/O and namespace sharing capabilities of the controller and NVM subsystem.												
				<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:4</td> <td>Reserved</td> </tr> <tr> <td>3</td> <td><b>Asymmetric Namespace Access Reporting Support (ANARS):</b> If this bit is set to '1', then the NVM subsystem supports Asymmetric Namespace Access Reporting (refer to section 8.1.1). If this bit is cleared to '0', then the NVM subsystem does not support Asymmetric Namespace Access Reporting.</td> </tr> <tr> <td>2</td> <td><b>Function Type (FT):</b> If this bit is set to '1', then the controller is associated with an SR-IOV Virtual Function. If this bit is cleared to '0', then the controller is associated with a PCI Function or a Fabrics connection.</td> </tr> <tr> <td>1</td> <td><b>Multiple Controllers (MCTRS):</b> If this bit is set to '1', then the NVM subsystem may contain two or more controllers. If this bit is cleared to '0', then the NVM subsystem contains only a single controller. As described in section 2.4.1, an NVM subsystem that contains multiple controllers may be used by multiple hosts, or may provide multiple paths for a single host.</td> </tr> <tr> <td>0</td> <td><b>Multiple Ports (MPORTS):</b> If this bit is set to '1', then the NVM subsystem may contain more than one NVM subsystem port. If this bit is cleared to '0', then the NVM subsystem contains only a single NVM subsystem port.</td> </tr> </tbody> </table>	Bits	Description	7:4	Reserved	3	<b>Asymmetric Namespace Access Reporting Support (ANARS):</b> If this bit is set to '1', then the NVM subsystem supports Asymmetric Namespace Access Reporting (refer to section 8.1.1). If this bit is cleared to '0', then the NVM subsystem does not support Asymmetric Namespace Access Reporting.	2	<b>Function Type (FT):</b> If this bit is set to '1', then the controller is associated with an SR-IOV Virtual Function. If this bit is cleared to '0', then the controller is associated with a PCI Function or a Fabrics connection.	1	<b>Multiple Controllers (MCTRS):</b> If this bit is set to '1', then the NVM subsystem may contain two or more controllers. If this bit is cleared to '0', then the NVM subsystem contains only a single controller. As described in section 2.4.1, an NVM subsystem that contains multiple controllers may be used by multiple hosts, or may provide multiple paths for a single host.	0	<b>Multiple Ports (MPORTS):</b> If this bit is set to '1', then the NVM subsystem may contain more than one NVM subsystem port. If this bit is cleared to '0', then the NVM subsystem contains only a single NVM subsystem port.
				Bits	Description											
				7:4	Reserved											
				3	<b>Asymmetric Namespace Access Reporting Support (ANARS):</b> If this bit is set to '1', then the NVM subsystem supports Asymmetric Namespace Access Reporting (refer to section 8.1.1). If this bit is cleared to '0', then the NVM subsystem does not support Asymmetric Namespace Access Reporting.											
2	<b>Function Type (FT):</b> If this bit is set to '1', then the controller is associated with an SR-IOV Virtual Function. If this bit is cleared to '0', then the controller is associated with a PCI Function or a Fabrics connection.															
1	<b>Multiple Controllers (MCTRS):</b> If this bit is set to '1', then the NVM subsystem may contain two or more controllers. If this bit is cleared to '0', then the NVM subsystem contains only a single controller. As described in section 2.4.1, an NVM subsystem that contains multiple controllers may be used by multiple hosts, or may provide multiple paths for a single host.															
0	<b>Multiple Ports (MPORTS):</b> If this bit is set to '1', then the NVM subsystem may contain more than one NVM subsystem port. If this bit is cleared to '0', then the NVM subsystem contains only a single NVM subsystem port.															

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description														
77	M	M	M	<p><b>Maximum Data Transfer Size (MDTS):</b> This field indicates the maximum data transfer size for a command that transfers data between host-accessible memory (refer to section 1.5.44) and the controller. The host should not submit a command that exceeds this maximum data transfer size. If a command is submitted that exceeds this transfer size, then the command is aborted with a status code of Invalid Field in Command. The value is in units of the minimum memory page size (CAP.MPSMIN) and is reported as a power of two (2<sup>n</sup>). A value of 0h indicates that there is no maximum data transfer size.</p> <p>If the MEM bit is cleared to '0' in the CTRATT field, then this field includes the length of metadata, if metadata is interleaved with the user data.</p> <p>If the MEM bit is set to '1', then this field excludes the length of metadata.</p> <p>This field does not apply to commands that do not transfer data between host-accessible memory and the controller (e.g., the Verify command, the Write Uncorrectable command, and the Write Zeroes command); refer to the ONCS field for restrictions on these commands and other commands that transfer data.</p> <p>If SGL Bit Bucket descriptors are supported, their lengths shall be included in determining if a command exceeds the Maximum Data Transfer Size for destination data buffers. Their length in a source data buffer is not included for a Maximum Data Transfer Size calculation.</p>														
79:78	M	M	M	<b>Controller ID (CNTLID):</b> Contains the NVM subsystem unique controller identifier associated with the controller.														
83:80	M	M	M	<b>Version (VER):</b> This field contains the value reported in the Version property (i.e., VS property) defined in section 3.1.4.2. Implementations compliant with NVM Express Base Specification, Revision 1.2 or later shall report a non-zero value in this field.														
87:84	M	M	R	<b>RTD3 Resume Latency (RTD3R):</b> This field indicates the expected latency in microseconds to resume from Runtime D3 (RTD3). Refer to section 8.1.17.4. A value of 0h indicates RTD3 Resume Latency is not reported.														
91:88	M	M	R	<b>RTD3 Entry Latency (RTD3E):</b> This field indicates the typical latency in microseconds to enter Runtime D3 (RTD3). Refer to section 8.1.17.4. A value of 0h indicates RTD3 Entry Latency is not reported.														
95:92	M	M	M	<p><b>Optional Asynchronous Events Supported (OAES):</b> This field indicates the optional asynchronous events supported by the controller. A controller shall not send optional asynchronous events before they are enabled by host software.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>31</td> <td><b>Discovery Log Page Change Notification (DLPCN):</b> If this bit is set to '1', then the controller supports sending Discovery Log Page Change Notifications. If this bit is cleared to '0', then the controller does not support the Discovery Log Page Change Notification events.</td> </tr> <tr> <td>30:28</td> <td>Reserved</td> </tr> <tr> <td>27</td> <td><b>Zone Descriptor Changed Notices (ZDCN):</b> If this bit is set to '1', then the controller supports the Zone Descriptor Changed Notices event and the associated Changed Zone List log page (refer to the Zoned Namespace Command Set specification). If this bit is cleared to '0', then the controller does not support the Zone Descriptor Changed Notices event nor the associated Changed Zone List log page.</td> </tr> <tr> <td>26:20</td> <td>Reserved</td> </tr> <tr> <td>19</td> <td><b>Allocated Namespace Attribute Notices (ANSAN):</b> If this bit is set to '1', then the controller supports the Allocated Namespace Attribute Notices event and the associated Changed Allocated Namespace List log page. If this bit is cleared to '0', then the controller does not support the Allocated Namespace Attribute Notices event nor the associated Changed Allocated Namespace List log page.</td> </tr> <tr> <td>18</td> <td>Reserved</td> </tr> </tbody> </table>	Bits	Description	31	<b>Discovery Log Page Change Notification (DLPCN):</b> If this bit is set to '1', then the controller supports sending Discovery Log Page Change Notifications. If this bit is cleared to '0', then the controller does not support the Discovery Log Page Change Notification events.	30:28	Reserved	27	<b>Zone Descriptor Changed Notices (ZDCN):</b> If this bit is set to '1', then the controller supports the Zone Descriptor Changed Notices event and the associated Changed Zone List log page (refer to the Zoned Namespace Command Set specification). If this bit is cleared to '0', then the controller does not support the Zone Descriptor Changed Notices event nor the associated Changed Zone List log page.	26:20	Reserved	19	<b>Allocated Namespace Attribute Notices (ANSAN):</b> If this bit is set to '1', then the controller supports the Allocated Namespace Attribute Notices event and the associated Changed Allocated Namespace List log page. If this bit is cleared to '0', then the controller does not support the Allocated Namespace Attribute Notices event nor the associated Changed Allocated Namespace List log page.	18	Reserved
Bits	Description																	
31	<b>Discovery Log Page Change Notification (DLPCN):</b> If this bit is set to '1', then the controller supports sending Discovery Log Page Change Notifications. If this bit is cleared to '0', then the controller does not support the Discovery Log Page Change Notification events.																	
30:28	Reserved																	
27	<b>Zone Descriptor Changed Notices (ZDCN):</b> If this bit is set to '1', then the controller supports the Zone Descriptor Changed Notices event and the associated Changed Zone List log page (refer to the Zoned Namespace Command Set specification). If this bit is cleared to '0', then the controller does not support the Zone Descriptor Changed Notices event nor the associated Changed Zone List log page.																	
26:20	Reserved																	
19	<b>Allocated Namespace Attribute Notices (ANSAN):</b> If this bit is set to '1', then the controller supports the Allocated Namespace Attribute Notices event and the associated Changed Allocated Namespace List log page. If this bit is cleared to '0', then the controller does not support the Allocated Namespace Attribute Notices event nor the associated Changed Allocated Namespace List log page.																	
18	Reserved																	

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description				
				17 <b>Reachability Groups Change Notices Support (RGCNS):</b> If this bit is set to '1', then the controller supports the Reachability Groups Change Notices event, and the Reachability Association Change Notices event. If this bit is cleared to '0', then the controller does not support the Reachability Groups Change Notices event, nor the Reachability Association Change Notices event.				
				16 <b>Temperature Threshold Hysteresis Recovery (TTHR):</b> If this bit is set to '1', then the controller supports the Temperature Threshold Hysteresis Recovery event. If this bit is cleared to '0', then the controller does not support the Temperature Threshold Hysteresis Recovery event. No log page is associated with this event.				
				15 <b>Normal NVM Subsystem Shutdown (NNSS):</b> If this bit is set to '1', then the controller supports the Normal NVM Subsystem Shutdown event. If this bit is cleared to '0', then the controller does not support the Normal NVM Subsystem Shutdown event. No log page is associated with this event.				
				14 <b>Endurance Group Event Aggregate Log Page Change Notices (EGEAN):</b> If this bit is set to '1', then the controller supports the Endurance Group Event Aggregate Log Page Change Notices event. If this bit is cleared to '0', then the controller does not support the Endurance Group Event Aggregate Log Page Change Notices event.				
				13 <b>LBA Status Information Alert Notices (LSIAN):</b> If this bit is set to '1', then the controller supports the LBA Status Information Alert Notices event (refer to the NVM Command Set Specification). If this bit is cleared to '0', then the controller does not support the LBA Status Information Alert Notices event.				
				12 <b>Predictable Latency Event Aggregate Log Change Notices (PLEAN):</b> If this bit is set to '1', then controller supports the Predictable Latency Event Aggregate Log Change Notices event. If this bit is cleared to '0', then the controller does not support the Predictable Latency Event Aggregate Log Change Notices event.				
				11 <b>Asymmetric Namespace Access Change Notices (ANACN):</b> If this bit is set to '1', then the controller supports sending Asymmetric Namespace Access Change Notices. If this bit is cleared to '0', then the controller does not support the Asymmetric Namespace Access Change Notices event.				
				10 Reserved				
				9 <b>Firmware Activation Notices (FAN):</b> If this bit is set to '1', then the controller supports the Firmware Activation Notices event. If this bit is cleared to '0', then the controller does not support the Firmware Activation Notices event.				
				8 <b>Attached Namespace Attribute Notices (NSAN):</b> If this bit is set to '1', then the controller supports the Attached Namespace Attribute Notices event and the associated Changed Attached Namespace List log page. If this bit is cleared to '0', then the controller does not support the Attached Namespace Attribute Notices event nor the associated Changed Attached Namespace List log page.				
7:0 Reserved								
99:96	M	M	O <sup>4</sup>	<b>Controller Attributes (CTRATT):</b> This field indicates attributes of the controller. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>31:20</td> <td>Reserved</td> </tr> </tbody> </table>	Bits	Description	31:20	Reserved
Bits	Description							
31:20	Reserved							

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description						
				<p>19 <b>Flexible Data Placement Support (FDPS):</b> If this bit is set to '1', then the controller supports the Flexible Data Placement capability (refer to section 8.1.10). If this bit is cleared to '0', then the controller does not support the Flexible Data Placement capability.</p> <p>If Fixed Capacity Management is supported (i.e., the FCM bit is set to '1'), then this bit shall be cleared to '0'.</p>						
				<p>18 <b>Reservations and Host Identifier Interaction (RHII):</b> This bit indicates the reservations and Host Identifier interaction support for the controller.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Not Reported</td> </tr> <tr> <td>1b</td> <td>The Host Identifier is required to be set to a non-zero value for a host to use reservations (refer to section 8.1.22).</td> </tr> </tbody> </table>	Value	Definition	0b	Not Reported	1b	The Host Identifier is required to be set to a non-zero value for a host to use reservations (refer to section 8.1.22).
Value	Definition									
0b	Not Reported									
1b	The Host Identifier is required to be set to a non-zero value for a host to use reservations (refer to section 8.1.22).									
				<p>17 <b>HMB Restrict Non-Operational Power State Access (HMBR):</b> If this bit is set to '1', then the controller supports restricting HMB access in non-operational power states as defined in section 5.1.25.2.4. If this bit is cleared to '0', then the controller does not support restricting HMB access in non-operational power states as defined by section 5.1.25.2.4.</p>						
				<p>16 <b>MDTS and Size Limits Exclude Metadata (MEM):</b> If this bit is set to '1', then:</p> <ul style="list-style-type: none"> <li>The controller reported MDTS values do not include interleaved metadata.</li> <li>The controller reported VSL, WZSL, and WUSL values in the I/O Command Set specific Identify Controller data structure (refer to the NVM Command Set Specification) do not include interleaved metadata.</li> </ul> <p>If this bit is cleared to '0', then:</p> <ul style="list-style-type: none"> <li>The controller reported MDTS values include interleaved metadata.</li> <li>The controller reported VSL, WZSL, and WUSL values in the I/O Command Set specific Identify Controller data structure include interleaved metadata.</li> </ul>						
				<p>15 <b>Extended LBA Formats Supported (ELBAS):</b> If this bit is set to '1', then the controller supports the I/O command set specific extended protection information formats (refer to the Protection Information Formats section of the applicable I/O command set specification).</p> <p>If this bit is cleared to '0', then the controller does not support the I/O command set specific extended protection information formats (refer to the Protection Information Formats section of the NVM Command Set Specification).</p> <p>Refer to the LBA Format Extension Enable (LBAFEE) field in the Host Behavior Support feature (refer to section 5.1.25.1.14) for details for host software to enable the controller to operate on namespaces using the protection information formats.</p> <p>NOTE: This bit field applies to all I/O Command Sets. The original name has been retained for historical continuity.</p>						
				<p>14 <b>Delete NVM Set (DNVMS):</b> If this bit is set to '1', then the controller supports the Delete NVM Set operation (refer to section 8.1.4.3). If this bit is cleared to '0', then the controller does not support the Delete NVM Set operation.</p>						



**Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent**

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description
				<p>13 <b>Delete Endurance Group (DEG):</b> If this bit is set to '1', then the controller supports the Delete Endurance Group operation (refer to section 8.1.4.3). If this bit is cleared to '0', then the controller does not support the Delete Endurance Group operation.</p>
				<p>12 <b>Variable Capacity Management (VCM):</b> If this bit is set to '1', then the controller supports Variable Capacity Management (refer to section 8.1.4.3). If this bit is cleared to '0', then the controller does not support Variable Capacity Management.</p>
				<p>11 <b>Fixed Capacity Management (FCM):</b> If this bit is set to '1', then the controller supports Fixed Capacity Management (refer to section 8.1.4.2). If this bit is cleared to '0', then the controller does not support Fixed Capacity Management.</p> <p>If the Flexible Data Placement capability is supported (i.e., the FDPS bit is set to '1'), then this bit shall be cleared to '0'.</p>
				<p>10 <b>Multi-Domain Subsystem (MDS):</b> If this bit is set to '1', then the NVM subsystem supports the multiple domains (refer to section 3.2.5). If this bit is cleared to '0', then the NVM subsystem does not support the reporting of multiple domains and the NVM subsystem consists of a single domain.</p>
				<p>9 <b>UUID List (ULIST):</b> If this bit is set to '1', then the controller supports reporting of a UUID List (refer to Figure 320). If this bit is cleared to '0', then the controller does not support reporting of a UUID List (refer to section 8.1.28).</p>
				<p>8 <b>SQ Associations (SQA):</b> If this bit is set to '1', then the controller supports SQ Associations (refer to section 8.1.25). If this bit is cleared to '0', then the controller does not support SQ Associations.</p>
				<p>7 <b>Namespace Granularity (NG):</b> If this bit is set to '1', then the controller supports reporting of Namespace Granularity (refer to section 5.1.13.2.13). If this bit is cleared to '0', then the controller does not support reporting of Namespace Granularity. If the Namespace Management capability (refer to section 8.1.15) is not supported, then this bit shall be cleared to '0'.</p>
				<p>6 <b>Traffic Based Keep Alive Support (TBKAS):</b> If this bit is set to '1', then the controller uses Traffic Based Keep Alive (refer to section 3.9.4.1). If this bit is cleared to '0', then the controller does not use Traffic Based Keep Alive.</p>
				<p>5 <b>Predictable Latency Mode (PLM):</b> If this bit is set to '1', then the controller supports Predictable Latency Mode (refer to section 8.1.18). If this bit is cleared to '0', then the controller does not support Predictable Latency Mode.</p>
				<p>4 <b>Endurance Groups (EGS):</b> If this bit is set to '1', then the controller supports Endurance Groups (refer to section 3.2.3). If this bit is cleared to '0', then the controller does not support Endurance Groups.</p>
				<p>3 <b>Read Recovery Levels (RRLVLS):</b> If this bit is set to '1', then the controller supports Read Recovery Levels (refer to section 8.1.20). If this bit is cleared to '0', then the controller does not support Read Recovery Levels.</p>
				<p>2 <b>NVM Sets (NSETS):</b> If this bit is set to '1', then the controller supports NVM Sets (refer to section 3.2.2). If this bit is cleared to '0', then the controller does not support NVM Sets.</p>

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description																																		
				<table border="1"> <tr> <td>1</td> <td><b>Non-Operational Power State Permissive Mode (NOPSPM):</b> If this bit is set to '1', then the controller supports host control of whether the controller may temporarily exceed the power of a non-operational power state for the purpose of executing controller-initiated background operations in a non-operational power state (i.e., Non-Operational Power State Permissive Mode supported). If this bit is cleared to '0', then the controller does not support host control of whether the controller may exceed the power of a non-operational state for the purpose of executing controller-initiated background operations in a non-operational state (i.e., Non-Operational Power State Permissive Mode not supported). Refer to section 5.1.25.1.10.</td> </tr> <tr> <td>0</td> <td><b>Host Identifier Support (HIDS):</b> If this bit is set to '1', then the controller supports a 128-bit Host Identifier. If this bit is cleared to '0', then the controller does not support a 128-bit Host Identifier.</td> </tr> </table>	1	<b>Non-Operational Power State Permissive Mode (NOPSPM):</b> If this bit is set to '1', then the controller supports host control of whether the controller may temporarily exceed the power of a non-operational power state for the purpose of executing controller-initiated background operations in a non-operational power state (i.e., Non-Operational Power State Permissive Mode supported). If this bit is cleared to '0', then the controller does not support host control of whether the controller may exceed the power of a non-operational state for the purpose of executing controller-initiated background operations in a non-operational state (i.e., Non-Operational Power State Permissive Mode not supported). Refer to section 5.1.25.1.10.	0	<b>Host Identifier Support (HIDS):</b> If this bit is set to '1', then the controller supports a 128-bit Host Identifier. If this bit is cleared to '0', then the controller does not support a 128-bit Host Identifier.																														
1	<b>Non-Operational Power State Permissive Mode (NOPSPM):</b> If this bit is set to '1', then the controller supports host control of whether the controller may temporarily exceed the power of a non-operational power state for the purpose of executing controller-initiated background operations in a non-operational power state (i.e., Non-Operational Power State Permissive Mode supported). If this bit is cleared to '0', then the controller does not support host control of whether the controller may exceed the power of a non-operational state for the purpose of executing controller-initiated background operations in a non-operational state (i.e., Non-Operational Power State Permissive Mode not supported). Refer to section 5.1.25.1.10.																																					
0	<b>Host Identifier Support (HIDS):</b> If this bit is set to '1', then the controller supports a 128-bit Host Identifier. If this bit is cleared to '0', then the controller does not support a 128-bit Host Identifier.																																					
101:100	O	O	R	<p><b>Read Recovery Levels Supported (RRLS):</b> If Read Recovery Levels (RRL) are supported, then this field shall be supported. If a bit is set to '1', then the corresponding Read Recovery Level is supported. If a bit is cleared to '0', then the corresponding Read Recovery Level is not supported.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15</td> <td>Read Recovery Level 15 (RRL15): Fast Fail<sup>1</sup></td> </tr> <tr> <td>14</td> <td>Read Recovery Level 14 (RRL14)</td> </tr> <tr> <td>13</td> <td>Read Recovery Level 13 (RRL13)</td> </tr> <tr> <td>12</td> <td>Read Recovery Level 12 (RRL12)</td> </tr> <tr> <td>11</td> <td>Read Recovery Level 11 (RRL11)</td> </tr> <tr> <td>10</td> <td>Read Recovery Level 10 (RRL10)</td> </tr> <tr> <td>9</td> <td>Read Recovery Level 9 (RRL9)</td> </tr> <tr> <td>8</td> <td>Read Recovery Level 8 (RRL8)</td> </tr> <tr> <td>7</td> <td>Read Recovery Level 7 (RRL7)</td> </tr> <tr> <td>6</td> <td>Read Recovery Level 6 (RRL6)</td> </tr> <tr> <td>5</td> <td>Read Recovery Level 5 (RRL5)</td> </tr> <tr> <td>4</td> <td>Read Recovery Level 4 (RRL4): Default<sup>1</sup></td> </tr> <tr> <td>3</td> <td>Read Recovery Level 3 (RRL3)</td> </tr> <tr> <td>2</td> <td>Read Recovery Level 2 (RRL2)</td> </tr> <tr> <td>1</td> <td>Read Recovery Level 1 (RRL1)</td> </tr> <tr> <td>0</td> <td>Read Recovery Level 0 (RRL0)</td> </tr> </tbody> </table> <p>Notes:                      1. If Read Recovery Levels are supported, then this bit shall be set to '1'.</p>	Bits	Description	15	Read Recovery Level 15 (RRL15): Fast Fail <sup>1</sup>	14	Read Recovery Level 14 (RRL14)	13	Read Recovery Level 13 (RRL13)	12	Read Recovery Level 12 (RRL12)	11	Read Recovery Level 11 (RRL11)	10	Read Recovery Level 10 (RRL10)	9	Read Recovery Level 9 (RRL9)	8	Read Recovery Level 8 (RRL8)	7	Read Recovery Level 7 (RRL7)	6	Read Recovery Level 6 (RRL6)	5	Read Recovery Level 5 (RRL5)	4	Read Recovery Level 4 (RRL4): Default <sup>1</sup>	3	Read Recovery Level 3 (RRL3)	2	Read Recovery Level 2 (RRL2)	1	Read Recovery Level 1 (RRL1)	0	Read Recovery Level 0 (RRL0)
Bits	Description																																					
15	Read Recovery Level 15 (RRL15): Fast Fail <sup>1</sup>																																					
14	Read Recovery Level 14 (RRL14)																																					
13	Read Recovery Level 13 (RRL13)																																					
12	Read Recovery Level 12 (RRL12)																																					
11	Read Recovery Level 11 (RRL11)																																					
10	Read Recovery Level 10 (RRL10)																																					
9	Read Recovery Level 9 (RRL9)																																					
8	Read Recovery Level 8 (RRL8)																																					
7	Read Recovery Level 7 (RRL7)																																					
6	Read Recovery Level 6 (RRL6)																																					
5	Read Recovery Level 5 (RRL5)																																					
4	Read Recovery Level 4 (RRL4): Default <sup>1</sup>																																					
3	Read Recovery Level 3 (RRL3)																																					
2	Read Recovery Level 2 (RRL2)																																					
1	Read Recovery Level 1 (RRL1)																																					
0	Read Recovery Level 0 (RRL0)																																					

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description																								
102	M	R	R	<p><b>Boot Partition Capabilities (BPCAP):</b> This field indicates the Boot Partition capabilities supported by the controller.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>07:03</td> <td>Reserved</td> </tr> <tr> <td>02</td> <td> <p><b>Set Features Boot Partition Write Protection Support (SFBPWPS):</b> This bit indicates if Set Features Boot Partition Write Protection is supported by the controller. If supported, this capability allows a host to configure Boot Partition write protection states via the Boot Partition Write Protection Config feature in the Set Features command. Refer to section 8.1.3.3.1.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Set Features Boot Partition Write Protection is not supported by this controller.</td> </tr> <tr> <td>1b</td> <td>Set Features Boot Partition Write Protection is supported by this controller.</td> </tr> </tbody> </table> </td> </tr> <tr> <td>01:00</td> <td> <p><b>RPMB Boot Partition Write Protection Support (RPMBPWPS):</b> This field indicates if RPMB Boot Partition Write Protection is supported by the controller. If supported, this capability allows a host to configure Boot Partition write protection states via RPMB. Refer to section 8.1.3.3.2.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Support for RPMB Boot Partition Write Protection is not specified. Only controllers compliant with NVM Express Base Specification, Revision 2.0 and earlier are allowed to report this value. Refer to section 8.1.3.3 for more details on when a controller may return this value.</td> </tr> <tr> <td>01b</td> <td>RPMB Boot Partition Write Protection is not supported by this controller.</td> </tr> <tr> <td>10b</td> <td>RPMB Boot Partition Write Protection is supported by this controller.</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table> </td> </tr> </tbody> </table>	Bits	Description	07:03	Reserved	02	<p><b>Set Features Boot Partition Write Protection Support (SFBPWPS):</b> This bit indicates if Set Features Boot Partition Write Protection is supported by the controller. If supported, this capability allows a host to configure Boot Partition write protection states via the Boot Partition Write Protection Config feature in the Set Features command. Refer to section 8.1.3.3.1.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Set Features Boot Partition Write Protection is not supported by this controller.</td> </tr> <tr> <td>1b</td> <td>Set Features Boot Partition Write Protection is supported by this controller.</td> </tr> </tbody> </table>	Value	Definition	0b	Set Features Boot Partition Write Protection is not supported by this controller.	1b	Set Features Boot Partition Write Protection is supported by this controller.	01:00	<p><b>RPMB Boot Partition Write Protection Support (RPMBPWPS):</b> This field indicates if RPMB Boot Partition Write Protection is supported by the controller. If supported, this capability allows a host to configure Boot Partition write protection states via RPMB. Refer to section 8.1.3.3.2.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Support for RPMB Boot Partition Write Protection is not specified. Only controllers compliant with NVM Express Base Specification, Revision 2.0 and earlier are allowed to report this value. Refer to section 8.1.3.3 for more details on when a controller may return this value.</td> </tr> <tr> <td>01b</td> <td>RPMB Boot Partition Write Protection is not supported by this controller.</td> </tr> <tr> <td>10b</td> <td>RPMB Boot Partition Write Protection is supported by this controller.</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	Support for RPMB Boot Partition Write Protection is not specified. Only controllers compliant with NVM Express Base Specification, Revision 2.0 and earlier are allowed to report this value. Refer to section 8.1.3.3 for more details on when a controller may return this value.	01b	RPMB Boot Partition Write Protection is not supported by this controller.	10b	RPMB Boot Partition Write Protection is supported by this controller.	11b	Reserved
				Bits	Description																							
				07:03	Reserved																							
				02	<p><b>Set Features Boot Partition Write Protection Support (SFBPWPS):</b> This bit indicates if Set Features Boot Partition Write Protection is supported by the controller. If supported, this capability allows a host to configure Boot Partition write protection states via the Boot Partition Write Protection Config feature in the Set Features command. Refer to section 8.1.3.3.1.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>Set Features Boot Partition Write Protection is not supported by this controller.</td> </tr> <tr> <td>1b</td> <td>Set Features Boot Partition Write Protection is supported by this controller.</td> </tr> </tbody> </table>	Value	Definition	0b	Set Features Boot Partition Write Protection is not supported by this controller.	1b	Set Features Boot Partition Write Protection is supported by this controller.																	
Value	Definition																											
0b	Set Features Boot Partition Write Protection is not supported by this controller.																											
1b	Set Features Boot Partition Write Protection is supported by this controller.																											
01:00	<p><b>RPMB Boot Partition Write Protection Support (RPMBPWPS):</b> This field indicates if RPMB Boot Partition Write Protection is supported by the controller. If supported, this capability allows a host to configure Boot Partition write protection states via RPMB. Refer to section 8.1.3.3.2.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Support for RPMB Boot Partition Write Protection is not specified. Only controllers compliant with NVM Express Base Specification, Revision 2.0 and earlier are allowed to report this value. Refer to section 8.1.3.3 for more details on when a controller may return this value.</td> </tr> <tr> <td>01b</td> <td>RPMB Boot Partition Write Protection is not supported by this controller.</td> </tr> <tr> <td>10b</td> <td>RPMB Boot Partition Write Protection is supported by this controller.</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	Support for RPMB Boot Partition Write Protection is not specified. Only controllers compliant with NVM Express Base Specification, Revision 2.0 and earlier are allowed to report this value. Refer to section 8.1.3.3 for more details on when a controller may return this value.	01b	RPMB Boot Partition Write Protection is not supported by this controller.	10b	RPMB Boot Partition Write Protection is supported by this controller.	11b	Reserved																	
Value	Definition																											
00b	Support for RPMB Boot Partition Write Protection is not specified. Only controllers compliant with NVM Express Base Specification, Revision 2.0 and earlier are allowed to report this value. Refer to section 8.1.3.3 for more details on when a controller may return this value.																											
01b	RPMB Boot Partition Write Protection is not supported by this controller.																											
10b	RPMB Boot Partition Write Protection is supported by this controller.																											
11b	Reserved																											
103				Reserved																								
107:104	O	O	R	<p><b>NVM Subsystem Shutdown Latency (NSSL):</b> This field indicates the typical latency in microseconds for an NVM Subsystem Shutdown to complete. Refer to section 3.6.3. A value of 0h indicates that NVM Subsystem Shutdown Latency is not reported.</p>																								
109:108				Reserved																								
110	O	O	R	<p><b>Power Loss Signaling Information (PLSI):</b> This field indicates information about Power Loss Signaling processing capabilities.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:2</td> <td>Reserved</td> </tr> <tr> <td>1</td> <td> <p><b>PLS Forced Quiescence (PLSFQ):</b> If this bit is set to '1', then the controller supports Power Loss Signaling with Forced Quiescence (refer to section 8.2.5.2). If this bit is cleared to '0', then the controller does not support Power Loss Signaling with Forced Quiescence.</p> </td> </tr> <tr> <td>0</td> <td> <p><b>PLS Emergency Power Fail (PLSEPF):</b> If this bit is set to '1', then the controller supports Power Loss Signaling with Emergency Power Fail (refer to section 8.2.5.3). If this bit is cleared to '0', then the controller does not support Power Loss Signaling with Emergency Power Fail.</p> </td> </tr> </tbody> </table>	Bits	Description	7:2	Reserved	1	<p><b>PLS Forced Quiescence (PLSFQ):</b> If this bit is set to '1', then the controller supports Power Loss Signaling with Forced Quiescence (refer to section 8.2.5.2). If this bit is cleared to '0', then the controller does not support Power Loss Signaling with Forced Quiescence.</p>	0	<p><b>PLS Emergency Power Fail (PLSEPF):</b> If this bit is set to '1', then the controller supports Power Loss Signaling with Emergency Power Fail (refer to section 8.2.5.3). If this bit is cleared to '0', then the controller does not support Power Loss Signaling with Emergency Power Fail.</p>																
				Bits	Description																							
				7:2	Reserved																							
1	<p><b>PLS Forced Quiescence (PLSFQ):</b> If this bit is set to '1', then the controller supports Power Loss Signaling with Forced Quiescence (refer to section 8.2.5.2). If this bit is cleared to '0', then the controller does not support Power Loss Signaling with Forced Quiescence.</p>																											
0	<p><b>PLS Emergency Power Fail (PLSEPF):</b> If this bit is set to '1', then the controller supports Power Loss Signaling with Emergency Power Fail (refer to section 8.2.5.3). If this bit is cleared to '0', then the controller does not support Power Loss Signaling with Emergency Power Fail.</p>																											

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description												
111	M	M	M	<p><b>Controller Type (CNTRLTYPE):</b> This field specifies the controller type. A value of 0h indicates that the controller type is not reported.</p> <p>Implementations compliant with NVM Express Base Specification, Revision 1.4 or later shall report a controller type (i.e., the value 0h is reserved and shall not be used). Implementations compliant with an earlier specification version may report a value of 0h to indicate that a controller type is not reported.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Controller Type</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Reserved (controller type not reported)</td> </tr> <tr> <td>1h</td> <td>I/O controller</td> </tr> <tr> <td>2h</td> <td>Discovery controller</td> </tr> <tr> <td>3h</td> <td>Administrative controller</td> </tr> <tr> <td>4h to FFh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Controller Type	0h	Reserved (controller type not reported)	1h	I/O controller	2h	Discovery controller	3h	Administrative controller	4h to FFh	Reserved
Value	Controller Type															
0h	Reserved (controller type not reported)															
1h	I/O controller															
2h	Discovery controller															
3h	Administrative controller															
4h to FFh	Reserved															
127:112	O	O	R	<p><b>FRU Globally Unique Identifier (FGUID):</b> This field contains a 128-bit value that is globally unique for a given Field Replaceable Unit (FRU). Refer to the NVM Express® Management Interface Specification for the definition of a FRU. This field remains fixed throughout the life of the FRU. This field shall contain the same value for each controller associated with a given FRU.</p> <p>This field uses the EUI-64 based 16-byte designator format. Bytes 122:120 contain the 24-bit Organizationally Unique Identifier (OUI) value assigned by the IEEE Registration Authority. Bytes 127:123 contain an extension identifier assigned by the corresponding organization. Bytes 119:112 contain the vendor specific extension identifier assigned by the corresponding organization. Refer to the IEEE EUI-64 guidelines for more information. This field is big endian (refer to section 4.5.4).</p> <p>When not implemented, this field contains a value of 0h.</p>												
129:128	O	O	R	<p><b>Command Retry Delay Time 1 (CRDT1):</b> If the Do Not Retry (DNR) bit is cleared to '0' in the CQE and the Command Retry Delay (CRD) field is set to 01b in the CQE, then this value indicates the command retry delay time in units of 100 milliseconds.</p>												
131:130	O	O	R	<p><b>Command Retry Delay Time 2 (CRDT2):</b> If the DNR bit is cleared to '0' in the CQE and the CRD field is set to 10b in the CQE, then this value indicates the command retry delay time in units of 100 milliseconds.</p>												
133:132	O	O	R	<p><b>Command Retry Delay Time 3 (CRDT3):</b> If the DNR bit is cleared to '0' in the CQE and CRD field is set to 11b in the CQE, then this value indicates the command retry delay time in units of 100 milliseconds.</p>												
134	O	R	R	<p><b>Controller Reachability Capabilities (CRCAP):</b> This field specifies reachability capabilities of the controller and NVM subsystem.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:2</td> <td>Reserved</td> </tr> <tr> <td>1</td> <td><b>Reachability Group ID Changeable (RGIDC):</b> If this bit is set to '1', then the RGRPID field in the I/O Command Set Independent Identify Namespace data structure (refer to Figure 319) does not change while the namespace is attached to any controller. If this bit is cleared to '0', then the RGRPID field may change while the namespace is attached to any controller. Refer to section 8.1.19.</td> </tr> <tr> <td>0</td> <td><b>Reachability Reporting Supported (RRSUP):</b> If this bit is set to '1', then the NVM subsystem supports Reachability Reporting (refer to section 8.1.19). If this bit is cleared to '0', then the NVM subsystem does not support Reachability Reporting.</td> </tr> </tbody> </table>	Bits	Description	7:2	Reserved	1	<b>Reachability Group ID Changeable (RGIDC):</b> If this bit is set to '1', then the RGRPID field in the I/O Command Set Independent Identify Namespace data structure (refer to Figure 319) does not change while the namespace is attached to any controller. If this bit is cleared to '0', then the RGRPID field may change while the namespace is attached to any controller. Refer to section 8.1.19.	0	<b>Reachability Reporting Supported (RRSUP):</b> If this bit is set to '1', then the NVM subsystem supports Reachability Reporting (refer to section 8.1.19). If this bit is cleared to '0', then the NVM subsystem does not support Reachability Reporting.				
Bits	Description															
7:2	Reserved															
1	<b>Reachability Group ID Changeable (RGIDC):</b> If this bit is set to '1', then the RGRPID field in the I/O Command Set Independent Identify Namespace data structure (refer to Figure 319) does not change while the namespace is attached to any controller. If this bit is cleared to '0', then the RGRPID field may change while the namespace is attached to any controller. Refer to section 8.1.19.															
0	<b>Reachability Reporting Supported (RRSUP):</b> If this bit is set to '1', then the NVM subsystem supports Reachability Reporting (refer to section 8.1.19). If this bit is cleared to '0', then the NVM subsystem does not support Reachability Reporting.															
239:135				Reserved												
252:240				Reserved for the NVMe Management Interface.												

**Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent**

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description								
253	M	M	M	<p><b>NVM Subsystem Report (NVMSR):</b> This field reports information associated with the NVM subsystem. If the controller is compliant with the NVM Express Management Interface Specification, then at least one bit in this field is set to '1'. If the NVM subsystem does not support the NVM Express Management Interface Specification, then this field shall be cleared to 0h. Refer to the NVM Express Management Interface Specification.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:2</td> <td>Reserved</td> </tr> <tr> <td>1</td> <td><b>NVMe Enclosure (NVMEE):</b> If this bit is set to '1', then the NVM subsystem is part of an NVMe Enclosure. If this bit is cleared to '0', then the NVM subsystem is not part of an NVMe Enclosure.</td> </tr> <tr> <td>0</td> <td><b>NVMe Storage Device (NVMESD):</b> If this bit is set to '1', then the NVM subsystem is part of an NVMe Storage Device. If this bit is cleared to '0', then the NVM subsystem is not part of an NVMe Storage Device.</td> </tr> </tbody> </table>	Bits	Description	7:2	Reserved	1	<b>NVMe Enclosure (NVMEE):</b> If this bit is set to '1', then the NVM subsystem is part of an NVMe Enclosure. If this bit is cleared to '0', then the NVM subsystem is not part of an NVMe Enclosure.	0	<b>NVMe Storage Device (NVMESD):</b> If this bit is set to '1', then the NVM subsystem is part of an NVMe Storage Device. If this bit is cleared to '0', then the NVM subsystem is not part of an NVMe Storage Device.
				Bits	Description							
				7:2	Reserved							
				1	<b>NVMe Enclosure (NVMEE):</b> If this bit is set to '1', then the NVM subsystem is part of an NVMe Enclosure. If this bit is cleared to '0', then the NVM subsystem is not part of an NVMe Enclosure.							
0	<b>NVMe Storage Device (NVMESD):</b> If this bit is set to '1', then the NVM subsystem is part of an NVMe Storage Device. If this bit is cleared to '0', then the NVM subsystem is not part of an NVMe Storage Device.											
254	M	M	M	<p><b>VPD Write Cycle Information (VWCI):</b> This field indicates information about the remaining number of times that VPD contents are able to be updated using the VPD Write command. Refer to the NVM Express Management Interface Specification for details on VPD contents and the VPD Write command.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7</td> <td><b>VPD Write Cycles Remaining Valid (VWCRV):</b> If this bit is set to '1', then the VPD Write Cycles Remaining field is valid. If this bit is cleared to '0', then the VPD Write Cycles Remaining field is invalid and cleared to '0'.</td> </tr> <tr> <td>6:0</td> <td><b>VPD Write Cycles Remaining (VWCR):</b> If the VPD Write Cycle Remaining Valid bit is set to '1', then this field contains a value indicating the remaining number of times that VPD contents are able to be updated in units of 256 bytes using the VPD Write command. For example, a 1 KiB FRU Information Device that can be updated 8 times would indicate a value of 32 in this field. If this field is set to 7Fh, then the remaining number of times that VPD contents are able to be updated using the VPD Write command is greater than or equal to 7Fh.  If the VPD Write Cycle Remaining Valid bit is cleared to '0', then this field is not valid and shall be cleared to a value of 0h.</td> </tr> </tbody> </table>	Bits	Description	7	<b>VPD Write Cycles Remaining Valid (VWCRV):</b> If this bit is set to '1', then the VPD Write Cycles Remaining field is valid. If this bit is cleared to '0', then the VPD Write Cycles Remaining field is invalid and cleared to '0'.	6:0	<b>VPD Write Cycles Remaining (VWCR):</b> If the VPD Write Cycle Remaining Valid bit is set to '1', then this field contains a value indicating the remaining number of times that VPD contents are able to be updated in units of 256 bytes using the VPD Write command. For example, a 1 KiB FRU Information Device that can be updated 8 times would indicate a value of 32 in this field. If this field is set to 7Fh, then the remaining number of times that VPD contents are able to be updated using the VPD Write command is greater than or equal to 7Fh.  If the VPD Write Cycle Remaining Valid bit is cleared to '0', then this field is not valid and shall be cleared to a value of 0h.		
				Bits	Description							
7	<b>VPD Write Cycles Remaining Valid (VWCRV):</b> If this bit is set to '1', then the VPD Write Cycles Remaining field is valid. If this bit is cleared to '0', then the VPD Write Cycles Remaining field is invalid and cleared to '0'.											
6:0	<b>VPD Write Cycles Remaining (VWCR):</b> If the VPD Write Cycle Remaining Valid bit is set to '1', then this field contains a value indicating the remaining number of times that VPD contents are able to be updated in units of 256 bytes using the VPD Write command. For example, a 1 KiB FRU Information Device that can be updated 8 times would indicate a value of 32 in this field. If this field is set to 7Fh, then the remaining number of times that VPD contents are able to be updated using the VPD Write command is greater than or equal to 7Fh.  If the VPD Write Cycle Remaining Valid bit is cleared to '0', then this field is not valid and shall be cleared to a value of 0h.											
255	M	M	M	<p><b>Management Endpoint Capabilities (MEC):</b> This field indicates the support for Management Endpoints in the NVM subsystem. Refer to the NVM Express Management Interface Specification for details.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:2</td> <td>Reserved</td> </tr> <tr> <td>1</td> <td><b>PCIe Port Management Endpoint (PCIEME):</b> If the NVM subsystem contains one or more Management Endpoints on one or more PCIe ports, then this bit shall be set to '1'; otherwise, this bit shall be cleared to '0'.</td> </tr> <tr> <td>0</td> <td><b>2-Wire Port Management Endpoint (TWPME):</b> If the NVM subsystem contains one or more Management Endpoints on the 2-Wire port, then this bit shall be set to '1'; otherwise, this bit shall be cleared to '0'.  This bit was formerly named SMBus/I2C Port Management Endpoint (SMBUSME).</td> </tr> </tbody> </table>	Bits	Description	7:2	Reserved	1	<b>PCIe Port Management Endpoint (PCIEME):</b> If the NVM subsystem contains one or more Management Endpoints on one or more PCIe ports, then this bit shall be set to '1'; otherwise, this bit shall be cleared to '0'.	0	<b>2-Wire Port Management Endpoint (TWPME):</b> If the NVM subsystem contains one or more Management Endpoints on the 2-Wire port, then this bit shall be set to '1'; otherwise, this bit shall be cleared to '0'.  This bit was formerly named SMBus/I2C Port Management Endpoint (SMBUSME).
				Bits	Description							
				7:2	Reserved							
1	<b>PCIe Port Management Endpoint (PCIEME):</b> If the NVM subsystem contains one or more Management Endpoints on one or more PCIe ports, then this bit shall be set to '1'; otherwise, this bit shall be cleared to '0'.											
0	<b>2-Wire Port Management Endpoint (TWPME):</b> If the NVM subsystem contains one or more Management Endpoints on the 2-Wire port, then this bit shall be set to '1'; otherwise, this bit shall be cleared to '0'.  This bit was formerly named SMBus/I2C Port Management Endpoint (SMBUSME).											
<b>Admin Command Set Attributes &amp; Optional Controller Capabilities</b>												
257:256	M	M	R	<p><b>Optional Admin Command Support (OACS):</b> This field indicates the optional Admin commands and features supported by the controller. Refer to section 3.1.3.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:12</td> <td>Reserved</td> </tr> </tbody> </table>	Bits	Description	15:12	Reserved				
				Bits	Description							
15:12	Reserved											

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description
				<p><b>11</b> <b>Host Managed Live Migration Support (HMLMS):</b> If this bit is set to '1', then the controller supports the Host Managed Live Migration capability (refer to section 8.1.12).</p> <p>If this bit is cleared to '0', then the controller does not support the Host Managed Live Migration capability.</p> <p>Secondary controllers shall clear this bit to '0'.</p>
				<p><b>10</b> <b>Command and Feature Lockdown Supported (CFLS):</b> If this bit is set to '1', then the controller supports the Command and Feature Lockdown capability (refer to section 8.1.5). If this bit is cleared to '0', then the controller does not support the Command and Feature Lockdown capability. The value of this bit shall be the same for all controllers in the NVM subsystem.</p>
				<p><b>9</b> <b>Get LBA Status Supported (GLSS):</b> If this bit is set to '1', then the controller supports the Get LBA Status capability with the Action Type values of 10h and 11h (refer to the NVM Command Set Specification). If this bit is cleared to '0', then the controller does not support the Get LBA Status capability with the Action Type values of 10h and 11h.</p>
				<p><b>8</b> <b>Doorbell Buffer Config Supported (DBCS):</b> If this bit is set to '1', then the controller supports the Doorbell Buffer Config command. If this bit is cleared to '0', then the controller does not support the Doorbell Buffer Config command.</p>
				<p><b>7</b> <b>Virtualization Management Supported (VMS):</b> If this bit is set to '1', then the controller supports the Virtualization Management command. If this bit is cleared to '0', then the controller does not support the Virtualization Management command.</p>
				<p><b>6</b> <b>NVMe-MI Send Receive Supported (NSRS):</b> If this bit is set to '1', then the controller supports the NVMe-MI Send and NVMe-MI Receive commands. If this bit is cleared to '0', then the controller does not support the NVMe-MI Send and NVMe-MI Receive commands.</p>
				<p><b>5</b> <b>Directives Supported (DIRS):</b> If this bit is set to '1', then the controller supports Directives. If this bit is cleared to '0', then the controller does not support Directives. A controller that supports Directives shall support the Directive Send and Directive Receive commands. Refer to section 8.1.8.</p>
				<p><b>4</b> <b>Device Self-test Supported (DSTS):</b> If this bit is set to '1', then the controller supports the Device Self-test command. If this bit is cleared to '0', then the controller does not support the Device Self-test command.</p>
				<p><b>3</b> <b>Namespace Management Supported (NMS):</b> If this bit is set to '1', then the controller supports the Namespace Management capability (refer to section 8.1.15). If this bit is cleared to '0', then the controller does not support the Namespace Management capability.</p>
				<p><b>2</b> <b>Firmware Download Supported (FWDS):</b> If this bit is set to '1', then the controller supports the Firmware Commit and Firmware Image Download commands. If this bit is cleared to '0', then the controller does not support the Firmware Commit and Firmware Image Download commands.</p>
				<p><b>1</b> <b>Format NVM Supported (FNVMS):</b> If this bit is set to '1', then the controller supports the Format NVM command. If this bit is cleared to '0', then the controller does not support the Format NVM command.</p>
				<p><b>0</b> <b>Security Send Receive Supported (SSRS):</b> If this bit is set to '1', then the controller supports the Security Send and Security Receive commands. If this bit is cleared to '0', then the controller does not support the Security Send and Security Receive commands.</p>
258	M	M	R	<p><b>Abort Command Limit (ACL):</b> This field indicates the maximum number of concurrently executing Abort commands (refer to section 5.1) on the Admin Queue supported by the controller. This is a 0's based value. It is recommended that implementations support concurrent execution of a minimum of four Abort commands.</p>

**Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent**

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description												
259	M	M	M <sup>3</sup>	<b>Asynchronous Event Request Limit (AERL):</b> This field is used to convey the maximum number of concurrently outstanding Asynchronous Event Request commands supported by the controller (refer to section 5.1.2). This is a 0's based value. It is recommended that implementations support a minimum of four Asynchronous Event Request commands outstanding simultaneously.												
260	M	M	R	<b>Firmware Updates (FRMW):</b> This field indicates capabilities regarding firmware updates. Refer to section 3.11 for more information on the firmware update process.												
				<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:6</td> <td>Reserved</td> </tr> <tr> <td>5</td> <td><b>Support Multiple Update Detection (SMUD):</b> If this bit is set to '1', then the controller is able to detect overlapping firmware/boot partition image update command sequences (refer to section 3.11 and section 8.1.3.2). If this bit is cleared to '0', then the controller is not able to detect overlapping firmware/boot partition image update command sequences.</td> </tr> <tr> <td>4</td> <td><b>Firmware Activation Without Reset (FAWR):</b> If this bit is set to '1', then the controller supports firmware activation without a reset. If this bit is cleared to '0', then the controller requires a reset for firmware to be activated.</td> </tr> <tr> <td>3:1</td> <td><b>Number Of Firmware Slots (NOFS):</b> This field indicates the number of firmware slots supported by the domain that contains this controller. This field shall specify a value from one to seven, indicating that at least one firmware slot is supported and up to seven maximum. This corresponds to firmware slots 1 through 7</td> </tr> <tr> <td>0</td> <td><b>First Firmware Slot Read Only (FFSRO):</b> If this bit is set to '1', then the first firmware slot (i.e., slot 1) is read only. If this bit is cleared to '0', then the first firmware slot (i.e., slot 1) is read/write. Implementations may choose to have a baseline read only firmware image.</td> </tr> </tbody> </table>	Bits	Description	7:6	Reserved	5	<b>Support Multiple Update Detection (SMUD):</b> If this bit is set to '1', then the controller is able to detect overlapping firmware/boot partition image update command sequences (refer to section 3.11 and section 8.1.3.2). If this bit is cleared to '0', then the controller is not able to detect overlapping firmware/boot partition image update command sequences.	4	<b>Firmware Activation Without Reset (FAWR):</b> If this bit is set to '1', then the controller supports firmware activation without a reset. If this bit is cleared to '0', then the controller requires a reset for firmware to be activated.	3:1	<b>Number Of Firmware Slots (NOFS):</b> This field indicates the number of firmware slots supported by the domain that contains this controller. This field shall specify a value from one to seven, indicating that at least one firmware slot is supported and up to seven maximum. This corresponds to firmware slots 1 through 7	0	<b>First Firmware Slot Read Only (FFSRO):</b> If this bit is set to '1', then the first firmware slot (i.e., slot 1) is read only. If this bit is cleared to '0', then the first firmware slot (i.e., slot 1) is read/write. Implementations may choose to have a baseline read only firmware image.
				Bits	Description											
				7:6	Reserved											
				5	<b>Support Multiple Update Detection (SMUD):</b> If this bit is set to '1', then the controller is able to detect overlapping firmware/boot partition image update command sequences (refer to section 3.11 and section 8.1.3.2). If this bit is cleared to '0', then the controller is not able to detect overlapping firmware/boot partition image update command sequences.											
4	<b>Firmware Activation Without Reset (FAWR):</b> If this bit is set to '1', then the controller supports firmware activation without a reset. If this bit is cleared to '0', then the controller requires a reset for firmware to be activated.															
3:1	<b>Number Of Firmware Slots (NOFS):</b> This field indicates the number of firmware slots supported by the domain that contains this controller. This field shall specify a value from one to seven, indicating that at least one firmware slot is supported and up to seven maximum. This corresponds to firmware slots 1 through 7															
0	<b>First Firmware Slot Read Only (FFSRO):</b> If this bit is set to '1', then the first firmware slot (i.e., slot 1) is read only. If this bit is cleared to '0', then the first firmware slot (i.e., slot 1) is read/write. Implementations may choose to have a baseline read only firmware image.															

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description																		
261	M	M	M	<p><b>Log Page Attributes (LPA):</b> This field indicates optional attributes for log pages that are accessed via the Get Log Page command.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7</td> <td>Reserved</td> </tr> <tr> <td>6</td> <td> <p><b>Data Area 4 Support (DA4S):</b> If this bit is set to '1', then the controller supports Data Area 4 for the Telemetry Host-Initiated and Telemetry Controller-Initiated log. If this bit is cleared to '0', then the controller does not support Data Area 4 for the Telemetry Host-Initiated and Telemetry Controller-Initiated log pages.</p> </td> </tr> <tr> <td>5</td> <td> <p><b>Miscellaneous Log Page Support (MLPS):</b> If this bit is set to '1', then the controller supports:</p> <ul style="list-style-type: none"> <li>the Supported Log Pages log page (Log Page Identifier 0h);</li> <li>returning the scope of each command in the Commands Supported and Effects log page (Log Page Identifier 05h);</li> <li>the Feature Identifiers Supported and Effects log page (Log Page Identifier 12h); and</li> <li>the NVMe-MI Commands Supported and Effects log page (Log Page Identifier 13h).</li> </ul> <p>If this bit is cleared to '0', then the controller:</p> <ul style="list-style-type: none"> <li>does not support returning the scope of each command in the Commands Supported and Effects log page;</li> <li>may support the Supported Log Pages log page;</li> <li>may support the Feature Identifiers Supported and Effects log page; and</li> <li>may support the NVMe-MI Commands Supported and Effects log page.</li> </ul> </td> </tr> <tr> <td>4</td> <td> <p><b>Persistent Event Support (PES):</b> If this bit is set to '1', then the controller supports the Persistent Event log. If this bit is cleared to '0', then the controller does not support the Persistent Event log page.</p> </td> </tr> <tr> <td>3</td> <td> <p><b>Telemetry Support (TS):</b> If this bit is set to '1', then the controller supports the Telemetry Host-Initiated and Telemetry Controller-Initiated log pages and sending Telemetry Log Notices. If this bit is cleared to '0', then the controller does not support the Telemetry Host-Initiated and Telemetry Controller-Initiated log pages and Telemetry Log Notice events.</p> </td> </tr> <tr> <td>2</td> <td> <p><b>Log Page Extended Data Support (LPEDS):</b> If this bit is set to '1', then the controller supports extended data for the Get Log Page command (including extended Number of Dwords and Log Page Offset fields). If this bit is cleared to '0', then the controller does not support extended data for the Get Log Page command.</p> </td> </tr> <tr> <td>1</td> <td> <p><b>Commands Supported and Effects Support (CSES):</b> If this bit is set to '1', then the controller supports the Commands Supported and Effects log page. If this bit is cleared to '0', then the controller does not support the Commands Supported and Effects log page.</p> </td> </tr> <tr> <td>0</td> <td> <p><b>SMART Support (SMARTS):</b> If this bit is set to '1', then the controller supports the SMART / Health Information log page on a per namespace basis. If this bit is cleared to '0', then the controller does not support the SMART / Health Information log page on a per namespace basis.</p> </td> </tr> </tbody> </table>	Bits	Description	7	Reserved	6	<p><b>Data Area 4 Support (DA4S):</b> If this bit is set to '1', then the controller supports Data Area 4 for the Telemetry Host-Initiated and Telemetry Controller-Initiated log. If this bit is cleared to '0', then the controller does not support Data Area 4 for the Telemetry Host-Initiated and Telemetry Controller-Initiated log pages.</p>	5	<p><b>Miscellaneous Log Page Support (MLPS):</b> If this bit is set to '1', then the controller supports:</p> <ul style="list-style-type: none"> <li>the Supported Log Pages log page (Log Page Identifier 0h);</li> <li>returning the scope of each command in the Commands Supported and Effects log page (Log Page Identifier 05h);</li> <li>the Feature Identifiers Supported and Effects log page (Log Page Identifier 12h); and</li> <li>the NVMe-MI Commands Supported and Effects log page (Log Page Identifier 13h).</li> </ul> <p>If this bit is cleared to '0', then the controller:</p> <ul style="list-style-type: none"> <li>does not support returning the scope of each command in the Commands Supported and Effects log page;</li> <li>may support the Supported Log Pages log page;</li> <li>may support the Feature Identifiers Supported and Effects log page; and</li> <li>may support the NVMe-MI Commands Supported and Effects log page.</li> </ul>	4	<p><b>Persistent Event Support (PES):</b> If this bit is set to '1', then the controller supports the Persistent Event log. If this bit is cleared to '0', then the controller does not support the Persistent Event log page.</p>	3	<p><b>Telemetry Support (TS):</b> If this bit is set to '1', then the controller supports the Telemetry Host-Initiated and Telemetry Controller-Initiated log pages and sending Telemetry Log Notices. If this bit is cleared to '0', then the controller does not support the Telemetry Host-Initiated and Telemetry Controller-Initiated log pages and Telemetry Log Notice events.</p>	2	<p><b>Log Page Extended Data Support (LPEDS):</b> If this bit is set to '1', then the controller supports extended data for the Get Log Page command (including extended Number of Dwords and Log Page Offset fields). If this bit is cleared to '0', then the controller does not support extended data for the Get Log Page command.</p>	1	<p><b>Commands Supported and Effects Support (CSES):</b> If this bit is set to '1', then the controller supports the Commands Supported and Effects log page. If this bit is cleared to '0', then the controller does not support the Commands Supported and Effects log page.</p>	0	<p><b>SMART Support (SMARTS):</b> If this bit is set to '1', then the controller supports the SMART / Health Information log page on a per namespace basis. If this bit is cleared to '0', then the controller does not support the SMART / Health Information log page on a per namespace basis.</p>
				Bits	Description																	
				7	Reserved																	
				6	<p><b>Data Area 4 Support (DA4S):</b> If this bit is set to '1', then the controller supports Data Area 4 for the Telemetry Host-Initiated and Telemetry Controller-Initiated log. If this bit is cleared to '0', then the controller does not support Data Area 4 for the Telemetry Host-Initiated and Telemetry Controller-Initiated log pages.</p>																	
				5	<p><b>Miscellaneous Log Page Support (MLPS):</b> If this bit is set to '1', then the controller supports:</p> <ul style="list-style-type: none"> <li>the Supported Log Pages log page (Log Page Identifier 0h);</li> <li>returning the scope of each command in the Commands Supported and Effects log page (Log Page Identifier 05h);</li> <li>the Feature Identifiers Supported and Effects log page (Log Page Identifier 12h); and</li> <li>the NVMe-MI Commands Supported and Effects log page (Log Page Identifier 13h).</li> </ul> <p>If this bit is cleared to '0', then the controller:</p> <ul style="list-style-type: none"> <li>does not support returning the scope of each command in the Commands Supported and Effects log page;</li> <li>may support the Supported Log Pages log page;</li> <li>may support the Feature Identifiers Supported and Effects log page; and</li> <li>may support the NVMe-MI Commands Supported and Effects log page.</li> </ul>																	
				4	<p><b>Persistent Event Support (PES):</b> If this bit is set to '1', then the controller supports the Persistent Event log. If this bit is cleared to '0', then the controller does not support the Persistent Event log page.</p>																	
				3	<p><b>Telemetry Support (TS):</b> If this bit is set to '1', then the controller supports the Telemetry Host-Initiated and Telemetry Controller-Initiated log pages and sending Telemetry Log Notices. If this bit is cleared to '0', then the controller does not support the Telemetry Host-Initiated and Telemetry Controller-Initiated log pages and Telemetry Log Notice events.</p>																	
				2	<p><b>Log Page Extended Data Support (LPEDS):</b> If this bit is set to '1', then the controller supports extended data for the Get Log Page command (including extended Number of Dwords and Log Page Offset fields). If this bit is cleared to '0', then the controller does not support extended data for the Get Log Page command.</p>																	
1	<p><b>Commands Supported and Effects Support (CSES):</b> If this bit is set to '1', then the controller supports the Commands Supported and Effects log page. If this bit is cleared to '0', then the controller does not support the Commands Supported and Effects log page.</p>																					
0	<p><b>SMART Support (SMARTS):</b> If this bit is set to '1', then the controller supports the SMART / Health Information log page on a per namespace basis. If this bit is cleared to '0', then the controller does not support the SMART / Health Information log page on a per namespace basis.</p>																					
262	M	M	M	<p><b>Error Log Page Entries (ELPE):</b> This field indicates the maximum number of Error Information Log Entries that are stored by the controller. This field is a 0's based value.</p>																		



**Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent**

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description						
263	M	M	R	<p><b>Number of Power States Support (NPSS):</b> This field indicates the number of NVM Express power states supported by the controller. This is a 0's based value. Refer to section 8.1.17.</p> <p>Power states are numbered sequentially starting at power state 0. A controller shall support at least one power state (i.e., power state 0) and may support up to 31 additional power states (i.e., up to 32 total).</p>						
264	M	M	R	<p><b>Admin Vendor Specific Command Configuration (AVSCC):</b> This field indicates the configuration settings for vendor specific Admin command handling. Refer to section 8.1.26.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td><b>Vendor Specific Command Format (VSCF):</b> If this bit is set to '1', then all vendor specific Admin commands use the format defined in Figure 93. If this bit is cleared to '0', then the format of all vendor specific Admin commands is vendor specific.</td> </tr> </tbody> </table>	Bits	Description	7:1	Reserved	0	<b>Vendor Specific Command Format (VSCF):</b> If this bit is set to '1', then all vendor specific Admin commands use the format defined in Figure 93. If this bit is cleared to '0', then the format of all vendor specific Admin commands is vendor specific.
Bits	Description									
7:1	Reserved									
0	<b>Vendor Specific Command Format (VSCF):</b> If this bit is set to '1', then all vendor specific Admin commands use the format defined in Figure 93. If this bit is cleared to '0', then the format of all vendor specific Admin commands is vendor specific.									
265	O	O	R	<p><b>Autonomous Power State Transition Attributes (APSTA):</b> This field indicates the attributes of the autonomous power state transition feature. Refer to section 8.1.17.2.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td><b>Autonomous Power Transition Support (APTS):</b> If this bit is set to '1', then the controller supports autonomous power state transitions. If this bit is cleared to '0', then the controller does not support autonomous power state transitions.</td> </tr> </tbody> </table>	Bits	Description	7:1	Reserved	0	<b>Autonomous Power Transition Support (APTS):</b> If this bit is set to '1', then the controller supports autonomous power state transitions. If this bit is cleared to '0', then the controller does not support autonomous power state transitions.
Bits	Description									
7:1	Reserved									
0	<b>Autonomous Power Transition Support (APTS):</b> If this bit is set to '1', then the controller supports autonomous power state transitions. If this bit is cleared to '0', then the controller does not support autonomous power state transitions.									
267:266	M	M	R	<p><b>Warning Composite Temperature Threshold (WCTEMP):</b> This field indicates the minimum Composite Temperature field value (reported in the SMART / Health Information log page in Figure 206) that indicates an overheating condition during which controller operation continues. Immediate remediation is recommended (e.g., additional cooling or workload reduction). The platform should strive to maintain a composite temperature less than this value.</p> <p>A value of 0h in this field indicates that no warning temperature threshold value is reported by the controller. Implementations compliant with NVM Express Base Specification, Revision 1.2 or later shall report a non-zero value in this field.</p> <p>It is recommended that implementations report a value of 0157h in this field.</p>						
269:268	M	M	R	<p><b>Critical Composite Temperature Threshold (CCTEMP):</b> This field indicates the minimum Composite Temperature field value (reported in the SMART / Health Information log page in Figure 206) that indicates a critical overheating condition (e.g., may prevent continued normal operation, possibility of data loss, automatic device shutdown, extreme performance throttling, or permanent damage).</p> <p>A value of 0h in this field indicates that no critical temperature threshold value is reported by the controller. Implementations compliant with NVM Express Base Specification, Revision 1.2 or later shall report a non-zero value in this field.</p>						
271:270	O	O	R	<p><b>Maximum Time for Firmware Activation (MTFA):</b> This field indicates the maximum time the controller temporarily stops processing commands to activate the firmware image. This field shall be valid if the controller supports firmware activation without a reset. This field is specified in 100 millisecond units. A value of 0h indicates that the maximum time is undefined.</p> <p>For the amount of time to process a Firmware Commit command that specifies a value of 011b in the Commit Action field (i.e., firmware activation without reset), refer to the Maximum Processing Time for Firmware Activation Without Reset field.</p>						

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description																		
275:272	O	O	R	<p><b>Host Memory Buffer Preferred Size (HMPRE):</b> This field indicates the preferred size that the host is requested to allocate for the Host Memory Buffer feature in 4 KiB units. This value shall be greater than or equal to the Host Memory Buffer Minimum Size. If this field is non-zero, then the Host Memory Buffer feature is supported. If this field is cleared to 0h, then the Host Memory Buffer feature is not supported.</p> <p>If the Host Managed Live Migration Support (HMLMS) bit is set to '1' in one or more controllers in the NVM subsystem; then this field shall be cleared to 0h.</p>																		
279:276	O	O	R	<p><b>Host Memory Buffer Minimum Size (HMMIN):</b> This field indicates the minimum size that the host is requested to allocate for the Host Memory Buffer feature in 4 KiB units. If this field is cleared to 0h, then the host is requested to allocate any amount of host memory possible up to the HMPRE value.</p>																		
295:280	O	O	R	<p><b>Total NVM Capacity (TNVMCAP):</b> This field indicates the total NVM capacity that is accessible by the controller. The value is in bytes. This field shall be supported if the Namespace Management capability (refer to section 8.1.15) is supported or if the Capacity Management capability (refer to section 8.1.4) is supported.</p> <p>Refer to section 3.8.</p>																		
311:296	O	O	R	<p><b>Unallocated NVM Capacity (UNVMCAP):</b> This field indicates the unallocated NVM capacity that is accessible by the controller. The value is in bytes. This field shall be supported if the Namespace Management capability (refer to section 8.1.15) is supported or if the Capacity Management capability (refer to section 8.1.4) is supported.</p> <p>Refer to section 3.8.</p>																		
315:312	O	O	R	<p><b>Replay Protected Memory Block Support (RPMBS):</b> This field indicates if the controller supports one or more Replay Protected Memory Blocks (RPMBs) and the capabilities. Refer to section 8.1.21.</p>																		
				<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>31:24</td> <td> <p><b>Access Size (ASZE):</b> If the Number of RPMB Units field is non-zero, then this field indicates the maximum number of 512B units of data that may be read or written per RPMB access by Security Send or Security Receive commands for the controller. This is a 0's based value. A value of 0h indicates support for one unit of 512B of data.</p> <p>If the Number of RPMB Units field is 0h, then this field should be ignored by the host.</p> </td> </tr> <tr> <td>23:16</td> <td> <p><b>Total Size (TSZE):</b> If the Number of RPMB Units field is non-zero, then this field indicates the number of 128 KiB units of data in each RPMB supported in the controller. This is a 0's based value. A value of 0h indicates support for one unit of 128 KiB of data.</p> <p>If the Number of RPMB Units field is 0h, then this field should be ignored by the host.</p> </td> </tr> <tr> <td>15:06</td> <td>Reserved</td> </tr> <tr> <td>05:03</td> <td> <p><b>Authentication Method (AUTHM):</b> This field indicates the authentication method used to access all RPMBs in the controller. The values for this field are:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>HMAC SHA-256 (refer to RFC 6234)</td> </tr> <tr> <td>001b to 111b</td> <td>Reserved</td> </tr> </tbody> </table> </td> </tr> <tr> <td>02:00</td> <td> <p><b>Number of RPMB Units (NRPMBU):</b> This field indicates the number of RPMB targets the controller supports. All RPMB targets supported shall have the same capabilities as defined in the RPMBS field. A value of 0h indicates the controller does not support Replay Protected Memory Blocks. If this value is non-zero, then the controller shall support the Security Send and Security Receive commands.</p> </td> </tr> </tbody> </table>	Bits	Description	31:24	<p><b>Access Size (ASZE):</b> If the Number of RPMB Units field is non-zero, then this field indicates the maximum number of 512B units of data that may be read or written per RPMB access by Security Send or Security Receive commands for the controller. This is a 0's based value. A value of 0h indicates support for one unit of 512B of data.</p> <p>If the Number of RPMB Units field is 0h, then this field should be ignored by the host.</p>	23:16	<p><b>Total Size (TSZE):</b> If the Number of RPMB Units field is non-zero, then this field indicates the number of 128 KiB units of data in each RPMB supported in the controller. This is a 0's based value. A value of 0h indicates support for one unit of 128 KiB of data.</p> <p>If the Number of RPMB Units field is 0h, then this field should be ignored by the host.</p>	15:06	Reserved	05:03	<p><b>Authentication Method (AUTHM):</b> This field indicates the authentication method used to access all RPMBs in the controller. The values for this field are:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>HMAC SHA-256 (refer to RFC 6234)</td> </tr> <tr> <td>001b to 111b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	000b	HMAC SHA-256 (refer to RFC 6234)	001b to 111b	Reserved	02:00	<p><b>Number of RPMB Units (NRPMBU):</b> This field indicates the number of RPMB targets the controller supports. All RPMB targets supported shall have the same capabilities as defined in the RPMBS field. A value of 0h indicates the controller does not support Replay Protected Memory Blocks. If this value is non-zero, then the controller shall support the Security Send and Security Receive commands.</p>
				Bits	Description																	
				31:24	<p><b>Access Size (ASZE):</b> If the Number of RPMB Units field is non-zero, then this field indicates the maximum number of 512B units of data that may be read or written per RPMB access by Security Send or Security Receive commands for the controller. This is a 0's based value. A value of 0h indicates support for one unit of 512B of data.</p> <p>If the Number of RPMB Units field is 0h, then this field should be ignored by the host.</p>																	
				23:16	<p><b>Total Size (TSZE):</b> If the Number of RPMB Units field is non-zero, then this field indicates the number of 128 KiB units of data in each RPMB supported in the controller. This is a 0's based value. A value of 0h indicates support for one unit of 128 KiB of data.</p> <p>If the Number of RPMB Units field is 0h, then this field should be ignored by the host.</p>																	
15:06	Reserved																					
05:03	<p><b>Authentication Method (AUTHM):</b> This field indicates the authentication method used to access all RPMBs in the controller. The values for this field are:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>HMAC SHA-256 (refer to RFC 6234)</td> </tr> <tr> <td>001b to 111b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	000b	HMAC SHA-256 (refer to RFC 6234)	001b to 111b	Reserved															
Value	Definition																					
000b	HMAC SHA-256 (refer to RFC 6234)																					
001b to 111b	Reserved																					
02:00	<p><b>Number of RPMB Units (NRPMBU):</b> This field indicates the number of RPMB targets the controller supports. All RPMB targets supported shall have the same capabilities as defined in the RPMBS field. A value of 0h indicates the controller does not support Replay Protected Memory Blocks. If this value is non-zero, then the controller shall support the Security Send and Security Receive commands.</p>																					

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description				
317:316	O	O	R	<b>Extended Device Self-test Time (EDSTT):</b> If the Device Self-test command is supported, then this field indicates the nominal amount of time in one minute units that the controller takes to complete an extended device self-test operation when in power state 0. If the Device Self-test command is not supported, then this field is reserved.				
318	O	O	R	<b>Device Self-test Options (DSTO):</b> This field indicates the optional Device Self-test command or operation behaviors supported by the controller or NVM subsystem.				
				<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:2</td> <td>Reserved</td> </tr> </tbody> </table>	Bits	Description	7:2	Reserved
				Bits	Description			
7:2	Reserved							
<table border="1"> <tbody> <tr> <td>1</td> <td> <b>Host-Initiated Refresh Support (HIRS):</b> If this bit is set to '1', then the controller supports the Host-Initiated Refresh capability (refer to section 8.1.11). If cleared to '0', then the controller does not support the Host-Initiated Refresh capability.                       This bit shall be cleared to '0' if the controller does not support the Device Self-test command (i.e., the Device Self-test Supported (DSTS) bit in the OACS field is cleared to '0').                       If the controller does not support the Device Self-test command (i.e., the DSTS bit in the OACS field is cleared to '0'), then This bit shall be cleared to '0'.                 </td> </tr> <tr> <td>0</td> <td> <b>Single Device Self-test Operation (SDSO):</b> If this bit is set to '1', then the NVM subsystem supports only one device self-test operation in progress at a time. If this bit is cleared to '0', then the NVM subsystem supports one device self-test operation per controller at a time.                 </td> </tr> </tbody> </table>	1	<b>Host-Initiated Refresh Support (HIRS):</b> If this bit is set to '1', then the controller supports the Host-Initiated Refresh capability (refer to section 8.1.11). If cleared to '0', then the controller does not support the Host-Initiated Refresh capability.  This bit shall be cleared to '0' if the controller does not support the Device Self-test command (i.e., the Device Self-test Supported (DSTS) bit in the OACS field is cleared to '0').  If the controller does not support the Device Self-test command (i.e., the DSTS bit in the OACS field is cleared to '0'), then This bit shall be cleared to '0'.	0	<b>Single Device Self-test Operation (SDSO):</b> If this bit is set to '1', then the NVM subsystem supports only one device self-test operation in progress at a time. If this bit is cleared to '0', then the NVM subsystem supports one device self-test operation per controller at a time.				
1	<b>Host-Initiated Refresh Support (HIRS):</b> If this bit is set to '1', then the controller supports the Host-Initiated Refresh capability (refer to section 8.1.11). If cleared to '0', then the controller does not support the Host-Initiated Refresh capability.  This bit shall be cleared to '0' if the controller does not support the Device Self-test command (i.e., the Device Self-test Supported (DSTS) bit in the OACS field is cleared to '0').  If the controller does not support the Device Self-test command (i.e., the DSTS bit in the OACS field is cleared to '0'), then This bit shall be cleared to '0'.							
0	<b>Single Device Self-test Operation (SDSO):</b> If this bit is set to '1', then the NVM subsystem supports only one device self-test operation in progress at a time. If this bit is cleared to '0', then the NVM subsystem supports one device self-test operation per controller at a time.							

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description																			
327:326	O	O	R	<p><b>Maximum Thermal Management Temperature (MXTMT):</b> This field indicates the maximum temperature, in Kelvins, that the host may request in the Thermal Management Temperature 1 field and Thermal Management Temperature 2 field of the Set Features command with the Feature Identifier set to 10h. A value of 0h indicates that the controller does not report this field or the host controlled thermal management feature is not supported.</p>																			
331:328	O	O	R	<p><b>Sanitize Capabilities (SANICAP):</b> This field indicates attributes for sanitize operations. If the Sanitize command is supported, then this field shall be non-zero. If the Sanitize command is not supported, then this field shall be cleared to 0h. Refer to section 8.1.24.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td rowspan="5">31:30</td> <td> <p><b>No-Deallocate Modifies Media After Sanitize (NODMMAS):</b> This field indicates if media is additionally modified by the controller as part of sanitize processing that had been started by a Sanitize command with the No-Deallocate After Sanitize bit set to '1'.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Additional media modification as part of sanitize processing is not defined. Only controllers compliant with NVM Express Base Specification, Revision 1.3 and earlier, or controllers that do not support the Sanitize command are allowed to return this value.</td> </tr> <tr> <td>01b</td> <td>Media shall not be additionally modified by the controller as part of sanitize processing.</td> </tr> <tr> <td>10b</td> <td>Media shall be additionally modified by the controller as part of sanitize processing that had been started by a Sanitize command with the No-Deallocate After Sanitize bit set to '1'.</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table> </td> </tr> <tr> <td> <p><b>No-Deallocate Inhibited (NDI):</b> If this bit is set to '1' and the No-Deallocate Response Mode bit (refer to Figure 408) is set to '1', then the controller deallocates all media allocated for user data before the Restricted Processing:Idle transition occurs or the Unrestricted Processing:Idle transition occurs (refer to section 8.1.24.3), even if the No-Deallocate After Sanitize bit is set to '1' in a Sanitize command.</p> <p>If:</p> <ul style="list-style-type: none"> <li>a) this bit is set to '1';</li> <li>b) the No-Deallocate After Sanitize bit is set to '1' in a Sanitize command, and: <ul style="list-style-type: none"> <li>1) the No-Deallocate Response Mode bit is cleared to '0'; or</li> <li>2) the Sanitize Config Feature (refer to section 5.1.25.1.15) is not supported,</li> </ul> </li> </ul> <p>then the controller aborts the Sanitize command with a status code of Invalid Field in Command.</p> <p>If the No-Deallocate After Sanitize bit is cleared to '0' in a Sanitize command, then the value of this bit has no effect on the processing of that Sanitize command or on the sanitize operation that is started by that Sanitize command.</p> <p>If this bit is cleared to '0', then the controller supports the No-Deallocate After Sanitize bit in a Sanitize command.</p> </td> </tr> <tr> <td>28:04</td> <td colspan="3">Reserved</td> </tr> </tbody> </table>	Bits	Description	31:30	<p><b>No-Deallocate Modifies Media After Sanitize (NODMMAS):</b> This field indicates if media is additionally modified by the controller as part of sanitize processing that had been started by a Sanitize command with the No-Deallocate After Sanitize bit set to '1'.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Additional media modification as part of sanitize processing is not defined. Only controllers compliant with NVM Express Base Specification, Revision 1.3 and earlier, or controllers that do not support the Sanitize command are allowed to return this value.</td> </tr> <tr> <td>01b</td> <td>Media shall not be additionally modified by the controller as part of sanitize processing.</td> </tr> <tr> <td>10b</td> <td>Media shall be additionally modified by the controller as part of sanitize processing that had been started by a Sanitize command with the No-Deallocate After Sanitize bit set to '1'.</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	Additional media modification as part of sanitize processing is not defined. Only controllers compliant with NVM Express Base Specification, Revision 1.3 and earlier, or controllers that do not support the Sanitize command are allowed to return this value.	01b	Media shall not be additionally modified by the controller as part of sanitize processing.	10b	Media shall be additionally modified by the controller as part of sanitize processing that had been started by a Sanitize command with the No-Deallocate After Sanitize bit set to '1'.	11b	Reserved	<p><b>No-Deallocate Inhibited (NDI):</b> If this bit is set to '1' and the No-Deallocate Response Mode bit (refer to Figure 408) is set to '1', then the controller deallocates all media allocated for user data before the Restricted Processing:Idle transition occurs or the Unrestricted Processing:Idle transition occurs (refer to section 8.1.24.3), even if the No-Deallocate After Sanitize bit is set to '1' in a Sanitize command.</p> <p>If:</p> <ul style="list-style-type: none"> <li>a) this bit is set to '1';</li> <li>b) the No-Deallocate After Sanitize bit is set to '1' in a Sanitize command, and: <ul style="list-style-type: none"> <li>1) the No-Deallocate Response Mode bit is cleared to '0'; or</li> <li>2) the Sanitize Config Feature (refer to section 5.1.25.1.15) is not supported,</li> </ul> </li> </ul> <p>then the controller aborts the Sanitize command with a status code of Invalid Field in Command.</p> <p>If the No-Deallocate After Sanitize bit is cleared to '0' in a Sanitize command, then the value of this bit has no effect on the processing of that Sanitize command or on the sanitize operation that is started by that Sanitize command.</p> <p>If this bit is cleared to '0', then the controller supports the No-Deallocate After Sanitize bit in a Sanitize command.</p>	28:04	Reserved		
				Bits	Description																		
31:30	<p><b>No-Deallocate Modifies Media After Sanitize (NODMMAS):</b> This field indicates if media is additionally modified by the controller as part of sanitize processing that had been started by a Sanitize command with the No-Deallocate After Sanitize bit set to '1'.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Additional media modification as part of sanitize processing is not defined. Only controllers compliant with NVM Express Base Specification, Revision 1.3 and earlier, or controllers that do not support the Sanitize command are allowed to return this value.</td> </tr> <tr> <td>01b</td> <td>Media shall not be additionally modified by the controller as part of sanitize processing.</td> </tr> <tr> <td>10b</td> <td>Media shall be additionally modified by the controller as part of sanitize processing that had been started by a Sanitize command with the No-Deallocate After Sanitize bit set to '1'.</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	Additional media modification as part of sanitize processing is not defined. Only controllers compliant with NVM Express Base Specification, Revision 1.3 and earlier, or controllers that do not support the Sanitize command are allowed to return this value.	01b		Media shall not be additionally modified by the controller as part of sanitize processing.	10b	Media shall be additionally modified by the controller as part of sanitize processing that had been started by a Sanitize command with the No-Deallocate After Sanitize bit set to '1'.	11b	Reserved											
	Value	Definition																					
	00b	Additional media modification as part of sanitize processing is not defined. Only controllers compliant with NVM Express Base Specification, Revision 1.3 and earlier, or controllers that do not support the Sanitize command are allowed to return this value.																					
	01b	Media shall not be additionally modified by the controller as part of sanitize processing.																					
	10b	Media shall be additionally modified by the controller as part of sanitize processing that had been started by a Sanitize command with the No-Deallocate After Sanitize bit set to '1'.																					
11b	Reserved																						
<p><b>No-Deallocate Inhibited (NDI):</b> If this bit is set to '1' and the No-Deallocate Response Mode bit (refer to Figure 408) is set to '1', then the controller deallocates all media allocated for user data before the Restricted Processing:Idle transition occurs or the Unrestricted Processing:Idle transition occurs (refer to section 8.1.24.3), even if the No-Deallocate After Sanitize bit is set to '1' in a Sanitize command.</p> <p>If:</p> <ul style="list-style-type: none"> <li>a) this bit is set to '1';</li> <li>b) the No-Deallocate After Sanitize bit is set to '1' in a Sanitize command, and: <ul style="list-style-type: none"> <li>1) the No-Deallocate Response Mode bit is cleared to '0'; or</li> <li>2) the Sanitize Config Feature (refer to section 5.1.25.1.15) is not supported,</li> </ul> </li> </ul> <p>then the controller aborts the Sanitize command with a status code of Invalid Field in Command.</p> <p>If the No-Deallocate After Sanitize bit is cleared to '0' in a Sanitize command, then the value of this bit has no effect on the processing of that Sanitize command or on the sanitize operation that is started by that Sanitize command.</p> <p>If this bit is cleared to '0', then the controller supports the No-Deallocate After Sanitize bit in a Sanitize command.</p>																							
28:04	Reserved																						

**Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent**

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description								
				<table border="1"> <tr> <td>03</td> <td><b>Verification Support (VERS):</b> If this bit is set to '1', then the controller supports the Media Verification state and the Post-Verification Deallocation state. If this bit is cleared to '0', then the controller does not support the Media Verification state and does not support the Post-Verification Deallocation state.  If the BES bit is cleared to '0' and the CES bit is cleared to '0', then this bit shall be cleared to '0'.</td> </tr> <tr> <td>02</td> <td><b>Overwrite Support (OWS):</b> If this bit is set to '1', then the controller supports the Overwrite sanitize operation. If this bit is cleared to '0', then the controller does not support the Overwrite sanitize operation.</td> </tr> <tr> <td>01</td> <td><b>Block Erase Support (BES):</b> If this bit is set to '1', then the controller supports the Block Erase sanitize operation. If this bit is cleared to '0', then the controller does not support the Block Erase sanitize operation.</td> </tr> <tr> <td>00</td> <td><b>Crypto Erase Support (CES):</b> If this bit is set to '1', then the controller supports the Crypto Erase sanitize operation. If this bit is cleared to '0', then the controller does not support the Crypto Erase sanitize operation.</td> </tr> </table>	03	<b>Verification Support (VERS):</b> If this bit is set to '1', then the controller supports the Media Verification state and the Post-Verification Deallocation state. If this bit is cleared to '0', then the controller does not support the Media Verification state and does not support the Post-Verification Deallocation state.  If the BES bit is cleared to '0' and the CES bit is cleared to '0', then this bit shall be cleared to '0'.	02	<b>Overwrite Support (OWS):</b> If this bit is set to '1', then the controller supports the Overwrite sanitize operation. If this bit is cleared to '0', then the controller does not support the Overwrite sanitize operation.	01	<b>Block Erase Support (BES):</b> If this bit is set to '1', then the controller supports the Block Erase sanitize operation. If this bit is cleared to '0', then the controller does not support the Block Erase sanitize operation.	00	<b>Crypto Erase Support (CES):</b> If this bit is set to '1', then the controller supports the Crypto Erase sanitize operation. If this bit is cleared to '0', then the controller does not support the Crypto Erase sanitize operation.
03	<b>Verification Support (VERS):</b> If this bit is set to '1', then the controller supports the Media Verification state and the Post-Verification Deallocation state. If this bit is cleared to '0', then the controller does not support the Media Verification state and does not support the Post-Verification Deallocation state.  If the BES bit is cleared to '0' and the CES bit is cleared to '0', then this bit shall be cleared to '0'.											
02	<b>Overwrite Support (OWS):</b> If this bit is set to '1', then the controller supports the Overwrite sanitize operation. If this bit is cleared to '0', then the controller does not support the Overwrite sanitize operation.											
01	<b>Block Erase Support (BES):</b> If this bit is set to '1', then the controller supports the Block Erase sanitize operation. If this bit is cleared to '0', then the controller does not support the Block Erase sanitize operation.											
00	<b>Crypto Erase Support (CES):</b> If this bit is set to '1', then the controller supports the Crypto Erase sanitize operation. If this bit is cleared to '0', then the controller does not support the Crypto Erase sanitize operation.											
335:332	O	O	R	<b>Host Memory Buffer Minimum Descriptor Entry Size (HMMINDS):</b> This field indicates the minimum usable size of a Host Memory Buffer Descriptor Entry in 4 KiB units. If this field is cleared to 0h, then the controller does not indicate any limitations on the Host Memory Buffer Descriptor Entry size.								
337:336	O	O	R	<b>Host Memory Maximum Descriptors Entries (HMMAXD):</b> This field indicates the number of usable Host Memory Buffer Descriptor Entries. If this field is cleared to 0h, then the controller does not indicate a maximum number of Host Memory Buffer Descriptor Entries.								
339:338	O	O	R	<b>NVM Set Identifier Maximum (NSETIDMAX):</b> This field defines the maximum value of a valid NVM Set Identifier for any controller in the NVM subsystem. The number of NVM Sets supported by the NVM subsystem is less than or equal to NSETIDMAX.								
341:340	O	O	R	<b>Endurance Group Identifier Maximum (ENDGIDMAX):</b> This field defines the maximum value of a valid Endurance Group Identifier for any controller in the NVM subsystem. The number of Endurance Groups supported by the NVM subsystem is less than or equal to ENDGIDMAX.								
342	O	O	R	<b>ANA Transition Time (ANATT):</b> This field indicates the maximum amount of time, in seconds, for a transition between ANA states or the maximum amount of time, in seconds, that the controller reports the ANA change state. If the controller supports Asymmetric Namespace Access Reporting (refer to the CMIC field in Figure 312), then this field shall be set to a non-zero value. If the controller does not support Asymmetric Namespace Access Reporting, then this field shall be cleared to 0h. Refer to section 8.1.2.4.								

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description																		
343	O	O	R	<p><b>Asymmetric Namespace Access Capabilities (ANACAP):</b> This field indicates the capabilities associated with Asymmetric Namespace Access Reporting (refer to section 8.1.1).</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7</td> <td><b>ANA Group ID Support (ANAGIDS):</b> If this bit is set to '1', then the controller supports a non-zero value in the ANAGRPID field of the Namespace Management command. If this bit is cleared to '0', then the controller does not support a non-zero value in the ANAGRPID field of the Namespace Management command. If the Namespace Management command is not supported, then this bit shall be cleared to '0'.</td> </tr> <tr> <td>6</td> <td><b>ANA Group ID Locked When Attached Support (ANAGIDLWAS):</b> If this bit is set to '1', then the ANAGRPID field in the Identify Namespace data structure (refer to the NVM Command Set Specification) does not change while the namespace is attached to any controller. If this bit is cleared to '0', then the ANAGRPID field may change while the namespace is attached to any controller. Refer to section 8.1.1.2.</td> </tr> <tr> <td>5</td> <td>Reserved</td> </tr> <tr> <td>4</td> <td><b>Report ANA Change State (RANACS) :</b> If this bit is set to '1', then the controller is able to report ANA Change state (refer to section 8.1.1.8). If this bit is cleared to '0', then the controller does not report ANA Change state.</td> </tr> <tr> <td>3</td> <td><b>Report ANA Persistent Loss State (RANAPLS):</b> If this bit is set to '1', then the controller is able to report ANA Persistent Loss state (refer to section 8.1.1.7). If this bit is cleared to '0', then the controller does not report ANA Persistent Loss state.</td> </tr> <tr> <td>2</td> <td><b>Report ANA Inaccessible State (RANAIS):</b> If this bit is set to '1', then the controller is able to report ANA Inaccessible state (refer to section 8.1.1.6). If this bit is cleared to '0', then the controller does not report ANA Inaccessible state.</td> </tr> <tr> <td>1</td> <td><b>Report ANA Non-Optimized State (RANANOS):</b> If this bit is set to '1', then the controller is able to report ANA Non-Optimized state (refer to section 8.1.1.5). If this bit is cleared to '0', then the controller does not report ANA Non-Optimized state.</td> </tr> <tr> <td>0</td> <td><b>Report ANA Optimized State (RANAOS):</b> If this bit is set to '1', then the controller is able to report ANA Inaccessible state (refer to section 8.1.1.4). If this bit is cleared to '0', then the controller does not report ANA Inaccessible state.</td> </tr> </tbody> </table>	Bits	Description	7	<b>ANA Group ID Support (ANAGIDS):</b> If this bit is set to '1', then the controller supports a non-zero value in the ANAGRPID field of the Namespace Management command. If this bit is cleared to '0', then the controller does not support a non-zero value in the ANAGRPID field of the Namespace Management command. If the Namespace Management command is not supported, then this bit shall be cleared to '0'.	6	<b>ANA Group ID Locked When Attached Support (ANAGIDLWAS):</b> If this bit is set to '1', then the ANAGRPID field in the Identify Namespace data structure (refer to the NVM Command Set Specification) does not change while the namespace is attached to any controller. If this bit is cleared to '0', then the ANAGRPID field may change while the namespace is attached to any controller. Refer to section 8.1.1.2.	5	Reserved	4	<b>Report ANA Change State (RANACS) :</b> If this bit is set to '1', then the controller is able to report ANA Change state (refer to section 8.1.1.8). If this bit is cleared to '0', then the controller does not report ANA Change state.	3	<b>Report ANA Persistent Loss State (RANAPLS):</b> If this bit is set to '1', then the controller is able to report ANA Persistent Loss state (refer to section 8.1.1.7). If this bit is cleared to '0', then the controller does not report ANA Persistent Loss state.	2	<b>Report ANA Inaccessible State (RANAIS):</b> If this bit is set to '1', then the controller is able to report ANA Inaccessible state (refer to section 8.1.1.6). If this bit is cleared to '0', then the controller does not report ANA Inaccessible state.	1	<b>Report ANA Non-Optimized State (RANANOS):</b> If this bit is set to '1', then the controller is able to report ANA Non-Optimized state (refer to section 8.1.1.5). If this bit is cleared to '0', then the controller does not report ANA Non-Optimized state.	0	<b>Report ANA Optimized State (RANAOS):</b> If this bit is set to '1', then the controller is able to report ANA Inaccessible state (refer to section 8.1.1.4). If this bit is cleared to '0', then the controller does not report ANA Inaccessible state.
				Bits	Description																	
				7	<b>ANA Group ID Support (ANAGIDS):</b> If this bit is set to '1', then the controller supports a non-zero value in the ANAGRPID field of the Namespace Management command. If this bit is cleared to '0', then the controller does not support a non-zero value in the ANAGRPID field of the Namespace Management command. If the Namespace Management command is not supported, then this bit shall be cleared to '0'.																	
				6	<b>ANA Group ID Locked When Attached Support (ANAGIDLWAS):</b> If this bit is set to '1', then the ANAGRPID field in the Identify Namespace data structure (refer to the NVM Command Set Specification) does not change while the namespace is attached to any controller. If this bit is cleared to '0', then the ANAGRPID field may change while the namespace is attached to any controller. Refer to section 8.1.1.2.																	
				5	Reserved																	
				4	<b>Report ANA Change State (RANACS) :</b> If this bit is set to '1', then the controller is able to report ANA Change state (refer to section 8.1.1.8). If this bit is cleared to '0', then the controller does not report ANA Change state.																	
				3	<b>Report ANA Persistent Loss State (RANAPLS):</b> If this bit is set to '1', then the controller is able to report ANA Persistent Loss state (refer to section 8.1.1.7). If this bit is cleared to '0', then the controller does not report ANA Persistent Loss state.																	
				2	<b>Report ANA Inaccessible State (RANAIS):</b> If this bit is set to '1', then the controller is able to report ANA Inaccessible state (refer to section 8.1.1.6). If this bit is cleared to '0', then the controller does not report ANA Inaccessible state.																	
1	<b>Report ANA Non-Optimized State (RANANOS):</b> If this bit is set to '1', then the controller is able to report ANA Non-Optimized state (refer to section 8.1.1.5). If this bit is cleared to '0', then the controller does not report ANA Non-Optimized state.																					
0	<b>Report ANA Optimized State (RANAOS):</b> If this bit is set to '1', then the controller is able to report ANA Inaccessible state (refer to section 8.1.1.4). If this bit is cleared to '0', then the controller does not report ANA Inaccessible state.																					
347:344	O	O	R	<b>ANA Group Identifier Maximum (ANAGRPMAX):</b> This field indicates the maximum value of a valid ANA Group Identifier for any controller in the NVM subsystem. If the controller supports Asymmetric Namespace Access Reporting (refer to the CMIC field in Figure 312), then this field shall be set to a non-zero value. If the controller does not support Asymmetric Namespace Access Reporting, then this field shall be cleared to 0h.																		
351:348	O	O	R	<b>Number of ANA Group Identifiers (NANAGRPID):</b> This field indicates the number of ANA groups supported by the controller. If the controller supports Asymmetric Namespace Access Reporting (refer to the CMIC field in Figure 312), then this field shall be set to a non-zero value that is less than or equal to the ANAGRPMAX value. If the controller does not support Asymmetric Namespace Access Reporting, then this field shall be cleared to 0h.																		
355:352	O	O	R	<b>Persistent Event Log Size (PELS):</b> This field indicates the maximum reportable size for the Persistent Event Log (Refer to section 5.1.12.1.14) in 64 KiB units. If the Persistent Event Log is not supported, then this field is reserved.																		
357:356	O	O	O	<b>Domain Identifier (DID):</b> This field indicates the identifier of the domain (refer to section 3.2.5.3) that contains this controller. If the MDS bit is set to '1', then this field shall be set to a non-zero value. If the NVM subsystem does not support multiple domains (i.e., the NVM subsystem consists of a single domain), then this field shall be cleared to 0h.																		

**Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent**

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description								
358	O	P	P	<p><b>Key Per I/O Capabilities (KPIOC):</b> This field indicates the attributes for the Key Per I/O capability (refer to section 8.1.11).</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:2</td> <td>Reserved</td> </tr> <tr> <td>1</td> <td> <p><b>Key Per I/O Scope (KPIOSC):</b> If this bit is set to '1', then the Key Per I/O capability applies to all namespaces in the NVM subsystem when the Key Per I/O capability is enabled. If this bit is cleared to '0', then the Key Per I/O capability does not apply to all namespaces in the NVM subsystem and is allowed to be independently enabled and disabled uniquely on each namespace within the NVM subsystem. If the KPIOS bit is cleared to '0', then this bit should be ignored by the host.</p> </td> </tr> <tr> <td>0</td> <td> <p><b>Key Per I/O Supported (KPIOS):</b> If this bit is set to '1', then the controller supports the Key Per I/O capability. If this bit is cleared to '0', then the controller does not support the Key Per I/O capability.</p> </td> </tr> </tbody> </table>	Bits	Description	7:2	Reserved	1	<p><b>Key Per I/O Scope (KPIOSC):</b> If this bit is set to '1', then the Key Per I/O capability applies to all namespaces in the NVM subsystem when the Key Per I/O capability is enabled. If this bit is cleared to '0', then the Key Per I/O capability does not apply to all namespaces in the NVM subsystem and is allowed to be independently enabled and disabled uniquely on each namespace within the NVM subsystem. If the KPIOS bit is cleared to '0', then this bit should be ignored by the host.</p>	0	<p><b>Key Per I/O Supported (KPIOS):</b> If this bit is set to '1', then the controller supports the Key Per I/O capability. If this bit is cleared to '0', then the controller does not support the Key Per I/O capability.</p>
				Bits	Description							
				7:2	Reserved							
				1	<p><b>Key Per I/O Scope (KPIOSC):</b> If this bit is set to '1', then the Key Per I/O capability applies to all namespaces in the NVM subsystem when the Key Per I/O capability is enabled. If this bit is cleared to '0', then the Key Per I/O capability does not apply to all namespaces in the NVM subsystem and is allowed to be independently enabled and disabled uniquely on each namespace within the NVM subsystem. If the KPIOS bit is cleared to '0', then this bit should be ignored by the host.</p>							
0	<p><b>Key Per I/O Supported (KPIOS):</b> If this bit is set to '1', then the controller supports the Key Per I/O capability. If this bit is cleared to '0', then the controller does not support the Key Per I/O capability.</p>											
359				Reserved								
361:360	O	O	R	<p><b>Maximum Processing Time for Firmware Activation Without Reset (MPTFAWR):</b> This field shall indicate the estimated maximum time in 100 ms units required by the controller to process a Firmware Commit command that specifies a value of 011b in the Commit Action field (i.e., firmware activation without reset).</p> <p>This field applies to Firmware Commit commands received on an NVM Express controller Admin Submission Queue or received out-of-band on a Management Endpoint (refer to the NVM Express Management Interface Specification).</p> <p>If firmware activation without reset is not supported, then this field shall be cleared to 0h. A value of 0h indicates that no time is indicated. This field shall not be cleared to 0h on implementations that support firmware activation without reset and that are compliant with revision 2.1 or later of this specification.</p>								
				367:362				Reserved				
383:368	O	R	R	<p><b>Max Endurance Group Capacity (MEGCAP):</b> This field indicates the maximum capacity of a single Endurance Group. If this field is cleared to 0h, then the NVM subsystem does not report a maximum Endurance Group Capacity value.</p>								
384	O	O	O	<p><b>Temperature Threshold Hysteresis Attributes (TMPTHHA):</b> This field indicates attributes related to temperature threshold hysteresis. Refer to section 5.1.25.1.3.1.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:3</td> <td>Reserved</td> </tr> <tr> <td>2:0</td> <td> <p><b>Temperature Threshold Maximum Hysteresis (TMPTHMH):</b> This field indicates the maximum temperature hysteresis value in Kelvins that is supported by the controller. This is the absolute value of the difference.</p> <p>If the controller does not support this attribute, this field shall be cleared to 000b.</p> </td> </tr> </tbody> </table>	Bits	Description	7:3	Reserved	2:0	<p><b>Temperature Threshold Maximum Hysteresis (TMPTHMH):</b> This field indicates the maximum temperature hysteresis value in Kelvins that is supported by the controller. This is the absolute value of the difference.</p> <p>If the controller does not support this attribute, this field shall be cleared to 000b.</p>		
				Bits	Description							
				7:3	Reserved							
2:0	<p><b>Temperature Threshold Maximum Hysteresis (TMPTHMH):</b> This field indicates the maximum temperature hysteresis value in Kelvins that is supported by the controller. This is the absolute value of the difference.</p> <p>If the controller does not support this attribute, this field shall be cleared to 000b.</p>											
385				Reserved								
387:386	M	M	M	<p><b>Command Quiesce Time (CQT):</b> This field indicates the expected worst-case time in 1 millisecond units for the controller to quiesce all outstanding commands (i.e., the controller shall satisfy all Immediate Abort requirements for those commands (refer to section 5.1.1.1)) after a Keep Alive Timeout (refer to section 3.9) or other communication loss (refer to section 9.6). If this field is cleared to 0h, then a command quiesce time is not reported. If the controller does not require any time to quiesce, the controller should set this field to 1h (i.e., 1 millisecond).</p>								
511:388				Reserved								

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description
<b>NVM Command Set Attributes</b>				
512	M	M	R	<b>Submission Queue Entry Size (SQES):</b> This field defines the required and maximum I/O Submission Queue entry size.
				<b>Bits</b>   <b>Description</b>
				7:4   <b>Maximum I/O Submission Queue Entry Size (MAXSQES):</b> This field identifies the maximum I/O Submission Queue entry size when using the NVM Command Set. This value is greater than or equal to the required SQ entry size (i.e., the MIOSQES field). The value is in bytes and is reported as a power of two (2 <sup>n</sup> ). The recommended value is 6, corresponding to a standard SQ entry size of 64 bytes. Controllers that implement proprietary extensions may support a larger value.
3:0   <b>Minimum I/O Submission Queue Entry Size (MINSQES):</b> This field identifies the required (i.e., minimum) I/O Submission Queue entry size. This is the minimum entry size that may be used. The value is in bytes and is reported as a power of two (2 <sup>n</sup> ). The required value shall be 6, corresponding to 64.				
513	M	M	R	<b>Completion Queue Entry Size (CQES):</b> This field defines the required and maximum I/O Completion Queue entry size.
				<b>Bits</b>   <b>Description</b>
				7:4   <b>Maximum I/O Completion Queue Entry Size (MAXCQES):</b> This field identifies the maximum I/O Completion Queue entry size. This value is greater than or equal to the required CQ entry size (i.e., the MIOCQES field). The value is in bytes and is reported as a power of two (2 <sup>n</sup> ). The recommended value is 4, corresponding to a standard CQ entry size of 16 bytes. Controllers that implement proprietary extensions may support a larger value.
3:0   <b>Minimum I/O Completion Queue Entry Size (MINCQES):</b> This field identifies the required (i.e., minimum) I/O Completion Queue entry size. This is the minimum entry size that may be used. The value is in bytes and is reported as a power of two (2 <sup>n</sup> ). The required value shall be 4, corresponding to 16.				
515:514	M	M	M	<b>Maximum Outstanding Commands (MAXCMD):</b> Indicates the maximum number of commands that the controller processes at one time for a particular queue (which may be larger than the size of the corresponding Submission Queue). The host may use this value to size Completion Queues and optimize the number of commands submitted at one time to a particular I/O Queue. This field is mandatory for NVMe over Fabrics implementations and optional for NVMe over PCIe implementations. If the field is not used, it shall be cleared to 0h.
519:516	M	M	R	<b>Number of Namespaces (NN):</b> This field indicates the maximum value of a valid NSID for the NVM subsystem. Refer to the MNAN field for the number of supported namespaces in the NVM subsystem.



521:520	M	M	R	<p><b>Optional NVM Command Support (ONCS):</b> This field indicates the optional I/O commands and features supported by the controller. Refer to section 3.1.3.</p> <p>If a controller supports more than one I/O Command Set, then the bits in the field associated with command support reflect the aggregate support across all of the I/O Command Sets supported or I/O Command Sets inherited from I/O Command Sets (e.g., Zoned Namespace Command Set). The Commands Supported and Effects log page (refer to section 5.1.12.1.6) may be used to determine the support of commands for a specific I/O Command Set.</p>																		
				<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:13</td> <td>Reserved</td> </tr> <tr> <td>12</td> <td> <p><b>Namespace Zeroes Support (NSZS):</b> If this bit is set to '1', then the controller supports the Namespace Zeroes (NSZ) bit in the NVM Command Set Write Zeroes command. If this bit is cleared to '0', then the controller does not support the Namespace Zeroes (NSZ) bit in the Write Zeroes command. If the Write Zeroes command is not supported, then this bit shall be cleared to '0'.</p> </td> </tr> <tr> <td>11</td> <td> <p><b>Maximum Write Zeroes with Deallocate (MAXWZD):</b> If this bit is set to '1', then the maximum data size for the NVM Command Set Write Zeroes command depends on the value of the Deallocate bit in the Write Zeroes command and the value in the WZDSL field in the I/O Command Set specific Identify Controller data structure for the NVM Command Set (refer to the I/O Command Set specific Identify Controller Data Structure (CNS 06h, CSI 00h) section in the NVM Command Set Specification). If this bit is cleared to '0', then this bit has no effect. If the Write Zeroes command is not supported or the WZSL field in that data structure is cleared to 0h, then this bit shall be cleared to '0'.</p> </td> </tr> <tr> <td>10</td> <td> <p><b>NVM All Fast Copy (NVMAFC):</b> If this bit is set to '1', then within this NVM subsystem, all copy operations performed by the controller are fast copy operations (refer to the Fast copy operations section of the NVM Command Set Specification).</p> <p>If this bit is cleared to '0', then within this NVM subsystem, some copy operations performed by the controller may not be fast copy operations.</p> </td> </tr> <tr> <td>9</td> <td> <p><b>NVM Copy Single Atomicity (NVMCSA):</b> If this bit is set to '1', then the write portion of any NVM Command Set Copy command is performed as a single write command to which the atomicity requirements described in the Atomic operation section of the NVM Command Set Specification apply. If this bit is cleared to '0', then the atomicity behavior of the write portion of Copy commands is specified by the Copy atomicity section of the NVM Command Set Specification.</p> <p>This bit should be set to '1' if the controller supports the NVM Command Set Copy command. The Copy atomicity section of the NVM Command Set Specification specifies additional conditions under which this bit shall be set to '1'.</p> <p>This bit is ignored by the host if the controller does not support the NVM Command Set Copy command.</p> </td> </tr> <tr> <td>8</td> <td> <p><b>Copy Support (NVMCPYS):</b> If this bit is set to '1', then the controller supports the NVM Command Set Copy command. If this bit is cleared to '0', then the controller does not support the NVM Command Set Copy command.</p> </td> </tr> <tr> <td>7</td> <td> <p><b>Verify Support (NVMVFYS):</b> If this bit is set to '1', then the controller supports the NVM Command Set Verify command and the Verify Size Limit (VSL) field indicates the recommended maximum data size for Verify commands. If this bit is cleared to '0', then controller support of the NVM Command Set Verify command is indicated by a non-zero data size limit in the VSL field.</p> </td> </tr> <tr> <td>6</td> <td> <p><b>Timestamp Support (TSS):</b> If this bit is set to '1', then the controller supports the Timestamp feature. If this bit is cleared to '0', then the controller does not support the Timestamp feature. Refer to section 5.1.25.1.7.</p> </td> </tr> </tbody> </table>	Bits	Description	15:13	Reserved	12	<p><b>Namespace Zeroes Support (NSZS):</b> If this bit is set to '1', then the controller supports the Namespace Zeroes (NSZ) bit in the NVM Command Set Write Zeroes command. If this bit is cleared to '0', then the controller does not support the Namespace Zeroes (NSZ) bit in the Write Zeroes command. If the Write Zeroes command is not supported, then this bit shall be cleared to '0'.</p>	11	<p><b>Maximum Write Zeroes with Deallocate (MAXWZD):</b> If this bit is set to '1', then the maximum data size for the NVM Command Set Write Zeroes command depends on the value of the Deallocate bit in the Write Zeroes command and the value in the WZDSL field in the I/O Command Set specific Identify Controller data structure for the NVM Command Set (refer to the I/O Command Set specific Identify Controller Data Structure (CNS 06h, CSI 00h) section in the NVM Command Set Specification). If this bit is cleared to '0', then this bit has no effect. If the Write Zeroes command is not supported or the WZSL field in that data structure is cleared to 0h, then this bit shall be cleared to '0'.</p>	10	<p><b>NVM All Fast Copy (NVMAFC):</b> If this bit is set to '1', then within this NVM subsystem, all copy operations performed by the controller are fast copy operations (refer to the Fast copy operations section of the NVM Command Set Specification).</p> <p>If this bit is cleared to '0', then within this NVM subsystem, some copy operations performed by the controller may not be fast copy operations.</p>	9	<p><b>NVM Copy Single Atomicity (NVMCSA):</b> If this bit is set to '1', then the write portion of any NVM Command Set Copy command is performed as a single write command to which the atomicity requirements described in the Atomic operation section of the NVM Command Set Specification apply. If this bit is cleared to '0', then the atomicity behavior of the write portion of Copy commands is specified by the Copy atomicity section of the NVM Command Set Specification.</p> <p>This bit should be set to '1' if the controller supports the NVM Command Set Copy command. The Copy atomicity section of the NVM Command Set Specification specifies additional conditions under which this bit shall be set to '1'.</p> <p>This bit is ignored by the host if the controller does not support the NVM Command Set Copy command.</p>	8	<p><b>Copy Support (NVMCPYS):</b> If this bit is set to '1', then the controller supports the NVM Command Set Copy command. If this bit is cleared to '0', then the controller does not support the NVM Command Set Copy command.</p>	7	<p><b>Verify Support (NVMVFYS):</b> If this bit is set to '1', then the controller supports the NVM Command Set Verify command and the Verify Size Limit (VSL) field indicates the recommended maximum data size for Verify commands. If this bit is cleared to '0', then controller support of the NVM Command Set Verify command is indicated by a non-zero data size limit in the VSL field.</p>	6	<p><b>Timestamp Support (TSS):</b> If this bit is set to '1', then the controller supports the Timestamp feature. If this bit is cleared to '0', then the controller does not support the Timestamp feature. Refer to section 5.1.25.1.7.</p>
				Bits	Description																	
				15:13	Reserved																	
				12	<p><b>Namespace Zeroes Support (NSZS):</b> If this bit is set to '1', then the controller supports the Namespace Zeroes (NSZ) bit in the NVM Command Set Write Zeroes command. If this bit is cleared to '0', then the controller does not support the Namespace Zeroes (NSZ) bit in the Write Zeroes command. If the Write Zeroes command is not supported, then this bit shall be cleared to '0'.</p>																	
				11	<p><b>Maximum Write Zeroes with Deallocate (MAXWZD):</b> If this bit is set to '1', then the maximum data size for the NVM Command Set Write Zeroes command depends on the value of the Deallocate bit in the Write Zeroes command and the value in the WZDSL field in the I/O Command Set specific Identify Controller data structure for the NVM Command Set (refer to the I/O Command Set specific Identify Controller Data Structure (CNS 06h, CSI 00h) section in the NVM Command Set Specification). If this bit is cleared to '0', then this bit has no effect. If the Write Zeroes command is not supported or the WZSL field in that data structure is cleared to 0h, then this bit shall be cleared to '0'.</p>																	
				10	<p><b>NVM All Fast Copy (NVMAFC):</b> If this bit is set to '1', then within this NVM subsystem, all copy operations performed by the controller are fast copy operations (refer to the Fast copy operations section of the NVM Command Set Specification).</p> <p>If this bit is cleared to '0', then within this NVM subsystem, some copy operations performed by the controller may not be fast copy operations.</p>																	
				9	<p><b>NVM Copy Single Atomicity (NVMCSA):</b> If this bit is set to '1', then the write portion of any NVM Command Set Copy command is performed as a single write command to which the atomicity requirements described in the Atomic operation section of the NVM Command Set Specification apply. If this bit is cleared to '0', then the atomicity behavior of the write portion of Copy commands is specified by the Copy atomicity section of the NVM Command Set Specification.</p> <p>This bit should be set to '1' if the controller supports the NVM Command Set Copy command. The Copy atomicity section of the NVM Command Set Specification specifies additional conditions under which this bit shall be set to '1'.</p> <p>This bit is ignored by the host if the controller does not support the NVM Command Set Copy command.</p>																	
				8	<p><b>Copy Support (NVMCPYS):</b> If this bit is set to '1', then the controller supports the NVM Command Set Copy command. If this bit is cleared to '0', then the controller does not support the NVM Command Set Copy command.</p>																	
7	<p><b>Verify Support (NVMVFYS):</b> If this bit is set to '1', then the controller supports the NVM Command Set Verify command and the Verify Size Limit (VSL) field indicates the recommended maximum data size for Verify commands. If this bit is cleared to '0', then controller support of the NVM Command Set Verify command is indicated by a non-zero data size limit in the VSL field.</p>																					
6	<p><b>Timestamp Support (TSS):</b> If this bit is set to '1', then the controller supports the Timestamp feature. If this bit is cleared to '0', then the controller does not support the Timestamp feature. Refer to section 5.1.25.1.7.</p>																					

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description												
				<table border="1"> <tr> <td>5</td> <td><b>Reservations Support (RESERVS):</b> If this bit is set to '1', then the controller supports reservations. If this bit is cleared to '0', then the controller does not support reservations. If the controller supports reservations, then the following commands associated with reservations shall be supported: Reservation Report, Reservation Register, Reservation Acquire, and Reservation Release. Refer to section 8.1.22 for additional requirements.</td> </tr> <tr> <td>4</td> <td><b>Save and Select Feature Support (SSFS):</b> If this bit is set to '1', then the controller supports the Save field set to a non-zero value in the Set Features command and the Select field set to a non-zero value in the Get Features command. If this bit is cleared to '0', then the controller does not support the Save field set to a non-zero value in the Set Features command and does not support the Select field set to a non-zero value in the Get Features command.</td> </tr> <tr> <td>3</td> <td><b>Write Zeroes Support Variants (NVMWZSV):</b> If this bit is set to '1', then the controller supports the NVM Command Set Write Zeroes command and the Write Zeroes Size Limit (WZSL) field indicates the recommended maximum data size for Write Zeroes commands. If this bit is cleared to '0', then controller support of the NVM Command Set Write Zeroes command is indicated by a non-zero data size limit in the WZSL field.</td> </tr> <tr> <td>2</td> <td><b>Dataset Management Support Variants (NVMDSMSV):</b> If this bit is set to '1', then the controller supports the NVM Command Set Dataset Management command and limits, if any, on controller support of the Dataset Management command are indicated by non-zero values in the Dataset Management Ranges Limit (DMRL) field, the Dataset Management Size Limit (DMSL) field and the Dataset Management Range Size Limit (DMRSL) field. If this bit is cleared to '0', then controller support of the NVM Command Set Dataset Management command is indicated by a non-zero data size limit in the DMRL, DMSL, and DMRSL fields.</td> </tr> <tr> <td>1</td> <td><b>Write Uncorrectable Support Variants (NVMWUSV):</b> If this bit is set to '1', then the controller supports the NVM Command Set Write Uncorrectable command and the Write Uncorrectable Size Limit (WUSL) field indicates the recommended maximum data size for Write Uncorrectable commands. If this bit is cleared to '0', then controller support of the NVM Command Set Write Uncorrectable command is indicated by a non-zero data size limit in the WUSL field.</td> </tr> <tr> <td>0</td> <td><b>Compare Command Support (NVMCMPS):</b> If this bit is set to '1', then the controller supports the NVM Command Set Compare command. If this bit is cleared to '0', then the controller does not support the NVM Command Set Compare command.</td> </tr> </table> <p>NOTE: This field applies to all I/O Command Sets. The original name has been retained for historical continuity.</p>	5	<b>Reservations Support (RESERVS):</b> If this bit is set to '1', then the controller supports reservations. If this bit is cleared to '0', then the controller does not support reservations. If the controller supports reservations, then the following commands associated with reservations shall be supported: Reservation Report, Reservation Register, Reservation Acquire, and Reservation Release. Refer to section 8.1.22 for additional requirements.	4	<b>Save and Select Feature Support (SSFS):</b> If this bit is set to '1', then the controller supports the Save field set to a non-zero value in the Set Features command and the Select field set to a non-zero value in the Get Features command. If this bit is cleared to '0', then the controller does not support the Save field set to a non-zero value in the Set Features command and does not support the Select field set to a non-zero value in the Get Features command.	3	<b>Write Zeroes Support Variants (NVMWZSV):</b> If this bit is set to '1', then the controller supports the NVM Command Set Write Zeroes command and the Write Zeroes Size Limit (WZSL) field indicates the recommended maximum data size for Write Zeroes commands. If this bit is cleared to '0', then controller support of the NVM Command Set Write Zeroes command is indicated by a non-zero data size limit in the WZSL field.	2	<b>Dataset Management Support Variants (NVMDSMSV):</b> If this bit is set to '1', then the controller supports the NVM Command Set Dataset Management command and limits, if any, on controller support of the Dataset Management command are indicated by non-zero values in the Dataset Management Ranges Limit (DMRL) field, the Dataset Management Size Limit (DMSL) field and the Dataset Management Range Size Limit (DMRSL) field. If this bit is cleared to '0', then controller support of the NVM Command Set Dataset Management command is indicated by a non-zero data size limit in the DMRL, DMSL, and DMRSL fields.	1	<b>Write Uncorrectable Support Variants (NVMWUSV):</b> If this bit is set to '1', then the controller supports the NVM Command Set Write Uncorrectable command and the Write Uncorrectable Size Limit (WUSL) field indicates the recommended maximum data size for Write Uncorrectable commands. If this bit is cleared to '0', then controller support of the NVM Command Set Write Uncorrectable command is indicated by a non-zero data size limit in the WUSL field.	0	<b>Compare Command Support (NVMCMPS):</b> If this bit is set to '1', then the controller supports the NVM Command Set Compare command. If this bit is cleared to '0', then the controller does not support the NVM Command Set Compare command.
5	<b>Reservations Support (RESERVS):</b> If this bit is set to '1', then the controller supports reservations. If this bit is cleared to '0', then the controller does not support reservations. If the controller supports reservations, then the following commands associated with reservations shall be supported: Reservation Report, Reservation Register, Reservation Acquire, and Reservation Release. Refer to section 8.1.22 for additional requirements.															
4	<b>Save and Select Feature Support (SSFS):</b> If this bit is set to '1', then the controller supports the Save field set to a non-zero value in the Set Features command and the Select field set to a non-zero value in the Get Features command. If this bit is cleared to '0', then the controller does not support the Save field set to a non-zero value in the Set Features command and does not support the Select field set to a non-zero value in the Get Features command.															
3	<b>Write Zeroes Support Variants (NVMWZSV):</b> If this bit is set to '1', then the controller supports the NVM Command Set Write Zeroes command and the Write Zeroes Size Limit (WZSL) field indicates the recommended maximum data size for Write Zeroes commands. If this bit is cleared to '0', then controller support of the NVM Command Set Write Zeroes command is indicated by a non-zero data size limit in the WZSL field.															
2	<b>Dataset Management Support Variants (NVMDSMSV):</b> If this bit is set to '1', then the controller supports the NVM Command Set Dataset Management command and limits, if any, on controller support of the Dataset Management command are indicated by non-zero values in the Dataset Management Ranges Limit (DMRL) field, the Dataset Management Size Limit (DMSL) field and the Dataset Management Range Size Limit (DMRSL) field. If this bit is cleared to '0', then controller support of the NVM Command Set Dataset Management command is indicated by a non-zero data size limit in the DMRL, DMSL, and DMRSL fields.															
1	<b>Write Uncorrectable Support Variants (NVMWUSV):</b> If this bit is set to '1', then the controller supports the NVM Command Set Write Uncorrectable command and the Write Uncorrectable Size Limit (WUSL) field indicates the recommended maximum data size for Write Uncorrectable commands. If this bit is cleared to '0', then controller support of the NVM Command Set Write Uncorrectable command is indicated by a non-zero data size limit in the WUSL field.															
0	<b>Compare Command Support (NVMCMPS):</b> If this bit is set to '1', then the controller supports the NVM Command Set Compare command. If this bit is cleared to '0', then the controller does not support the NVM Command Set Compare command.															
523:522	M	M	R	<p><b>Fused Operation Support (FUSES):</b> This field indicates the fused operations that the controller supports. Refer to section 3.4.2.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td><b>Fused Compare and Write Supported (FCWS):</b> If this bit is set to '1', then the controller supports the NVM Command Set Compare and Write fused operation. If this bit is cleared to '0', then the controller does not support the NVM Command Set Compare and Write fused operation. Compare shall be the first command in the sequence.</td> </tr> </tbody> </table>	Bits	Description	7:1	Reserved	0	<b>Fused Compare and Write Supported (FCWS):</b> If this bit is set to '1', then the controller supports the NVM Command Set Compare and Write fused operation. If this bit is cleared to '0', then the controller does not support the NVM Command Set Compare and Write fused operation. Compare shall be the first command in the sequence.						
Bits	Description															
7:1	Reserved															
0	<b>Fused Compare and Write Supported (FCWS):</b> If this bit is set to '1', then the controller supports the NVM Command Set Compare and Write fused operation. If this bit is cleared to '0', then the controller does not support the NVM Command Set Compare and Write fused operation. Compare shall be the first command in the sequence.															

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description												
524	M	M	R	<p><b>Format NVM Attributes (FNA):</b> This field indicates attributes for the Format NVM command.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:4</td> <td>Reserved</td> </tr> <tr> <td>3</td> <td> <p><b>Format NVM Broadcast Support (FNVMB):</b> This bit indicates whether the Format NVM command supports an NSID value set to FFFFFFFFh. If this bit is set to '1', then the Format NVM command does not support an NSID value set to FFFFFFFFh. If this bit is cleared to '0', then the Format NVM command supports an NSID value set to FFFFFFFFh.</p> </td> </tr> <tr> <td>2</td> <td> <p><b>Cryptographic Erase Supported (CRYES):</b> This bit indicates whether cryptographic erase is supported as part of the secure erase functionality. If this bit is set to '1', then cryptographic erase is supported. If this bit is cleared to '0', then cryptographic erase is not supported.</p> </td> </tr> <tr> <td>1</td> <td> <p><b>Secure Erase Namespace Scope (SENS):</b> This bit indicates whether secure erase functionality applies to all namespaces in the NVM subsystem or is specific to a particular namespace. If this bit is set to '1', then any secure erase performed as part of a format operation results in a secure erase of all namespaces in the NVM subsystem. If this bit is cleared to '0', then any secure erase performed as part of a format results in a secure erase of the particular specified namespace. If the FNVMB bit is set to '1', then this bit shall be cleared to '0'.</p> </td> </tr> <tr> <td>0</td> <td> <p><b>Format Namespace Scope (FNS):</b> This bit indicates whether the format operation (excluding secure erase) applies to all namespaces in the NVM subsystem or is specific to a particular namespace. If this bit is set to '1', then all namespaces in the NVM subsystem shall be configured with the same attributes and a format (excluding secure erase) of any namespace results in a format of all namespaces in the NVM subsystem. If this bit is cleared to '0', then the controller supports format on a per namespace basis. If the FNVMB bit is set to '1', then this bit shall be cleared to '0'.</p> </td> </tr> </tbody> </table>	Bits	Description	7:4	Reserved	3	<p><b>Format NVM Broadcast Support (FNVMB):</b> This bit indicates whether the Format NVM command supports an NSID value set to FFFFFFFFh. If this bit is set to '1', then the Format NVM command does not support an NSID value set to FFFFFFFFh. If this bit is cleared to '0', then the Format NVM command supports an NSID value set to FFFFFFFFh.</p>	2	<p><b>Cryptographic Erase Supported (CRYES):</b> This bit indicates whether cryptographic erase is supported as part of the secure erase functionality. If this bit is set to '1', then cryptographic erase is supported. If this bit is cleared to '0', then cryptographic erase is not supported.</p>	1	<p><b>Secure Erase Namespace Scope (SENS):</b> This bit indicates whether secure erase functionality applies to all namespaces in the NVM subsystem or is specific to a particular namespace. If this bit is set to '1', then any secure erase performed as part of a format operation results in a secure erase of all namespaces in the NVM subsystem. If this bit is cleared to '0', then any secure erase performed as part of a format results in a secure erase of the particular specified namespace. If the FNVMB bit is set to '1', then this bit shall be cleared to '0'.</p>	0	<p><b>Format Namespace Scope (FNS):</b> This bit indicates whether the format operation (excluding secure erase) applies to all namespaces in the NVM subsystem or is specific to a particular namespace. If this bit is set to '1', then all namespaces in the NVM subsystem shall be configured with the same attributes and a format (excluding secure erase) of any namespace results in a format of all namespaces in the NVM subsystem. If this bit is cleared to '0', then the controller supports format on a per namespace basis. If the FNVMB bit is set to '1', then this bit shall be cleared to '0'.</p>
				Bits	Description											
				7:4	Reserved											
				3	<p><b>Format NVM Broadcast Support (FNVMB):</b> This bit indicates whether the Format NVM command supports an NSID value set to FFFFFFFFh. If this bit is set to '1', then the Format NVM command does not support an NSID value set to FFFFFFFFh. If this bit is cleared to '0', then the Format NVM command supports an NSID value set to FFFFFFFFh.</p>											
				2	<p><b>Cryptographic Erase Supported (CRYES):</b> This bit indicates whether cryptographic erase is supported as part of the secure erase functionality. If this bit is set to '1', then cryptographic erase is supported. If this bit is cleared to '0', then cryptographic erase is not supported.</p>											
1	<p><b>Secure Erase Namespace Scope (SENS):</b> This bit indicates whether secure erase functionality applies to all namespaces in the NVM subsystem or is specific to a particular namespace. If this bit is set to '1', then any secure erase performed as part of a format operation results in a secure erase of all namespaces in the NVM subsystem. If this bit is cleared to '0', then any secure erase performed as part of a format results in a secure erase of the particular specified namespace. If the FNVMB bit is set to '1', then this bit shall be cleared to '0'.</p>															
0	<p><b>Format Namespace Scope (FNS):</b> This bit indicates whether the format operation (excluding secure erase) applies to all namespaces in the NVM subsystem or is specific to a particular namespace. If this bit is set to '1', then all namespaces in the NVM subsystem shall be configured with the same attributes and a format (excluding secure erase) of any namespace results in a format of all namespaces in the NVM subsystem. If this bit is cleared to '0', then the controller supports format on a per namespace basis. If the FNVMB bit is set to '1', then this bit shall be cleared to '0'.</p>															

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description																		
525	M	M	R	<p><b>Volatile Write Cache (VWC):</b> This field indicates attributes related to the presence of a volatile write cache in the controller.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>07:03</td> <td>Reserved</td> </tr> <tr> <td>02:01</td> <td> <p><b>Flush Behavior (FB):</b> This field indicates Flush command behavior (refer to section 7.2) if the NSID value is set to FFFFFFFFh as follows:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Support for the NSID field set to FFFFFFFFh is not indicated. Only controllers compliant with NVM Express Base Specification revision 1.3 and earlier shall be allowed to return this value.</td> </tr> <tr> <td>01b</td> <td>Reserved</td> </tr> <tr> <td>10b</td> <td>The Flush command does not support the NSID field set to FFFFFFFFh. The controller shall fail a Flush command with the NSID set to FFFFFFFFh with a status code of Invalid Namespace or Format.</td> </tr> <tr> <td>11b</td> <td>The Flush command supports the NSID field set to FFFFFFFFh.</td> </tr> </tbody> </table> </td> </tr> <tr> <td>00</td> <td> <p><b>Volatile Write Cache Present (VWCP):</b> If this bit is set to '1', then a volatile write cache is present in the controller. If this bit is cleared to '0', then a volatile write cache is not present in the controller.</p> <p>If a volatile write cache is present, then the host controls whether the volatile write cache is enabled with a Set Features command specifying the Volatile Write Cache feature identifier (refer to section 5.1.25.1.4). The Flush command (refer to section 7.2) is used to request that the contents of a volatile write cache be made non-volatile.</p> </td> </tr> </tbody> </table>	Bits	Description	07:03	Reserved	02:01	<p><b>Flush Behavior (FB):</b> This field indicates Flush command behavior (refer to section 7.2) if the NSID value is set to FFFFFFFFh as follows:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Support for the NSID field set to FFFFFFFFh is not indicated. Only controllers compliant with NVM Express Base Specification revision 1.3 and earlier shall be allowed to return this value.</td> </tr> <tr> <td>01b</td> <td>Reserved</td> </tr> <tr> <td>10b</td> <td>The Flush command does not support the NSID field set to FFFFFFFFh. The controller shall fail a Flush command with the NSID set to FFFFFFFFh with a status code of Invalid Namespace or Format.</td> </tr> <tr> <td>11b</td> <td>The Flush command supports the NSID field set to FFFFFFFFh.</td> </tr> </tbody> </table>	Value	Definition	00b	Support for the NSID field set to FFFFFFFFh is not indicated. Only controllers compliant with NVM Express Base Specification revision 1.3 and earlier shall be allowed to return this value.	01b	Reserved	10b	The Flush command does not support the NSID field set to FFFFFFFFh. The controller shall fail a Flush command with the NSID set to FFFFFFFFh with a status code of Invalid Namespace or Format.	11b	The Flush command supports the NSID field set to FFFFFFFFh.	00	<p><b>Volatile Write Cache Present (VWCP):</b> If this bit is set to '1', then a volatile write cache is present in the controller. If this bit is cleared to '0', then a volatile write cache is not present in the controller.</p> <p>If a volatile write cache is present, then the host controls whether the volatile write cache is enabled with a Set Features command specifying the Volatile Write Cache feature identifier (refer to section 5.1.25.1.4). The Flush command (refer to section 7.2) is used to request that the contents of a volatile write cache be made non-volatile.</p>
Bits	Description																					
07:03	Reserved																					
02:01	<p><b>Flush Behavior (FB):</b> This field indicates Flush command behavior (refer to section 7.2) if the NSID value is set to FFFFFFFFh as follows:</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Support for the NSID field set to FFFFFFFFh is not indicated. Only controllers compliant with NVM Express Base Specification revision 1.3 and earlier shall be allowed to return this value.</td> </tr> <tr> <td>01b</td> <td>Reserved</td> </tr> <tr> <td>10b</td> <td>The Flush command does not support the NSID field set to FFFFFFFFh. The controller shall fail a Flush command with the NSID set to FFFFFFFFh with a status code of Invalid Namespace or Format.</td> </tr> <tr> <td>11b</td> <td>The Flush command supports the NSID field set to FFFFFFFFh.</td> </tr> </tbody> </table>	Value	Definition	00b	Support for the NSID field set to FFFFFFFFh is not indicated. Only controllers compliant with NVM Express Base Specification revision 1.3 and earlier shall be allowed to return this value.	01b	Reserved	10b	The Flush command does not support the NSID field set to FFFFFFFFh. The controller shall fail a Flush command with the NSID set to FFFFFFFFh with a status code of Invalid Namespace or Format.	11b	The Flush command supports the NSID field set to FFFFFFFFh.											
Value	Definition																					
00b	Support for the NSID field set to FFFFFFFFh is not indicated. Only controllers compliant with NVM Express Base Specification revision 1.3 and earlier shall be allowed to return this value.																					
01b	Reserved																					
10b	The Flush command does not support the NSID field set to FFFFFFFFh. The controller shall fail a Flush command with the NSID set to FFFFFFFFh with a status code of Invalid Namespace or Format.																					
11b	The Flush command supports the NSID field set to FFFFFFFFh.																					
00	<p><b>Volatile Write Cache Present (VWCP):</b> If this bit is set to '1', then a volatile write cache is present in the controller. If this bit is cleared to '0', then a volatile write cache is not present in the controller.</p> <p>If a volatile write cache is present, then the host controls whether the volatile write cache is enabled with a Set Features command specifying the Volatile Write Cache feature identifier (refer to section 5.1.25.1.4). The Flush command (refer to section 7.2) is used to request that the contents of a volatile write cache be made non-volatile.</p>																					
527:526	M	R	R	<p><b>Atomic Write Unit Normal (AWUN):</b> This field is specific to namespaces that are associated with command sets that specify logical blocks (i.e., Command Set Identifier 0h or 2h), and shall be cleared to 0h for namespaces that are not associated with command sets that specify logical blocks. Refer to the applicable I/O Command Set specification (e.g., the Atomic Operation section of the NVM Command Set Specification).</p>																		
529:528	M	M	R	<p><b>Atomic Write Unit Power Fail (AWUPF):</b> This field is specific to namespaces that are associated with command sets that specify logical blocks (i.e., Command Set Identifier 0h or 2h), and shall be cleared to 0h for namespaces that are not associated with command sets that specify logical blocks. Refer to the applicable I/O Command Set specification (e.g., the Atomic Operation section of the NVM Command Set Specification).</p>																		
530	M	M	R	<p><b>I/O Command Set Vendor Specific Command Configuration (ICSVSCC):</b> This field indicates the configuration settings for vendor specific I/O command handling. Refer to section 8.1.26.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td> <p><b>Same NVM Vendor Specific Command Format (SNVSCF):</b> If this bit is set to '1', then all vendor specific I/O commands use the format defined in Figure 93. If this bit is cleared to '0', then the format of all vendor specific I/O commands is vendor specific.</p> </td> </tr> </tbody> </table>	Bits	Description	7:1	Reserved	0	<p><b>Same NVM Vendor Specific Command Format (SNVSCF):</b> If this bit is set to '1', then all vendor specific I/O commands use the format defined in Figure 93. If this bit is cleared to '0', then the format of all vendor specific I/O commands is vendor specific.</p>												
Bits	Description																					
7:1	Reserved																					
0	<p><b>Same NVM Vendor Specific Command Format (SNVSCF):</b> If this bit is set to '1', then all vendor specific I/O commands use the format defined in Figure 93. If this bit is cleared to '0', then the format of all vendor specific I/O commands is vendor specific.</p>																					

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description														
531	M	M	R	<p><b>Namespace Write Protection Capabilities (NWPC):</b> This field indicates the optional namespace write protection capabilities supported by the controller. Refer to section 8.1.16.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:3</td> <td>Reserved</td> </tr> <tr> <td>2</td> <td> <p><b>Permanent Write Protect Support (PWPS):</b> If this bit is set to '1', then the controller supports the Permanent Write Protect state. If this bit is cleared to '0', then the controller does not support the Permanent Write Protect state. If this bit is set to '1', then the controller shall support the Write Protection Control field (refer to section 8.1.16.1). If the NWPWPS bit is cleared to '0', then this bit shall be cleared to '0'.</p> </td> </tr> <tr> <td>1</td> <td> <p><b>Write Protect Until Power Cycle Support (WPUPCS):</b> If this bit is set to '1', then the controller supports the Write Protect Until Power Cycle state. If this bit is cleared to '0', then the controller does not support Write Protect Until Power Cycle state. If this bit is set to '1', then the controller shall support the Write Protection Control field (refer to section 8.1.16.1). If the NWPWPS bit is cleared to '0', then this bit shall be cleared to '0'.</p> </td> </tr> <tr> <td>0</td> <td> <p><b>No Write Protect and Write Protect Support (NWPWPS) :</b> If this bit is set to '1', then the controller shall support the No Write Protect and Write Protect namespace write protection states and may support the Write Protect Until Power Cycle state and Permanent Write Protect namespace write protection states (refer to section 8.1.16). If this bit is cleared to '0', then the controller does not support Namespace Write Protection.</p> </td> </tr> </tbody> </table>	Bits	Description	7:3	Reserved	2	<p><b>Permanent Write Protect Support (PWPS):</b> If this bit is set to '1', then the controller supports the Permanent Write Protect state. If this bit is cleared to '0', then the controller does not support the Permanent Write Protect state. If this bit is set to '1', then the controller shall support the Write Protection Control field (refer to section 8.1.16.1). If the NWPWPS bit is cleared to '0', then this bit shall be cleared to '0'.</p>	1	<p><b>Write Protect Until Power Cycle Support (WPUPCS):</b> If this bit is set to '1', then the controller supports the Write Protect Until Power Cycle state. If this bit is cleared to '0', then the controller does not support Write Protect Until Power Cycle state. If this bit is set to '1', then the controller shall support the Write Protection Control field (refer to section 8.1.16.1). If the NWPWPS bit is cleared to '0', then this bit shall be cleared to '0'.</p>	0	<p><b>No Write Protect and Write Protect Support (NWPWPS) :</b> If this bit is set to '1', then the controller shall support the No Write Protect and Write Protect namespace write protection states and may support the Write Protect Until Power Cycle state and Permanent Write Protect namespace write protection states (refer to section 8.1.16). If this bit is cleared to '0', then the controller does not support Namespace Write Protection.</p>				
Bits	Description																	
7:3	Reserved																	
2	<p><b>Permanent Write Protect Support (PWPS):</b> If this bit is set to '1', then the controller supports the Permanent Write Protect state. If this bit is cleared to '0', then the controller does not support the Permanent Write Protect state. If this bit is set to '1', then the controller shall support the Write Protection Control field (refer to section 8.1.16.1). If the NWPWPS bit is cleared to '0', then this bit shall be cleared to '0'.</p>																	
1	<p><b>Write Protect Until Power Cycle Support (WPUPCS):</b> If this bit is set to '1', then the controller supports the Write Protect Until Power Cycle state. If this bit is cleared to '0', then the controller does not support Write Protect Until Power Cycle state. If this bit is set to '1', then the controller shall support the Write Protection Control field (refer to section 8.1.16.1). If the NWPWPS bit is cleared to '0', then this bit shall be cleared to '0'.</p>																	
0	<p><b>No Write Protect and Write Protect Support (NWPWPS) :</b> If this bit is set to '1', then the controller shall support the No Write Protect and Write Protect namespace write protection states and may support the Write Protect Until Power Cycle state and Permanent Write Protect namespace write protection states (refer to section 8.1.16). If this bit is cleared to '0', then the controller does not support Namespace Write Protection.</p>																	
533:532	O	R	R	<p><b>Atomic Compare &amp; Write Unit (ACWU):</b> This field is specific to namespaces that are associated with command sets that specify logical blocks (i.e., Command Set Identifier 0h or 2h), and shall be cleared to 0h for namespaces that are not associated with command sets that specify logical blocks. Refer to the applicable I/O Command Set specification (e.g., the Atomic Operation section of the NVM Command Set Specification).</p>														
535:534	M	R	R	<p><b>Copy Descriptor Formats Supported (CDFS):</b></p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:5</td> <td>Reserved</td> </tr> <tr> <td>4</td> <td> <p><b>Copy Descriptor Format 4 Support (CDF4S):</b> If this bit is set to '1', then the controller supports Copy Descriptor Format 4h. If this bit is cleared to '0', then the controller does not support Copy Descriptor Format 4h.</p> </td> </tr> <tr> <td>3</td> <td> <p><b>Copy Descriptor Format 3 Support (CDF3S):</b> If this bit is set to '1', then the controller supports Copy Descriptor Format 3h. If this bit is cleared to '0', then the controller does not support Copy Descriptor Format 3h.</p> </td> </tr> <tr> <td>2</td> <td> <p><b>Copy Descriptor Format 2 Support (CDF2S):</b> If this bit is set to '1', then the controller supports Copy Descriptor Format 2h. If this bit is cleared to '0', then the controller does not support Copy Descriptor Format 2h.</p> </td> </tr> <tr> <td>1</td> <td> <p><b>Copy Descriptor Format 1 Support (CDF1S):</b> If this bit is set to '1', then the controller supports Copy Descriptor Format 1h. If this bit is cleared to '0', then the controller does not support Copy Descriptor Format 1h.</p> </td> </tr> <tr> <td>0</td> <td> <p><b>Copy Descriptor Format 0 Support (CDF0S):</b> If this bit is set to '1', then the controller supports Copy Descriptor Format 0h. If this bit is cleared to '0', then the controller does not support Copy Descriptor Format 0h.</p> </td> </tr> </tbody> </table>	Bits	Description	15:5	Reserved	4	<p><b>Copy Descriptor Format 4 Support (CDF4S):</b> If this bit is set to '1', then the controller supports Copy Descriptor Format 4h. If this bit is cleared to '0', then the controller does not support Copy Descriptor Format 4h.</p>	3	<p><b>Copy Descriptor Format 3 Support (CDF3S):</b> If this bit is set to '1', then the controller supports Copy Descriptor Format 3h. If this bit is cleared to '0', then the controller does not support Copy Descriptor Format 3h.</p>	2	<p><b>Copy Descriptor Format 2 Support (CDF2S):</b> If this bit is set to '1', then the controller supports Copy Descriptor Format 2h. If this bit is cleared to '0', then the controller does not support Copy Descriptor Format 2h.</p>	1	<p><b>Copy Descriptor Format 1 Support (CDF1S):</b> If this bit is set to '1', then the controller supports Copy Descriptor Format 1h. If this bit is cleared to '0', then the controller does not support Copy Descriptor Format 1h.</p>	0	<p><b>Copy Descriptor Format 0 Support (CDF0S):</b> If this bit is set to '1', then the controller supports Copy Descriptor Format 0h. If this bit is cleared to '0', then the controller does not support Copy Descriptor Format 0h.</p>
Bits	Description																	
15:5	Reserved																	
4	<p><b>Copy Descriptor Format 4 Support (CDF4S):</b> If this bit is set to '1', then the controller supports Copy Descriptor Format 4h. If this bit is cleared to '0', then the controller does not support Copy Descriptor Format 4h.</p>																	
3	<p><b>Copy Descriptor Format 3 Support (CDF3S):</b> If this bit is set to '1', then the controller supports Copy Descriptor Format 3h. If this bit is cleared to '0', then the controller does not support Copy Descriptor Format 3h.</p>																	
2	<p><b>Copy Descriptor Format 2 Support (CDF2S):</b> If this bit is set to '1', then the controller supports Copy Descriptor Format 2h. If this bit is cleared to '0', then the controller does not support Copy Descriptor Format 2h.</p>																	
1	<p><b>Copy Descriptor Format 1 Support (CDF1S):</b> If this bit is set to '1', then the controller supports Copy Descriptor Format 1h. If this bit is cleared to '0', then the controller does not support Copy Descriptor Format 1h.</p>																	
0	<p><b>Copy Descriptor Format 0 Support (CDF0S):</b> If this bit is set to '1', then the controller supports Copy Descriptor Format 0h. If this bit is cleared to '0', then the controller does not support Copy Descriptor Format 0h.</p>																	

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description																						
539:536	O	O	M	<p><b>SGL Support (SGLS):</b> This field indicates if SGLs are supported and the particular SGL types supported. Refer to section 4.3.2.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>31:22</td> <td>Reserved</td> </tr> <tr> <td>21</td> <td><b>Transport SGL Data Block Descriptor Support (TSDBDS):</b> If this bit is set to '1', then the controller supports the Transport SGL Data Block descriptor. If this bit is cleared to '0', then the controller does not support the Transport SGL Data Block descriptor.</td> </tr> <tr> <td>20</td> <td><b>SGL Address Offset Supported (SAOS):</b> If this bit is set to '1', then the controller supports the Address field in SGL Data Block, SGL Segment, and SGL Last Segment descriptor types specifying an offset. If this bit is cleared to '0', then the Address field specifying an offset is not supported.</td> </tr> <tr> <td>19</td> <td><b>MPTR SGL Descriptor Support (MSDS):</b> If this bit is set to '1', then use of a Metadata Pointer (MPTR) that contains an address of an SGL segment containing exactly one SGL Descriptor that is qword aligned is supported. If this bit is cleared to '0', then use of a MPTR containing an SGL Descriptor is not supported.</td> </tr> <tr> <td>18</td> <td><b>Length Larger than Data Transfer Support (LLDTS):</b> If this bit is set to '1', then the controller supports commands that contain a data or metadata SGL of a length larger than the amount of data to be transferred. If this bit is cleared to '0', then the SGL length shall be equal to the amount of data to be transferred.</td> </tr> <tr> <td>17</td> <td><b>Metadata Buffer Alignment (MBA):</b> This bit indicates metadata buffer alignment requirements when CDW0.PSDT is set to 01b (refer to Figure 91). If set to '1', then use of a byte aligned contiguous physical buffer of metadata (the Metadata Pointer field in Figure 92) is supported. If cleared to '0', then use of a byte aligned contiguous physical buffer of metadata is not supported.</td> </tr> <tr> <td>16</td> <td><b>SGL Bit Bucket Descriptor Supported (SBBDS):</b> If this bit is set to '1', then the SGL Bit Bucket descriptor is supported. If this bit is cleared to '0', then the SGL Bit Bucket descriptor is not supported.</td> </tr> <tr> <td>15:08</td> <td><b>SGL Descriptor Threshold (SDT):</b> This field indicates the recommended maximum number of SGL descriptors in a command (refer to section 4.3.2). If this field is cleared to 0h, then no recommended maximum number of SGL descriptors is reported.</td> </tr> <tr> <td>07:03</td> <td>Reserved</td> </tr> <tr> <td>02</td> <td><b>Keyed SGL Data Block Descriptor Support (KSDBDS):</b> If this bit is set to '1', then the controller supports the Keyed SGL Data Block descriptor. If this bit is cleared to '0', then the controller does not support the Keyed SGL Data Block descriptor.</td> </tr> </tbody> </table>	Bits	Description	31:22	Reserved	21	<b>Transport SGL Data Block Descriptor Support (TSDBDS):</b> If this bit is set to '1', then the controller supports the Transport SGL Data Block descriptor. If this bit is cleared to '0', then the controller does not support the Transport SGL Data Block descriptor.	20	<b>SGL Address Offset Supported (SAOS):</b> If this bit is set to '1', then the controller supports the Address field in SGL Data Block, SGL Segment, and SGL Last Segment descriptor types specifying an offset. If this bit is cleared to '0', then the Address field specifying an offset is not supported.	19	<b>MPTR SGL Descriptor Support (MSDS):</b> If this bit is set to '1', then use of a Metadata Pointer (MPTR) that contains an address of an SGL segment containing exactly one SGL Descriptor that is qword aligned is supported. If this bit is cleared to '0', then use of a MPTR containing an SGL Descriptor is not supported.	18	<b>Length Larger than Data Transfer Support (LLDTS):</b> If this bit is set to '1', then the controller supports commands that contain a data or metadata SGL of a length larger than the amount of data to be transferred. If this bit is cleared to '0', then the SGL length shall be equal to the amount of data to be transferred.	17	<b>Metadata Buffer Alignment (MBA):</b> This bit indicates metadata buffer alignment requirements when CDW0.PSDT is set to 01b (refer to Figure 91). If set to '1', then use of a byte aligned contiguous physical buffer of metadata (the Metadata Pointer field in Figure 92) is supported. If cleared to '0', then use of a byte aligned contiguous physical buffer of metadata is not supported.	16	<b>SGL Bit Bucket Descriptor Supported (SBBDS):</b> If this bit is set to '1', then the SGL Bit Bucket descriptor is supported. If this bit is cleared to '0', then the SGL Bit Bucket descriptor is not supported.	15:08	<b>SGL Descriptor Threshold (SDT):</b> This field indicates the recommended maximum number of SGL descriptors in a command (refer to section 4.3.2). If this field is cleared to 0h, then no recommended maximum number of SGL descriptors is reported.	07:03	Reserved	02	<b>Keyed SGL Data Block Descriptor Support (KSDBDS):</b> If this bit is set to '1', then the controller supports the Keyed SGL Data Block descriptor. If this bit is cleared to '0', then the controller does not support the Keyed SGL Data Block descriptor.
				Bits	Description																					
				31:22	Reserved																					
				21	<b>Transport SGL Data Block Descriptor Support (TSDBDS):</b> If this bit is set to '1', then the controller supports the Transport SGL Data Block descriptor. If this bit is cleared to '0', then the controller does not support the Transport SGL Data Block descriptor.																					
				20	<b>SGL Address Offset Supported (SAOS):</b> If this bit is set to '1', then the controller supports the Address field in SGL Data Block, SGL Segment, and SGL Last Segment descriptor types specifying an offset. If this bit is cleared to '0', then the Address field specifying an offset is not supported.																					
				19	<b>MPTR SGL Descriptor Support (MSDS):</b> If this bit is set to '1', then use of a Metadata Pointer (MPTR) that contains an address of an SGL segment containing exactly one SGL Descriptor that is qword aligned is supported. If this bit is cleared to '0', then use of a MPTR containing an SGL Descriptor is not supported.																					
				18	<b>Length Larger than Data Transfer Support (LLDTS):</b> If this bit is set to '1', then the controller supports commands that contain a data or metadata SGL of a length larger than the amount of data to be transferred. If this bit is cleared to '0', then the SGL length shall be equal to the amount of data to be transferred.																					
				17	<b>Metadata Buffer Alignment (MBA):</b> This bit indicates metadata buffer alignment requirements when CDW0.PSDT is set to 01b (refer to Figure 91). If set to '1', then use of a byte aligned contiguous physical buffer of metadata (the Metadata Pointer field in Figure 92) is supported. If cleared to '0', then use of a byte aligned contiguous physical buffer of metadata is not supported.																					
				16	<b>SGL Bit Bucket Descriptor Supported (SBBDS):</b> If this bit is set to '1', then the SGL Bit Bucket descriptor is supported. If this bit is cleared to '0', then the SGL Bit Bucket descriptor is not supported.																					
				15:08	<b>SGL Descriptor Threshold (SDT):</b> This field indicates the recommended maximum number of SGL descriptors in a command (refer to section 4.3.2). If this field is cleared to 0h, then no recommended maximum number of SGL descriptors is reported.																					
				07:03	Reserved																					
				02	<b>Keyed SGL Data Block Descriptor Support (KSDBDS):</b> If this bit is set to '1', then the controller supports the Keyed SGL Data Block descriptor. If this bit is cleared to '0', then the controller does not support the Keyed SGL Data Block descriptor.																					
				01:00	<p><b>SGL Support (SGLS):</b> This field is used to determine the SGL support. Valid values are shown in the table below.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>SGLs are not supported.</td> </tr> <tr> <td>01b</td> <td>SGLs are supported. There is no alignment nor granularity requirement for Data Blocks.</td> </tr> <tr> <td>10b</td> <td>SGLs are supported. There is a dword alignment and granularity requirement for Data Blocks (refer to section 4.3.2).</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	SGLs are not supported.	01b	SGLs are supported. There is no alignment nor granularity requirement for Data Blocks.	10b	SGLs are supported. There is a dword alignment and granularity requirement for Data Blocks (refer to section 4.3.2).	11b	Reserved											
Value	Definition																									
00b	SGLs are not supported.																									
01b	SGLs are supported. There is no alignment nor granularity requirement for Data Blocks.																									
10b	SGLs are supported. There is a dword alignment and granularity requirement for Data Blocks (refer to section 4.3.2).																									
11b	Reserved																									

**Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent**

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description
543:540	O	O	R	<b>Maximum Number of Allowed Namespaces (MNAN):</b> This field indicates the maximum number of namespaces supported by the NVM subsystem. If this field is cleared to 0h, then the maximum number of namespaces supported by the NVM subsystem is less than or equal to the value in the NN field. If the controller supports Asymmetric Namespace Access Reporting, then this field shall be set to a non-zero value that is less than or equal to the NN value.
559:544	O	R	R	<b>Maximum Domain Namespace Attachments (MAXDNA):</b> Indicates the maximum of the sum of the number of namespaces attached to each I/O controller in the Domain. If this field is cleared to 0h, then no maximum is specified.  The value of this field shall be the same value for all I/O controllers in the Domain.
563:560	O	R	R	<b>Maximum I/O Controller Namespace Attachments (MAXCNA):</b> Indicates the maximum number of namespaces that are allowed to be attached to this I/O controller. If this field is cleared to 0h, then no maximum is specified.  The value of this field shall be less than or equal to the number of namespaces supported by the NVM subsystem (refer to the MNAN field).
567:564	O	R	R	<b>Optimal Aggregated Queue Depth (OAQD):</b> Indicates the recommended maximum total number of outstanding I/O commands across all I/O queues on the controller for optimal operation. The host may use this value to limit the number of commands outstanding at one time across all I/O queues on the controller.  If this field is cleared to 0h, then the Optimal Aggregated Queue Depth is not reported.
568	O	R	R	<b>Recommended Host-Initiated Refresh Interval (RHIRI):</b> If the Host-Initiated Refresh capability is supported (i.e., the HIRS bit in the DSTO field is set to '1'), then this field indicates the recommended time interval in days from last power down to the time at which the host should initiate the Host-Initiated Refresh operation.  If this field is cleared to 0h, then this field is not reported.  If the HIRS bit in the DSTO field is cleared to '0', then This field shall be cleared to 0h.
569	O	R	R	<b>Host-Initiated Refresh Time (HIRT):</b> If the Host-Initiated Refresh capability is supported (i.e., the HIRS bit in the DSTO field is set to '1'), then this field indicates the nominal amount of time in minutes that the controller takes to complete the Host-Initiated Refresh operation.  If this field is cleared to 0h, then this field is not reported.  If the HIRS bit in the DSTO field is cleared to '0', then This field shall be cleared to 0h.
571:570	O	O	P	<b>Controller Maximum Memory Range Tracking Descriptors (CMMRTD):</b> This field indicates the maximum number of Memory Range Tracking Descriptors (refer to Figure 469) the controller supports (refer to section 5.1.27.1.2).  If the THMCS bit is cleared to '0', then this field shall be cleared to 0h and the host should ignore this field.
573:572	O	O	P	<b>NVM Subsystem Maximum Memory Range Tracking Descriptors (NMMRTD):</b> This field indicates the maximum number of Memory Range Tracking Descriptors (refer to Figure 469) the NVM subsystem supports (refer to section 5.1.27.1.2).  If the THMCS bit is cleared to '0', then this field shall be cleared to 0h and the host should ignore this field.
574	O	O	P	<b>Minimum Memory Range Tracking Granularity (MINMRTG):</b> This field indicates the minimum value supported in the Requested Memory Range Tracking Granularity (RMRTG) field (refer to Figure 469) of the Track Memory Ranges data structure.  If the THMCS bit is cleared to '0', then this field shall be cleared to 0h and the host should ignore this field.

Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description										
575	O	O	P	<b>Maximum Memory Range Tracking Granularity (MAXMRTG):</b> This field indicates the maximum value supported in the Requested Memory Range Tracking Granularity (RMRTG) field (refer to Figure 469) of the Track Memory Ranges data structure. If the THMCS bit is cleared to '0', then this field shall be cleared to 0h and the host should ignore this field.										
576	O	O	P	<p><b>Tracking Attributes (TRATTR):</b> This field indicates supported attributes for the Track Send command and Track Receive command.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:3</td> <td>Reserved</td> </tr> <tr> <td>2</td> <td><b>Memory Range Tracking Length Limit (MRTL):</b> If this bit is set to '1', then the controller only supports the length as specified by the Requested Memory Range Tracking Granularity field and the Length field in a Track Memory Changes data structure (refer to Figure 469) indicating a value that is a power of 2. If this bit is cleared to '0', then the length indicated by the Requested Memory Range Tracking Granularity field and the Length field in a Track Memory Changes data structure is not required to be a value that is a power of 2.</td> </tr> <tr> <td>1</td> <td><b>Track User Data Changes Support (TUDCS):</b> If this bit is set to '1', then the controller supports tracking user data changes as defined by section 5.1.26.1.1. If this bit is cleared to '0', then the controller does not support tracking user data changes as defined by section 5.1.26.1.1.</td> </tr> <tr> <td>0</td> <td><b>Track Host Memory Changes Support (THMCS):</b> If this bit is set to '1', then the controller supports tracked memory changes as defined by section 5.1.26.1.1 and section 5.1.27.1.2. If this bit is cleared to '0', then the controller does not support the tracked memory changes as defined by section 5.1.26.1.1 and section 5.1.27.1.2.</td> </tr> </tbody> </table>	Bits	Description	7:3	Reserved	2	<b>Memory Range Tracking Length Limit (MRTL):</b> If this bit is set to '1', then the controller only supports the length as specified by the Requested Memory Range Tracking Granularity field and the Length field in a Track Memory Changes data structure (refer to Figure 469) indicating a value that is a power of 2. If this bit is cleared to '0', then the length indicated by the Requested Memory Range Tracking Granularity field and the Length field in a Track Memory Changes data structure is not required to be a value that is a power of 2.	1	<b>Track User Data Changes Support (TUDCS):</b> If this bit is set to '1', then the controller supports tracking user data changes as defined by section 5.1.26.1.1. If this bit is cleared to '0', then the controller does not support tracking user data changes as defined by section 5.1.26.1.1.	0	<b>Track Host Memory Changes Support (THMCS):</b> If this bit is set to '1', then the controller supports tracked memory changes as defined by section 5.1.26.1.1 and section 5.1.27.1.2. If this bit is cleared to '0', then the controller does not support the tracked memory changes as defined by section 5.1.26.1.1 and section 5.1.27.1.2.
Bits	Description													
7:3	Reserved													
2	<b>Memory Range Tracking Length Limit (MRTL):</b> If this bit is set to '1', then the controller only supports the length as specified by the Requested Memory Range Tracking Granularity field and the Length field in a Track Memory Changes data structure (refer to Figure 469) indicating a value that is a power of 2. If this bit is cleared to '0', then the length indicated by the Requested Memory Range Tracking Granularity field and the Length field in a Track Memory Changes data structure is not required to be a value that is a power of 2.													
1	<b>Track User Data Changes Support (TUDCS):</b> If this bit is set to '1', then the controller supports tracking user data changes as defined by section 5.1.26.1.1. If this bit is cleared to '0', then the controller does not support tracking user data changes as defined by section 5.1.26.1.1.													
0	<b>Track Host Memory Changes Support (THMCS):</b> If this bit is set to '1', then the controller supports tracked memory changes as defined by section 5.1.26.1.1 and section 5.1.27.1.2. If this bit is cleared to '0', then the controller does not support the tracked memory changes as defined by section 5.1.26.1.1 and section 5.1.27.1.2.													
577				Reserved										
579:578	O	O	P	<p><b>Maximum Controller User Data Migration Queues (MCUDMQ):</b> This field indicates the maximum number of User Data Migration Queues supported by the controller (i.e., allowed to be created in this controller).</p> <p>If the TUDCS bit is cleared to '0', then this field shall be cleared to 0h and the host should ignore this field.</p>										
581:580	O	O	P	<p><b>Maximum NVM Subsystem User Data Migration Queues (MNSUDMQ):</b> This field indicates the maximum number of User Data Migration Queues supported by the NVM subsystem (i.e., allowed to be created in this NVM subsystem).</p> <p>If the TUDCS bit is cleared to '0', then this field shall be cleared to 0h and the host should ignore this field.</p>										
583:582	O	O	P	<b>Maximum CDQ Memory Ranges (MCMR):</b> This field indicates the maximum number of memory ranges allowed to be specified by the PRP1 field of a Controller Data Queue command. A value of 0h indicates that a maximum is not reported.										
585:584	O	O	P	<p><b>NVM Subsystem Maximum CDQ Memory Ranges (NMCMR):</b> This field indicates the maximum number of memory ranges for all Controller Data Queues in the NVM subsystem. A value of 0h indicates that a maximum is not reported.</p> <p>The value of this field shall be greater than or equal to the value of the MCMR field.</p>										
587:586	O	O	P	<b>Maximum Controller Data Queue PRP Count (MCDQPC):</b> This field indicates the maximum number of PRPs allowed to be specified in the PRP list in the Controller Data Queue command. A value of 0h indicates that a maximum is not reported.										
767:588				Reserved										



**Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent**

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description								
1023:768	M	M	R	<p><b>NVM Subsystem NVMe Qualified Name (SUBNQN):</b> This field specifies the NVM Subsystem NVMe Qualified Name as a UTF-8 null-terminated string. Refer to section 4.7 for the definition of NVMe Qualified Name.</p> <p>Support for this field is mandatory if the controller supports revision 1.2.1 or later as indicated in the Version property (refer to section 3.1.4.2).</p> <p>For a Discovery controller, if the Discovery subsystem containing the Discovery controller has a unique Discovery Service NQN, then this field shall be set to that unique Discovery Service NQN. If the Discovery subsystem containing the Discovery controller does not have a unique Discovery Service NQN, then this field shall be set to the well-known Discovery Service NQN (i.e., nqn.2014-08.org.nvmeexpress.discovery).</p>								
1791:1024				Reserved								
<b>Fabric Specific</b>												
1795:1792	M <sup>2</sup>	R	R	<b>I/O Queue Command Capsule Supported Size (IOCCSZ):</b> 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 <sup>2</sup>	R	R	<b>I/O Queue Response Capsule Supported Size (IORCSZ):</b> 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.								
1801:1800	M <sup>2</sup>	R	R	<p><b>In Capsule Data Offset (ICDOFF):</b> 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>								
1802	M <sup>2</sup>	M <sup>2</sup>	R	<p><b>Fabrics Controller Attributes (FCATT):</b> This field indicates attributes of the controller that are specific to NVMe over Fabrics.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:2</td> <td>Reserved</td> </tr> <tr> <td>1</td> <td><b>Non-Zero NVM Set ID Support (NZNSETIDS):</b> If this bit is cleared to '0', then the NVM controller does not support a non-zero value in the NVMSETID field in the Connect command (refer to section 6.3). If this bit is set to '1', then the NVM controller does support a non-zero value in the NVMSETID field in the Connect command.</td> </tr> <tr> <td>0</td> <td><b>Dynamic Controller Model Support (DCMS):</b> If this bit is cleared to '0', then the NVM subsystem uses a dynamic controller model. If this bit is set to '1', then the NVM subsystem uses a static controller model.</td> </tr> </tbody> </table>	Bits	Description	7:2	Reserved	1	<b>Non-Zero NVM Set ID Support (NZNSETIDS):</b> If this bit is cleared to '0', then the NVM controller does not support a non-zero value in the NVMSETID field in the Connect command (refer to section 6.3). If this bit is set to '1', then the NVM controller does support a non-zero value in the NVMSETID field in the Connect command.	0	<b>Dynamic Controller Model Support (DCMS):</b> If this bit is cleared to '0', then the NVM subsystem uses a dynamic controller model. If this bit is set to '1', then the NVM subsystem uses a static controller model.
Bits	Description											
7:2	Reserved											
1	<b>Non-Zero NVM Set ID Support (NZNSETIDS):</b> If this bit is cleared to '0', then the NVM controller does not support a non-zero value in the NVMSETID field in the Connect command (refer to section 6.3). If this bit is set to '1', then the NVM controller does support a non-zero value in the NVMSETID field in the Connect command.											
0	<b>Dynamic Controller Model Support (DCMS):</b> If this bit is cleared to '0', then the NVM subsystem uses a dynamic controller model. If this bit is set to '1', then the NVM subsystem uses a static controller model.											
1803	M <sup>2</sup>	M <sup>2</sup>	R	<b>Maximum SGL Data Block Descriptors (MSDBD):</b> This field indicates the maximum number of SGL Data Block or Keyed SGL Data Block descriptors that a host is allowed to place in a capsule. A value of 0h indicates no limit.								
1805:1804	M <sup>2</sup>	M <sup>2</sup>	R	<p><b>Optional Fabrics Commands Support (OFCS):</b> Indicate whether the controller supports optional Fabrics commands.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td><b>Disconnect Command Support (DCS):</b> If this bit is cleared to '0', then the controller does not support the Disconnect command. If this bit is set to '1', then the controller supports the Disconnect command and deletion of individual I/O Queues.</td> </tr> </tbody> </table>	Bits	Description	15:1	Reserved	0	<b>Disconnect Command Support (DCS):</b> If this bit is cleared to '0', then the controller does not support the Disconnect command. If this bit is set to '1', then the controller supports the Disconnect command and deletion of individual I/O Queues.		
Bits	Description											
15:1	Reserved											
0	<b>Disconnect Command Support (DCS):</b> If this bit is cleared to '0', then the controller does not support the Disconnect command. If this bit is set to '1', then the controller supports the Disconnect command and deletion of individual I/O Queues.											

**Figure 312: Identify – Identify Controller Data Structure, I/O Command Set Independent**

Bytes	I/O <sup>1</sup>	Admin <sup>1</sup>	Disc <sup>1</sup>	Description										
1806	R	R	O	<b>Discovery Controller Type (DCTYPE):</b> This field indicates what type of Discovery controller the controller is. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Discovery controller type is not reported</td> </tr> <tr> <td>1h</td> <td>DDC</td> </tr> <tr> <td>2h</td> <td>CDC</td> </tr> <tr> <td>3h to FFh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	Discovery controller type is not reported	1h	DDC	2h	CDC	3h to FFh	Reserved
Value	Definition													
0h	Discovery controller type is not reported													
1h	DDC													
2h	CDC													
3h to FFh	Reserved													
2047:1807				Reserved										
<b>Power State Descriptors</b>														
2079:2048	M	M	R	<b>Power State 0 Descriptor (PSD0):</b> This field indicates the characteristics of power state 0. The format of this field is defined in Figure 313.										
2111:2080	O	O	R	<b>Power State 1 Descriptor (PSD1):</b> This field indicates the characteristics of power state 1. The format of this field is defined in Figure 313.										
2143:2112	O	O	R	<b>Power State 2 Descriptor (PSD2):</b> This field indicates the characteristics of power state 2. The format of this field is defined in Figure 313.										
...														
3071:3040	O	O	R	<b>Power State 31 Descriptor (PSD31):</b> This field indicates the characteristics of power state 31. The format of this field is defined in Figure 313.										
<b>Vendor Specific</b>														
4095:3072	O	O	O	<b>Vendor Specific (VS)</b>										
Notes: <ol style="list-style-type: none"> <li>O/M/R definition: O = Optional, M = Mandatory, R = Reserved.</li> <li>Mandatory for message-based controllers. For memory-based controllers this field is reserved.</li> <li>Mandatory for Discovery controllers that support explicit persistent connections. For Discovery controllers that do not support explicit persistent connections this field is reserved.</li> <li>For Discovery controllers, the TBKAS bit is optional, and all other bits are reserved.</li> <li>If a Discovery controller returns a non-null value for either the Serial Number (SN) or Model Number (MN) fields, that Discovery controller shall return a non-null value for both of those fields.</li> </ol>														

Figure 313 defines the power state descriptor that describes the attributes of each power state. For more information on how the power state descriptor fields are used, refer to section 8.1.17 on power management.

**Figure 313: Identify – Power State Descriptor Data Structure**

Bits	Description
255:220	Reserved
219:216	<b>Emergency Power Fail Vault Time Scale (EPFVTS):</b> This field indicates the scale for the Emergency Power Fail Vault Time field as defined in Figure 314. If the EPFVT field is cleared to 0h, then this field: <ul style="list-style-type: none"> <li>shall be cleared to 0h; and</li> <li>is ignored by the host.</li> </ul>
215:212	<b>Forced Quiescence Vault Time Scale (FQVTS):</b> This field indicates the scale for the Forced Quiescence Vault Time field as defined in Figure 314. If the FQVT field is cleared to 0h, then this field: <ul style="list-style-type: none"> <li>shall be cleared to 0h; and</li> <li>is ignored by the host.</li> </ul>

**Figure 313: Identify – Power State Descriptor Data Structure**

Bits	Description										
211:208	<p><b>Emergency Power Fail Recovery Time Scale (EPFRTS):</b> This field indicates the scale for the Emergency Power Fail Recovery Time field as defined in Figure 314.</p> <p>If the EPFRT field is cleared to 0h, then this field:</p> <ul style="list-style-type: none"> <li>• shall be cleared to 0h; and</li> <li>• is ignored by the host.</li> </ul>										
207:200	<p><b>Emergency Power Fail Vault Time (EPFVT):</b> This field, with the EPFVTS field, indicates the worst-case time required for the controller to complete Emergency Power Fail Processing (refer to section 8.2.5.3) in the absence of failures. The time is equal to the value in this field multiplied by the scale indicated by the EPFVTS field.</p> <p>If Power Loss Signaling with Emergency Power Fail is not supported (i.e., the PLSEPF bit is cleared to '0' in the Power Loss Signaling Information field of the Identify Controller data structure (refer to Figure 312)), then this field shall be cleared to 0h.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Not reported</td> </tr> <tr> <td>1 to 99</td> <td>Time value</td> </tr> <tr> <td>100 to 255</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0	Not reported	1 to 99	Time value	100 to 255	Reserved		
Value	Definition										
0	Not reported										
1 to 99	Time value										
100 to 255	Reserved										
199:192	<p><b>Forced Quiescence Vault Time (FQVT):</b> This field, with the FQVTS field, indicates the worst-case time required for the controller to complete Forced Quiescence Processing (refer to section 8.2.5.2). The time is equal to the value in this field multiplied by the scale indicated in the FQVTS field.</p> <p>If Power Loss Signaling with Forced Quiescence is not supported (i.e., the PLSFQ bit is cleared to '0' in the Power Loss Signaling Information field of the Identify Controller data structure (refer to Figure 312)), then this field shall be cleared to 0h.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Not reported</td> </tr> <tr> <td>1 to 99</td> <td>Time value</td> </tr> <tr> <td>100 to 255</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0	Not reported	1 to 99	Time value	100 to 255	Reserved		
Value	Definition										
0	Not reported										
1 to 99	Time value										
100 to 255	Reserved										
191:184	<p><b>Emergency Power Fail Recovery Time (EPFRT):</b> This field, with the EPFRTS field, indicates the worst-case time required for the controller to complete the first initialization (as described in section 8.2.5.3) after an Emergency Power Fail process which completed before power loss. The time is equal to the value in this field multiplied by the scale indicated in the EPFRTS field.</p> <p>If Power Loss Signaling with Emergency Power Fail is not supported (i.e., the PLSEPF bit is cleared to '0' in the Power Loss Signaling Information field of the Identify Controller data structure (refer to Figure 312)), then this field shall be cleared to 0h.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Not reported</td> </tr> <tr> <td>1 to 99</td> <td>Time value</td> </tr> <tr> <td>100 to 255</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0	Not reported	1 to 99	Time value	100 to 255	Reserved		
Value	Definition										
0	Not reported										
1 to 99	Time value										
100 to 255	Reserved										
183:182	<p><b>Active Power Scale (APS):</b> This field indicates the scale for the Active Power field. If an Active Power Workload is reported for a power state, then the Active Power Scale shall also be reported for that power state.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Not reported for this power state</td> </tr> <tr> <td>01b</td> <td>0.0001 W</td> </tr> <tr> <td>10b</td> <td>0.01 W</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	Not reported for this power state	01b	0.0001 W	10b	0.01 W	11b	Reserved
Value	Definition										
00b	Not reported for this power state										
01b	0.0001 W										
10b	0.01 W										
11b	Reserved										
181:179	Reserved										
178:176	<p><b>Active Power Workload (APW):</b> This field indicates the workload used to calculate maximum power for this power state. Refer to section 8.1.17.3 for more details on each of the defined workloads. This field shall not be "No Workload" unless ACTP is 0h.</p>										

Figure 313: Identify – Power State Descriptor Data Structure

Bits	Description										
175:160	<b>Active Power (ACTP):</b> This field indicates the largest average power consumed by the NVM subsystem over a 10 second period in this power state with the workload indicated in the Active Power Workload field. The power in Watts is equal to the value in this field multiplied by the scale indicated in the Active Power Scale field. A value of 0h indicates Active Power is not reported.										
159:152	Reserved										
151:150	<p><b>Idle Power Scale (IPS):</b> This field indicates the scale for the Idle Power field.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Not reported for this power state</td> </tr> <tr> <td>01b</td> <td>0.0001 W</td> </tr> <tr> <td>10b</td> <td>0.01 W</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	Not reported for this power state	01b	0.0001 W	10b	0.01 W	11b	Reserved
Value	Definition										
00b	Not reported for this power state										
01b	0.0001 W										
10b	0.01 W										
11b	Reserved										
149:144	Reserved										
143:128	<p><b>Idle Power (IDL P):</b> This field indicates the typical power consumed by the NVM subsystem over 30 seconds in this power state when idle (e.g., there are no pending commands, property accesses, background processes, sanitize operation, nor device self-test operations). The measurement starts after the NVM subsystem has been idle for 10 seconds. The power in Watts is equal to the value in this field multiplied by the scale indicated in the Idle Power Scale field. A value of 0h indicates Idle Power is not reported. Refer to section 8.1.17.</p> <p>Note: This value may be used by hosts to manage power versus resume latency. Platform and form factor specifications may have additional power measurement and reporting requirements that are outside the scope of this specification.</p>										
127:125	Reserved										
124:120	<b>Relative Write Latency (RWL):</b> This field indicates the relative write latency associated with this power state. The value in this field shall be less than the number of supported power states (e.g., if the controller supports 16 power states, then valid values are 0 through 15). A lower value means lower write latency.										
119:117	Reserved										
116:112	<b>Relative Write Throughput (RWT):</b> This field indicates the relative write throughput associated with this power state. The value in this field shall be less than the number of supported power states (e.g., if the controller supports 16 power states, then valid values are 0 through 15). A lower value means higher write throughput.										
111:109	Reserved										
108:104	<b>Relative Read Latency (RRL):</b> This field indicates the relative read latency associated with this power state. The value in this field shall be less than the number of supported power states (e.g., if the controller supports 16 power states, then valid values are 0 through 15). A lower value means lower read latency.										
103:101	Reserved										
100:96	<b>Relative Read Throughput (RRT):</b> This field indicates the relative read throughput associated with this power state. The value in this field shall be less than the number of supported power states (e.g., if the controller supports 16 power states, then valid values are 0 through 15). A lower value means higher read throughput.										
95:64	<b>Exit Latency (EXLAT):</b> This field indicates the maximum exit latency in microseconds associated with exiting this power state. A value of 0h indicates Exit Latency is not reported.										
63:32	<b>Entry Latency (ENLAT):</b> This field indicates the maximum entry latency in microseconds associated with entering this power state. A value of 0h indicates Entry Latency is not reported.										
31:26	Reserved										
25	<b>Non-Operational State (NOPS):</b> This bit indicates whether the controller processes I/O commands in this power state. If this bit is cleared to '0', then the controller processes I/O commands in this power state. If this bit is set to '1', then the controller does not process I/O commands in this power state. Refer to section 8.1.17.1.										
24	<b>Max Power Scale (MXPS):</b> This bit indicates the scale for the Maximum Power field. If this bit is cleared to '0', then the scale of the Maximum Power field is in 0.01 Watts. If this bit is set to '1', then the scale of the Maximum Power field is in 0.0001 Watts.										
23:16	Reserved										

**Figure 313: Identify – Power State Descriptor Data Structure**

Bits	Description
15:00	<p><b>Maximum Power (MP):</b> This field indicates the sustained maximum power consumed by the NVM subsystem in this power state. The power in Watts is equal to the value in this field multiplied by the scale specified in the Max Power Scale bit. A value of 0h indicates Maximum Power is not reported. Refer to section 8.1.17.</p> <p>Note: This value is intended to provide an approximate guideline for hosts to manage power versus performance. Platform and form factor specifications may have additional power measurement and reporting requirements that are outside the scope of this specification.</p>

Figure 314 describes the time scales indicated by the EPFVTS field, the FQVTS field, and the EPFRTS field (refer to Figure 313).

**Figure 314: Time Scale Values**

Value	Definition
0h	1 microsecond
1h	10 microseconds
2h	100 microseconds
3h	1 millisecond
4h	10 milliseconds
5h	100 milliseconds
6h	1 second
7h	10 seconds
8h	100 seconds
9h	1,000 seconds
Ah	10,000 seconds
Bh	100,000 seconds
Ch	1,000,000 seconds
Dh	Reserved
Eh	Reserved
Fh	Reserved

#### 5.1.13.2.2 Active Namespace ID list (CNS 02h)

A list of 1,024 namespace IDs is returned to the host containing active NSIDs in increasing order that are greater than the value specified in the Namespace Identifier (NSID) field of the command. The controller shall abort the command with a status code of Invalid Namespace or Format if the NSID field is set to FFFFFFFEh or FFFFFFFFh. The NSID field may be cleared to 0h to retrieve a Namespace List including the namespace starting with NSID of 1h. The data structure returned is a Namespace List (refer to section 4.6.2).

#### 5.1.13.2.3 Namespace Identification Descriptor list (CNS 03h)

A list of Namespace Identification Descriptor structures (refer to Figure 315) is returned to the host for the specified namespace if the value in the Namespace Identifier (NSID) field is an active NSID. The controller shall abort the command with a status code of Invalid Namespace or Format if the NSID field is set to FFFFFFFEh or is set to FFFFFFFFh. Namespace Identification Descriptor structures consist of one or more Namespace Identifiers (NID) of various types as indicated by the Namespace Identifier Type (NIDT) field in each descriptor. Each NID is assigned to a namespace at namespace creation and remains fixed throughout the life of that namespace. If the NSID field does not specify an active NSID, then refer to section 3.2.1.5 for the status code to return. Textual representations related to Boot of NIDs may be found in the NVM Express Boot Specification.

The contents of the Namespace Identification Descriptor list is preserved across namespace and controller operations (e.g., Controller Level Reset, namespace format, etc.).

The controller may return any number of variable length Namespace Identification Descriptor structures that fit into the 4,096 byte Identify payload. All remaining bytes after the Namespace Identification Descriptor structures should be cleared to 0h, and the host shall interpret a Namespace Identifier Descriptor Length (NIDL) value of 0h as the end of the list. The host should ignore any Namespace Identification Descriptor with a Namespace Identifier Type not supported by the host.

A controller shall not return multiple Namespace Identification Descriptors with the same Namespace Identifier Type (NIDT). A controller shall return:

- at least one Namespace Identification Descriptor identifying the namespace (i.e., NIDT field set to 1h, 2h, or 3h); and
- A Namespace Identification Descriptor identifying the I/O Command Set (i.e., NIDT field set to 4h) if the CAP.CSS.IOCSS bit is set to '1'.

**Figure 315: Identify – Namespace Identification Descriptor**

Bytes	Description																					
00	<b>Namespace Identifier Type (NIDT):</b> This field indicates the data type contained in the Namespace Identifier field and the length of that type as defined in the following table.																					
	<table border="1"> <thead> <tr> <th>Value</th> <th>Length (NIDL)</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>Reserved</td> </tr> <tr> <td>1h</td> <td>8h</td> <td><b>IEEE Extended Unique Identifier (IEUID):</b> The NID field contains a copy of the EUI64 field in the Identify Namespace data structure (refer to the applicable I/O Command Set specification). If the EUI64 field of the Identify Namespace data structure is not supported, (i.e., EUI64 field is cleared to 0h), the controller shall not report a Namespace Identification Descriptor with a value of type 1h.</td> </tr> <tr> <td>2h</td> <td>10h</td> <td><b>Namespace Globally Unique Identifier (NGUID):</b> The NID field contains a copy of the NGUID field in the Identify Namespace data structure (refer to the applicable I/O Command Set specification). If the NGUID field of the Identify Namespace data structure is not supported (i.e., the NGUID field is cleared to 0h), the controller shall not report a Namespace Identification Descriptor with a value of type 2h.</td> </tr> <tr> <td>3h</td> <td>10h</td> <td><b>Namespace UUID (NUUID):</b> The NID field contains a 128-bit Universally Unique Identifier (UUID) as specified in RFC 9562. Refer to section 4.5.6. If the namespace does not support an IEEE Extended Unique Identifier (i.e., EUI64 field is cleared to 0h) and does not support a Namespace Globally Unique Identifier (i.e., the NGUID field is cleared to 0h), then the namespace shall report a Namespace Identification Descriptor with a value of type 3h.</td> </tr> <tr> <td>4h</td> <td>1h</td> <td><b>Command Set Identifier (CSI):</b> The NID field contains the Command Set Identifier that indicates the I/O Command Set that is associated with this namespace. Refer to Figure 311.</td> </tr> <tr> <td>5h to FFh</td> <td></td> <td>Reserved</td> </tr> </tbody> </table>	Value	Length (NIDL)	Definition	0h		Reserved	1h	8h	<b>IEEE Extended Unique Identifier (IEUID):</b> The NID field contains a copy of the EUI64 field in the Identify Namespace data structure (refer to the applicable I/O Command Set specification). If the EUI64 field of the Identify Namespace data structure is not supported, (i.e., EUI64 field is cleared to 0h), the controller shall not report a Namespace Identification Descriptor with a value of type 1h.	2h	10h	<b>Namespace Globally Unique Identifier (NGUID):</b> The NID field contains a copy of the NGUID field in the Identify Namespace data structure (refer to the applicable I/O Command Set specification). If the NGUID field of the Identify Namespace data structure is not supported (i.e., the NGUID field is cleared to 0h), the controller shall not report a Namespace Identification Descriptor with a value of type 2h.	3h	10h	<b>Namespace UUID (NUUID):</b> The NID field contains a 128-bit Universally Unique Identifier (UUID) as specified in RFC 9562. Refer to section 4.5.6. If the namespace does not support an IEEE Extended Unique Identifier (i.e., EUI64 field is cleared to 0h) and does not support a Namespace Globally Unique Identifier (i.e., the NGUID field is cleared to 0h), then the namespace shall report a Namespace Identification Descriptor with a value of type 3h.	4h	1h	<b>Command Set Identifier (CSI):</b> The NID field contains the Command Set Identifier that indicates the I/O Command Set that is associated with this namespace. Refer to Figure 311.	5h to FFh		Reserved
	Value	Length (NIDL)	Definition																			
	0h		Reserved																			
	1h	8h	<b>IEEE Extended Unique Identifier (IEUID):</b> The NID field contains a copy of the EUI64 field in the Identify Namespace data structure (refer to the applicable I/O Command Set specification). If the EUI64 field of the Identify Namespace data structure is not supported, (i.e., EUI64 field is cleared to 0h), the controller shall not report a Namespace Identification Descriptor with a value of type 1h.																			
	2h	10h	<b>Namespace Globally Unique Identifier (NGUID):</b> The NID field contains a copy of the NGUID field in the Identify Namespace data structure (refer to the applicable I/O Command Set specification). If the NGUID field of the Identify Namespace data structure is not supported (i.e., the NGUID field is cleared to 0h), the controller shall not report a Namespace Identification Descriptor with a value of type 2h.																			
3h	10h	<b>Namespace UUID (NUUID):</b> The NID field contains a 128-bit Universally Unique Identifier (UUID) as specified in RFC 9562. Refer to section 4.5.6. If the namespace does not support an IEEE Extended Unique Identifier (i.e., EUI64 field is cleared to 0h) and does not support a Namespace Globally Unique Identifier (i.e., the NGUID field is cleared to 0h), then the namespace shall report a Namespace Identification Descriptor with a value of type 3h.																				
4h	1h	<b>Command Set Identifier (CSI):</b> The NID field contains the Command Set Identifier that indicates the I/O Command Set that is associated with this namespace. Refer to Figure 311.																				
5h to FFh		Reserved																				
01	<b>Namespace Identifier Length (NIDL):</b> This field contains the length in bytes of the Namespace Identifier (NID) field. The total length of the Namespace Identification Descriptor in bytes is the value in this field plus four. If this field is cleared to 0h, then the end of the Namespace Identifier Descriptor list.																					
03:02	Reserved																					
(NIDL + 3):04	<b>Namespace Identifier (NID):</b> For an NIDT field value of 1h, 2h, and 3h, this field contains a value that is globally unique. The type of the value is specified by the Namespace Identifier Type (NIDT) field, and the size is specified by the Namespace Identifier Length (NIDL) field.																					

#### 5.1.13.2.4 NVM Set List (CNS 04h)

Figure 317 defines an NVM Set List. The data structure is an ordered list of NVM Set Attribute Entry data structures, sorted by NVM Set Identifier, starting with the first NVM Set Identifier supported by the NVM subsystem that is equal to or greater than the NVM Set Identifier specified by the CNS Specific Identifier

field as defined in Figure 316 and are accessible by the controller processing the command. The NVM Set List describes the attributes for each NVM Set in the list based on the NVM Set Attributes Entry in Figure 317.

**Figure 316: Command Dword 11 - CNS Specific Identifier**

Bits	Description
15:0	<b>NVM Set Identifier (NVMSETID):</b> This field specifies the NVM Set Identifier (refer to section 3.2.2) of the first NVM Set of the ordered list of NVM Set Attribute Entry data structures to be returned.

The NVM Set List shall not contain an entry cleared to 0h.

**Figure 317: NVM Set List**

Bytes	Description
<b>Header</b>	
00	<b>Number of Entries (NUMENT):</b> This field indicates the number of NVM Set Attributes Entries in the list. There are up to 31 entries in the list. A value of 0h indicates that there are no entries in the list.
127:01	Reserved
<b>NVM Set Attributes Entry List</b>	
255:128	<b>Entry 0:</b> This field contains the first NVM Set Attributes Entry in the list, if present.
383:256	<b>Entry 1:</b> This field contains the second NVM Set Attributes Entry in the list, if present.
...	...
((NUMENT-1)*128+255): ((NUMENT-1)*128+128)	<b>Entry NUMIDS-1:</b> This field contains the last NVM Set Attributes Entry in the list, if present.

**Figure 318: NVM Set Attributes Entry**

Bytes	Description
01:00	<b>NVM Set Identifier (NSETID):</b> This field indicates the identifier of the NVM Set in the NVM subsystem that is described by this entry.
03:02	<b>Endurance Group Identifier (ENDGID):</b> This field indicates the Endurance Group for this NVM Set. Refer to section 3.2.3.
07:04	Reserved
11:08	<b>Random 4 KiB Read Typical (R4KRT):</b> This field indicates the typical time to complete a 4 KiB random read in 100 nanosecond units when the NVM Set is in a Predictable Latency Mode Deterministic Window and there is 1 outstanding command per NVM Set.
15:12	<b>Optimal Write Size (OWS):</b> This field indicates the size in bytes for optimal write performance. A value of 0h indicates that no Optimal Write Size is specified. This field should be cleared to 0h when namespaces within an NVM Set have different User Data Formats that do not allow an Optimal Write Size to be specified.
31:16	<b>Total NVM Set Capacity (TNVMSC):</b> This field indicates the total NVM capacity in this NVM Set. The value is in bytes.
47:32	<b>Unallocated NVM Set Capacity (UNVMSC):</b> This field indicates the unallocated NVM capacity in this NVM Set. The value is in bytes.
127:48	Reserved

#### 5.1.13.2.5 I/O Command Set specific Identify Namespace data structure (CNS 05h)

An I/O Command Set specific Identify Namespace data structure (refer to the applicable I/O Command Set specification) is returned to the host for the specified namespace if the value in the Namespace Identifier (NSID) field is an active NSID. If the value in the NSID field specifies an inactive NSID, then the controller returns a zero filled data structure.

The specific Identify Namespace data structure that is returned by this command is specified by the Command Set Identifier (CSI) field (refer to Figure 311). If the I/O Command Set associated with the specified namespace does not support the Identify Namespace data structure specified by the CSI field, then the controller shall abort the command with a status code of Invalid Field in Command.

Each I/O Command Set specific Identify Namespace data structure specifies fields that define capabilities used by a host to format or create a namespace. If the NSID field is set to FFFFFFFFh, then the controller shall return an I/O Command Set specific Identify Namespace data structure for the I/O Command Set specified in the CSI field (refer to Figure 311) that has:

- the capability fields as specified by the I/O Command Set specific Identify Namespace data structure that requires a value that is the same value for all namespaces formatted using the User Data Formats associated with the Number of LBA Formats field; and
- fields that are not capability fields set to the value 0h.

If the controller supports the Namespace Management capability (refer to section 8.1.15) and the NSID field is set to FFFFFFFFh, then the controller shall return an I/O Command Set specific Identify Namespace data structure for the I/O Command Set specified in the CSI field.

If the controller does not support the Namespace Management capability and the NSID field is set to FFFFFFFFh, then the controller may abort the command with a status code of Invalid Namespace or Format.

#### **5.1.13.2.6 I/O Command Set specific Identify Controller data structure (CNS 06h)**

An I/O Command Set specific Identify Controller data structure is returned to the host for the controller processing the command. The specific Identify Controller data structure that is returned by this command is specified by the Command Set Identifier (CSI) field (refer to Figure 311). Data structures for specific I/O Command Sets are optionally defined by the I/O Command Set specifications. If the I/O Command Set specified by the CSI field does not have an Identify Controller data structure, then the controller shall return a zero filled data structure. If the host requests a data structure for an I/O Command Set that the controller does not support, the controller shall abort the command with a status code of Invalid Field in Command.

#### **5.1.13.2.7 I/O Command Set specific Active Namespace ID list (CNS 07h)**

A list of 1,024 namespace IDs is returned to the host containing active NSIDs in increasing order that are greater than the value specified in the Namespace Identifier (NSID) field of the command as specified by the Command Set Identifier (CSI) field of the command. Only namespaces associated with the I/O Command Set specified by the CSI value are returned. For CSI values that are not supported or not enabled the command is aborted with a status code of Invalid Field in Command.

The controller should abort the command with a status code of Invalid Namespace or Format if the NSID field is set to FFFFFFFEh or FFFFFFFFh. The NSID field may be cleared to 0h to retrieve a Namespace List including the namespace starting with NSID of 1h. The data structure returned is a Namespace List (refer to section 4.6.2).

#### **5.1.13.2.8 I/O Command Set Independent Identify Namespace data structure (CNS 08h)**

If the Namespace Identifier (NSID) field specifies an active NSID, then the I/O Command Set Independent Identify Namespace data structure (refer to Figure 319) is returned to the host for that specified namespace. If the value in the NSID field specifies an inactive NSID, then the controller returns a zero filled data structure.

The Reported column in Figure 319 specifies fields in the I/O Command Set Independent Identify Namespace data structure that define namespace capabilities used by a host to format or create a namespace. If the NSID field is set to FFFFFFFFh, then the controller shall return an I/O Command Set Independent Identify Namespace data structure that:

- for fields in Figure 319 that indicate “Yes” in the Reported column, contain the same value for all namespaces; and
- for fields in Figure 319 that indicate “No” in the Reported column, contain a value cleared to 0h.

If the controller supports the Namespace Management capability (refer to section 8.1.15) and the NSID field is set to FFFFFFFFh, then the controller shall return an I/O Command Set Independent Identify Namespace data structure. If the controller does not support the Namespace Management capability and



the NSID field is set to FFFFFFFFh, then the controller may abort the command with a status code of Invalid Namespace or Format.

**Figure 319: Identify – I/O Command Set Independent Identify Namespace Data Structure**

Bytes	O/M <sup>1</sup>	Description	Reported <sup>2</sup>												
00	M	<p><b>Common Namespace Features (NSFEAT):</b> This field defines features of the namespace.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:6</td> <td>Reserved</td> </tr> <tr> <td>5</td> <td> <p><b>Volatile Write Cache Not Present (VWCNP):</b> This bit indicates that a volatile write cache is not present for this namespace.</p> <p>If this bit is set to '1', then there is no volatile write cache present for this namespace and the host should ignore the Volatile Write Cache Present bit in the Identify Controller data structure (refer to Figure 312) for this namespace.</p> <p>If this bit is cleared to '0', then the presence of a volatile write cache for this namespace is defined in the Volatile Write Cache bit in the Identify Controller data structure (refer to Figure 312).</p> </td> </tr> <tr> <td>4</td> <td> <p><b>Rotational Media (RMEDIA):</b> If this bit is set to '1', then the namespace stores data on rotational media (refer to section 8.1.23). If this bit is cleared to '0', then the namespace does not store data on rotational media.</p> </td> </tr> <tr> <td>3</td> <td> <p><b>UID Reuse (UIDREUSE):</b> If this bit is set to '1', then the value in the NGUID field for this namespace, if non-zero, is never reused by the controller and that the value in the EUI64 field for this namespace, if non-zero, is never reused by the controller. If this bit is cleared to '0', then the NGUID value may be reused and the EUI64 value may be reused by the controller for a new namespace created after this namespace is deleted. This bit shall be cleared to '0' if both NGUID and EUI64 fields are cleared to 0h. Refer to section 4.7.1.</p> </td> </tr> <tr> <td>2:0</td> <td>Reserved</td> </tr> </tbody> </table>	Bits	Description	7:6	Reserved	5	<p><b>Volatile Write Cache Not Present (VWCNP):</b> This bit indicates that a volatile write cache is not present for this namespace.</p> <p>If this bit is set to '1', then there is no volatile write cache present for this namespace and the host should ignore the Volatile Write Cache Present bit in the Identify Controller data structure (refer to Figure 312) for this namespace.</p> <p>If this bit is cleared to '0', then the presence of a volatile write cache for this namespace is defined in the Volatile Write Cache bit in the Identify Controller data structure (refer to Figure 312).</p>	4	<p><b>Rotational Media (RMEDIA):</b> If this bit is set to '1', then the namespace stores data on rotational media (refer to section 8.1.23). If this bit is cleared to '0', then the namespace does not store data on rotational media.</p>	3	<p><b>UID Reuse (UIDREUSE):</b> If this bit is set to '1', then the value in the NGUID field for this namespace, if non-zero, is never reused by the controller and that the value in the EUI64 field for this namespace, if non-zero, is never reused by the controller. If this bit is cleared to '0', then the NGUID value may be reused and the EUI64 value may be reused by the controller for a new namespace created after this namespace is deleted. This bit shall be cleared to '0' if both NGUID and EUI64 fields are cleared to 0h. Refer to section 4.7.1.</p>	2:0	Reserved	No
		Bits	Description												
		7:6	Reserved												
		5	<p><b>Volatile Write Cache Not Present (VWCNP):</b> This bit indicates that a volatile write cache is not present for this namespace.</p> <p>If this bit is set to '1', then there is no volatile write cache present for this namespace and the host should ignore the Volatile Write Cache Present bit in the Identify Controller data structure (refer to Figure 312) for this namespace.</p> <p>If this bit is cleared to '0', then the presence of a volatile write cache for this namespace is defined in the Volatile Write Cache bit in the Identify Controller data structure (refer to Figure 312).</p>												
		4	<p><b>Rotational Media (RMEDIA):</b> If this bit is set to '1', then the namespace stores data on rotational media (refer to section 8.1.23). If this bit is cleared to '0', then the namespace does not store data on rotational media.</p>												
3	<p><b>UID Reuse (UIDREUSE):</b> If this bit is set to '1', then the value in the NGUID field for this namespace, if non-zero, is never reused by the controller and that the value in the EUI64 field for this namespace, if non-zero, is never reused by the controller. If this bit is cleared to '0', then the NGUID value may be reused and the EUI64 value may be reused by the controller for a new namespace created after this namespace is deleted. This bit shall be cleared to '0' if both NGUID and EUI64 fields are cleared to 0h. Refer to section 4.7.1.</p>														
2:0	Reserved														
01	O	<p><b>Namespace Multi-path I/O and Namespace Sharing Capabilities (NMIC):</b> This field specifies multi-path I/O and namespace sharing capabilities of the namespace.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:2</td> <td>Reserved</td> </tr> <tr> <td>1</td> <td> <p><b>Dispersed Namespace (DISNS):</b> If this bit is set to '1', then the namespace is a dispersed namespace (refer to section 8.1.9). If this bit is set to '1', then the namespace shall be a shared namespace (i.e., the Shared Namespace (SHRNS) bit shall also be set to '1'). If this bit is cleared to '0', then the namespace is not a dispersed namespace.</p> </td> </tr> <tr> <td>0</td> <td> <p><b>Shared Namespace (SHRNS):</b> If this bit is set to '1', then the namespace may be attached to two or more controllers in the NVM subsystem concurrently (i.e., may be a shared namespace). If this bit is cleared to '0', then the namespace is a private namespace and is able to be attached to only one controller at a time.</p> </td> </tr> </tbody> </table>	Bits	Description	7:2	Reserved	1	<p><b>Dispersed Namespace (DISNS):</b> If this bit is set to '1', then the namespace is a dispersed namespace (refer to section 8.1.9). If this bit is set to '1', then the namespace shall be a shared namespace (i.e., the Shared Namespace (SHRNS) bit shall also be set to '1'). If this bit is cleared to '0', then the namespace is not a dispersed namespace.</p>	0	<p><b>Shared Namespace (SHRNS):</b> If this bit is set to '1', then the namespace may be attached to two or more controllers in the NVM subsystem concurrently (i.e., may be a shared namespace). If this bit is cleared to '0', then the namespace is a private namespace and is able to be attached to only one controller at a time.</p>	Yes				
		Bits	Description												
		7:2	Reserved												
1	<p><b>Dispersed Namespace (DISNS):</b> If this bit is set to '1', then the namespace is a dispersed namespace (refer to section 8.1.9). If this bit is set to '1', then the namespace shall be a shared namespace (i.e., the Shared Namespace (SHRNS) bit shall also be set to '1'). If this bit is cleared to '0', then the namespace is not a dispersed namespace.</p>														
0	<p><b>Shared Namespace (SHRNS):</b> If this bit is set to '1', then the namespace may be attached to two or more controllers in the NVM subsystem concurrently (i.e., may be a shared namespace). If this bit is cleared to '0', then the namespace is a private namespace and is able to be attached to only one controller at a time.</p>														

Figure 319: Identify – I/O Command Set Independent Identify Namespace Data Structure

Bytes	O/M <sup>1</sup>	Description	Reported <sup>2</sup>																		
02	0	<p><b>Reservation Capabilities (RESCAP):</b> This field indicates the reservation capabilities of the namespace. A value of 0h in this field indicates that reservations are not supported by this namespace. Refer to section 8.1.22 for more details.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7</td> <td><b>Ignore Existing Key Support (IEKS):</b> If this bit is set to '1', then the Ignore Existing Key is used as defined in NVM Express Base Specification revision 1.3 or later. If this bit is cleared to '0', then the Ignore Existing Key is used as defined in NVM Express Base Specification revision 1.2.1 or earlier. This bit shall be set to '1' if the controller supports revision 1.3 or later as indicated in the Version register.</td> </tr> <tr> <td>6</td> <td><b>Exclusive Access – All Registrants Support (EAARS):</b> If this bit is set to '1', then the namespace supports the Exclusive Access – All Registrants reservation type. If this bit is cleared to '0', then the namespace does not support the Exclusive Access – All Registrants reservation type.</td> </tr> <tr> <td>5</td> <td><b>Write Exclusive – All Registrants Support (WEARS):</b> If this bit is set to '1', then the namespace supports the Write Exclusive – All Registrants reservation type. If this bit is cleared to '0', then the namespace does not support the Write Exclusive – All Registrants reservation type.</td> </tr> <tr> <td>4</td> <td><b>Exclusive Access – Registrants Only Support (EAROS):</b> If this bit is set to '1', then the namespace supports the Exclusive Access – Registrants Only reservation type. If this bit is cleared to '0', then the namespace does not support the Exclusive Access – Registrants Only reservation type.</td> </tr> <tr> <td>3</td> <td><b>Write Exclusive – Registrants Only Support (WEROS):</b> If this bit is set to '1', then the namespace supports the Write Exclusive – Registrants Only reservation type. If this bit is cleared to '0', then the namespace does not support the Write Exclusive – Registrants Only reservation type.</td> </tr> <tr> <td>2</td> <td><b>Exclusive Access Support (EAS):</b> If this bit is set to '1', then the namespace supports the Exclusive Access reservation type. If this bit is cleared to '0', then the namespace does not support the Exclusive Access reservation type.</td> </tr> <tr> <td>1</td> <td><b>Write Exclusive Support (WES):</b> If this bit is set to '1', then the namespace supports the Write Exclusive reservation type. If this bit is cleared to '0', then the namespace does not support the Write Exclusive reservation type.</td> </tr> <tr> <td>0</td> <td><b>Persist Through Power Loss Support (PTPLS):</b> If this bit is set to '1', then the namespace supports the Persist Through Power Loss capability. If this bit is cleared to '0', then the namespace does not support the Persist Through Power Loss capability.</td> </tr> </tbody> </table>	Bits	Description	7	<b>Ignore Existing Key Support (IEKS):</b> If this bit is set to '1', then the Ignore Existing Key is used as defined in NVM Express Base Specification revision 1.3 or later. If this bit is cleared to '0', then the Ignore Existing Key is used as defined in NVM Express Base Specification revision 1.2.1 or earlier. This bit shall be set to '1' if the controller supports revision 1.3 or later as indicated in the Version register.	6	<b>Exclusive Access – All Registrants Support (EAARS):</b> If this bit is set to '1', then the namespace supports the Exclusive Access – All Registrants reservation type. If this bit is cleared to '0', then the namespace does not support the Exclusive Access – All Registrants reservation type.	5	<b>Write Exclusive – All Registrants Support (WEARS):</b> If this bit is set to '1', then the namespace supports the Write Exclusive – All Registrants reservation type. If this bit is cleared to '0', then the namespace does not support the Write Exclusive – All Registrants reservation type.	4	<b>Exclusive Access – Registrants Only Support (EAROS):</b> If this bit is set to '1', then the namespace supports the Exclusive Access – Registrants Only reservation type. If this bit is cleared to '0', then the namespace does not support the Exclusive Access – Registrants Only reservation type.	3	<b>Write Exclusive – Registrants Only Support (WEROS):</b> If this bit is set to '1', then the namespace supports the Write Exclusive – Registrants Only reservation type. If this bit is cleared to '0', then the namespace does not support the Write Exclusive – Registrants Only reservation type.	2	<b>Exclusive Access Support (EAS):</b> If this bit is set to '1', then the namespace supports the Exclusive Access reservation type. If this bit is cleared to '0', then the namespace does not support the Exclusive Access reservation type.	1	<b>Write Exclusive Support (WES):</b> If this bit is set to '1', then the namespace supports the Write Exclusive reservation type. If this bit is cleared to '0', then the namespace does not support the Write Exclusive reservation type.	0	<b>Persist Through Power Loss Support (PTPLS):</b> If this bit is set to '1', then the namespace supports the Persist Through Power Loss capability. If this bit is cleared to '0', then the namespace does not support the Persist Through Power Loss capability.	No
		Bits	Description																		
		7	<b>Ignore Existing Key Support (IEKS):</b> If this bit is set to '1', then the Ignore Existing Key is used as defined in NVM Express Base Specification revision 1.3 or later. If this bit is cleared to '0', then the Ignore Existing Key is used as defined in NVM Express Base Specification revision 1.2.1 or earlier. This bit shall be set to '1' if the controller supports revision 1.3 or later as indicated in the Version register.																		
		6	<b>Exclusive Access – All Registrants Support (EAARS):</b> If this bit is set to '1', then the namespace supports the Exclusive Access – All Registrants reservation type. If this bit is cleared to '0', then the namespace does not support the Exclusive Access – All Registrants reservation type.																		
		5	<b>Write Exclusive – All Registrants Support (WEARS):</b> If this bit is set to '1', then the namespace supports the Write Exclusive – All Registrants reservation type. If this bit is cleared to '0', then the namespace does not support the Write Exclusive – All Registrants reservation type.																		
		4	<b>Exclusive Access – Registrants Only Support (EAROS):</b> If this bit is set to '1', then the namespace supports the Exclusive Access – Registrants Only reservation type. If this bit is cleared to '0', then the namespace does not support the Exclusive Access – Registrants Only reservation type.																		
		3	<b>Write Exclusive – Registrants Only Support (WEROS):</b> If this bit is set to '1', then the namespace supports the Write Exclusive – Registrants Only reservation type. If this bit is cleared to '0', then the namespace does not support the Write Exclusive – Registrants Only reservation type.																		
		2	<b>Exclusive Access Support (EAS):</b> If this bit is set to '1', then the namespace supports the Exclusive Access reservation type. If this bit is cleared to '0', then the namespace does not support the Exclusive Access reservation type.																		
		1	<b>Write Exclusive Support (WES):</b> If this bit is set to '1', then the namespace supports the Write Exclusive reservation type. If this bit is cleared to '0', then the namespace does not support the Write Exclusive reservation type.																		
0	<b>Persist Through Power Loss Support (PTPLS):</b> If this bit is set to '1', then the namespace supports the Persist Through Power Loss capability. If this bit is cleared to '0', then the namespace does not support the Persist Through Power Loss capability.																				

**Figure 319: Identify – I/O Command Set Independent Identify Namespace Data Structure**

Bytes	O/M <sup>1</sup>	Description	Reported <sup>2</sup>						
03	O	<p><b>Format Progress Indicator (FPI):</b> If a format operation is in progress, this field indicates the percentage of the namespace that remains to be formatted.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7</td> <td><b>Format Progress Indicator Support (FPIS):</b> If this bit is set to '1', then the namespace supports the Format Progress Indicator defined by the RFNVM field. If this bit is cleared to '0', then the namespace does not support the Format Progress Indicator.</td> </tr> <tr> <td>6:0</td> <td><b>Remaining Format NVM (RFNVM):</b> This field indicates the percentage of the Format NVM command that remains to be completed (e.g., a value of 25 indicates that 75% of the Format NVM command has been completed and 25% remains to be completed). If the FPIS bit is set to '1', then a value of 0h indicates that the namespace is formatted with the format specified by Identify Namespace data structures (refer to section 1.5.49) and there is no Format NVM command in progress. If the FPIS bit is cleared to '0', then this field shall be cleared to 0h.</td> </tr> </tbody> </table>	Bits	Description	7	<b>Format Progress Indicator Support (FPIS):</b> If this bit is set to '1', then the namespace supports the Format Progress Indicator defined by the RFNVM field. If this bit is cleared to '0', then the namespace does not support the Format Progress Indicator.	6:0	<b>Remaining Format NVM (RFNVM):</b> This field indicates the percentage of the Format NVM command that remains to be completed (e.g., a value of 25 indicates that 75% of the Format NVM command has been completed and 25% remains to be completed). If the FPIS bit is set to '1', then a value of 0h indicates that the namespace is formatted with the format specified by Identify Namespace data structures (refer to section 1.5.49) and there is no Format NVM command in progress. If the FPIS bit is cleared to '0', then this field shall be cleared to 0h.	No
Bits	Description								
7	<b>Format Progress Indicator Support (FPIS):</b> If this bit is set to '1', then the namespace supports the Format Progress Indicator defined by the RFNVM field. If this bit is cleared to '0', then the namespace does not support the Format Progress Indicator.								
6:0	<b>Remaining Format NVM (RFNVM):</b> This field indicates the percentage of the Format NVM command that remains to be completed (e.g., a value of 25 indicates that 75% of the Format NVM command has been completed and 25% remains to be completed). If the FPIS bit is set to '1', then a value of 0h indicates that the namespace is formatted with the format specified by Identify Namespace data structures (refer to section 1.5.49) and there is no Format NVM command in progress. If the FPIS bit is cleared to '0', then this field shall be cleared to 0h.								
07:04	O	<p><b>ANA Group Identifier (ANAGRPID):</b> For NSID other than FFFFFFFFh, this field indicates the ANA Group Identifier of the ANA group (refer to section 8.1.1.2) of which the namespace is a member. Each namespace that is attached to a controller that supports Asymmetric Namespace Access Reporting (refer to the CMIC field) shall report a valid ANAGRPID. If the controller does not support Asymmetric Namespace Access Reporting, then this field shall be cleared to 0h.</p> <p>If the value in this field changes and Asymmetric Namespace Access Change Notices are supported and enabled, then the controller shall issue an Asymmetric Namespace Access Change Notice.</p>	No						
08	O	<p><b>Namespace Attributes (NSATTR):</b> This field specifies attributes of the namespace.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td><b>Currently Write Protected (CWP):</b> If this bit is set to '1', then the namespace is currently write protected due to any condition (e.g., namespace write protection set for the namespace, media errors) and all write access to the namespace shall fail. If this bit is cleared to '0', then the namespace is not currently write protected.</td> </tr> </tbody> </table>	Bits	Description	7:1	Reserved	0	<b>Currently Write Protected (CWP):</b> If this bit is set to '1', then the namespace is currently write protected due to any condition (e.g., namespace write protection set for the namespace, media errors) and all write access to the namespace shall fail. If this bit is cleared to '0', then the namespace is not currently write protected.	No
Bits	Description								
7:1	Reserved								
0	<b>Currently Write Protected (CWP):</b> If this bit is set to '1', then the namespace is currently write protected due to any condition (e.g., namespace write protection set for the namespace, media errors) and all write access to the namespace shall fail. If this bit is cleared to '0', then the namespace is not currently write protected.								
09		Reserved							
11:10	O	<p><b>NVM Set Identifier (NVMSETID):</b> For NSID other than FFFFFFFFh, this field indicates the NVM Set with which this namespace is associated. If NVM Sets are not supported by the controller, then this field shall be cleared to 0h.</p> <p>For namespaces that do not contain formatted storage this field shall be cleared to 0h.</p>	No						
13:12	O	<p><b>Endurance Group Identifier (ENDGID):</b> For NSID other than FFFFFFFFh, this field indicates the Endurance Group with which this namespace is associated. If Endurance Groups are not supported by the controller, then this field shall be cleared to 0h.</p> <p>For namespaces that do not contain formatted storage this field shall be cleared to 0h.</p>	No						

Figure 319: Identify – I/O Command Set Independent Identify Namespace Data Structure

Bytes	O/M <sup>1</sup>	Description	Reported <sup>2</sup>																		
14	M	<p><b>Namespace Status (NSTAT):</b> This field indicates the status of the namespace with the specified NSID.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:3</td> <td>Reserved</td> </tr> <tr> <td>2:1</td> <td> <p><b>I/O Impacted (IOI):</b> This field indicates whether I/O performance is currently degraded (e.g., I/O command processing is delayed or slow).</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>I/O performance degradation is not reported</td> </tr> <tr> <td>01b</td> <td>Reserved</td> </tr> <tr> <td>10b</td> <td>I/O performance is not currently degraded</td> </tr> <tr> <td>11b</td> <td>I/O performance is currently degraded</td> </tr> </tbody> </table> </td> </tr> <tr> <td>0</td> <td> <p><b>Namespace Ready (NRDY):</b> A value of '1' indicates that the namespace is ready (refer to section 3.5.3). A value of '0' indicates that the namespace is not ready.</p> </td> </tr> </tbody> </table>	Bits	Description	7:3	Reserved	2:1	<p><b>I/O Impacted (IOI):</b> This field indicates whether I/O performance is currently degraded (e.g., I/O command processing is delayed or slow).</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>I/O performance degradation is not reported</td> </tr> <tr> <td>01b</td> <td>Reserved</td> </tr> <tr> <td>10b</td> <td>I/O performance is not currently degraded</td> </tr> <tr> <td>11b</td> <td>I/O performance is currently degraded</td> </tr> </tbody> </table>	Value	Definition	00b	I/O performance degradation is not reported	01b	Reserved	10b	I/O performance is not currently degraded	11b	I/O performance is currently degraded	0	<p><b>Namespace Ready (NRDY):</b> A value of '1' indicates that the namespace is ready (refer to section 3.5.3). A value of '0' indicates that the namespace is not ready.</p>	No
Bits	Description																				
7:3	Reserved																				
2:1	<p><b>I/O Impacted (IOI):</b> This field indicates whether I/O performance is currently degraded (e.g., I/O command processing is delayed or slow).</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>I/O performance degradation is not reported</td> </tr> <tr> <td>01b</td> <td>Reserved</td> </tr> <tr> <td>10b</td> <td>I/O performance is not currently degraded</td> </tr> <tr> <td>11b</td> <td>I/O performance is currently degraded</td> </tr> </tbody> </table>	Value	Definition	00b	I/O performance degradation is not reported	01b	Reserved	10b	I/O performance is not currently degraded	11b	I/O performance is currently degraded										
Value	Definition																				
00b	I/O performance degradation is not reported																				
01b	Reserved																				
10b	I/O performance is not currently degraded																				
11b	I/O performance is currently degraded																				
0	<p><b>Namespace Ready (NRDY):</b> A value of '1' indicates that the namespace is ready (refer to section 3.5.3). A value of '0' indicates that the namespace is not ready.</p>																				
15	O	<p><b>Key Per I/O Status (KPIOS):</b> This field indicates the namespace Key Per I/O capability status.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:2</td> <td>Reserved</td> </tr> <tr> <td>1</td> <td> <p><b>Key Per I/O Supported in Namespace (KPIOSNS):</b> If this bit is set to '1', then the Key Per I/O capability is supported by the namespace. If this bit is cleared to '0', then the Key Per I/O capability is not supported by the namespace.</p> </td> </tr> <tr> <td>0</td> <td> <p><b>Key Per I/O Enabled in Namespace (KPIOENS):</b> If this bit is set to '1', then the Key Per I/O capability is enabled on the namespace. The mechanism to enable the Key Per I/O capability on the namespace is outside the scope of this specification (refer to section 8.1.11). If this bit is cleared to '0', then the Key Per I/O capability is disabled on the namespace.</p> <p>If the KPIOSNS bit is cleared to '0', then this bit shall be cleared to '0'.</p> </td> </tr> </tbody> </table>	Bits	Description	7:2	Reserved	1	<p><b>Key Per I/O Supported in Namespace (KPIOSNS):</b> If this bit is set to '1', then the Key Per I/O capability is supported by the namespace. If this bit is cleared to '0', then the Key Per I/O capability is not supported by the namespace.</p>	0	<p><b>Key Per I/O Enabled in Namespace (KPIOENS):</b> If this bit is set to '1', then the Key Per I/O capability is enabled on the namespace. The mechanism to enable the Key Per I/O capability on the namespace is outside the scope of this specification (refer to section 8.1.11). If this bit is cleared to '0', then the Key Per I/O capability is disabled on the namespace.</p> <p>If the KPIOSNS bit is cleared to '0', then this bit shall be cleared to '0'.</p>	Yes										
Bits	Description																				
7:2	Reserved																				
1	<p><b>Key Per I/O Supported in Namespace (KPIOSNS):</b> If this bit is set to '1', then the Key Per I/O capability is supported by the namespace. If this bit is cleared to '0', then the Key Per I/O capability is not supported by the namespace.</p>																				
0	<p><b>Key Per I/O Enabled in Namespace (KPIOENS):</b> If this bit is set to '1', then the Key Per I/O capability is enabled on the namespace. The mechanism to enable the Key Per I/O capability on the namespace is outside the scope of this specification (refer to section 8.1.11). If this bit is cleared to '0', then the Key Per I/O capability is disabled on the namespace.</p> <p>If the KPIOSNS bit is cleared to '0', then this bit shall be cleared to '0'.</p>																				
17:16	O	<p><b>Maximum Key Tag (MAXKT):</b> This field indicates the maximum Key Tag field value allocated to this namespace in an I/O command that supports a Key Tag field.</p> <p>If the KPIOENS bit is set to '1', then the valid Key Tag field values for this namespace are 0h to the value of this field.</p> <p>If the KPIOENS bit is cleared to '0', then this field is reserved.</p>	No																		
19:18		Reserved																			
23:20	O	<p><b>Reachability Group Identifier (RGRPID):</b> For NSID other than FFFFFFFh, this field indicates the Reachability Group Identifier of the reachability group (refer to section 8.1.19.1) of which the namespace is a member. Each namespace that is attached to a controller that supports Reachability Group Reporting (refer to the RRSUP bit in the CRCAP field in Figure 312)) shall report a valid RGRPID. If the controller does not support Reachability Group Reporting, then this field shall be cleared to 0h.</p> <p>If the value in this field changes and Reachability Group Change Notices are supported and enabled, then the controller shall issue a Reachability Group Change Notice.</p>	No																		
4095:24		Reserved																			
<p>Notes:</p> <ol style="list-style-type: none"> <li>O/M definition: O = Optional, M = Mandatory.</li> <li>Identifies fields that report information for the Identify command when querying the capabilities of LBA formats.</li> </ol>																					

#### **5.1.13.2.9 Allocated Namespace ID list (CNS 10h)**

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

The controller shall abort the command with a status code of Invalid Namespace or Format if the NSID field is set to FFFFFFFEh or FFFFFFFFh. The NSID field may be cleared to 0h to retrieve a Namespace List including the namespace starting with NSID of 1h. The data structure returned is a Namespace List (refer to section 4.6.2).

#### **5.1.13.2.10 Identify Namespace data structure for an Allocated Namespace ID (CNS 11h)**

The Identify Namespace data structure (refer to the NVM Command Set Specification) is returned to the host for the specified namespace if the value in the Namespace Identifier (NSID) field is an allocated NSID. If the value in the NSID field specifies an unallocated NSID, then the controller returns a zero filled data structure. If the specified namespace is not associated with an I/O Command Set that specifies logical blocks (e.g., the NVM Command Set), then the controller shall abort the command with a status code of Invalid I/O Command Set.

If the value in the NSID field specifies an invalid NSID, then the controller shall abort the command with a status code of Invalid Namespace or Format. If the NSID field is set to FFFFFFFFh, then the controller shall abort the command with a status code of Invalid Namespace or Format.

#### **5.1.13.2.11 Namespace Attached Controller list (CNS 12h)**

A Controller List (refer to section 4.6.1) of up to 2,047 controller identifiers is returned containing a controller identifier greater than or equal to the value specified in the Controller Identifier (CDW10.CNTID) field. The list contains controller identifiers of controllers that have the specified namespace attached. If the NSID field is set to FFFFFFFFh, then the controller shall abort the command with a status code of Invalid Field in Command.

#### **5.1.13.2.12 Controller list (CNS 13h)**

A Controller List (refer to section 4.6.1) of up to 2,047 controller identifiers of I/O controllers is returned containing controller identifiers greater than or equal to the value specified in the Controller Identifier (CDW10.CNTID) field.

#### **5.1.13.2.13 Namespace Granularity List (CNS 16h)**

If the controller supports reporting of Namespace Granularity refer to the applicable I/O Command Set specification for details.

The controller shall abort the command with a status code of Invalid I/O Command Set if the Command Set Identifier is not associated with an I/O Command Set that supports the Namespace Granularity List.

#### **5.1.13.2.14 UUID List (CNS 17h)**

The format of the UUID List is defined in Figure 320. Each UUID List entry is either 0h, the NVMe Invalid UUID, or a valid UUID. Valid UUIDs are those which are non-zero and are not the NVMe Invalid UUID (refer to section 8.1.28).

If the UUID List bit is set to '1' in the Controller Attributes (CTRATT) field in the Identify Controller data structure (refer to Figure 312), then:

- The UUID List shall contain at least one valid UUID (refer to section 8.1.28);
- The UUID 1 field shall contain a non-zero value; and
- A UUID field cleared to 0h indicates the end of the UUID List.

The list may be in any order.

**Figure 320: UUID List**

Bytes	Description
31:00	Reserved
63:32	<b>UUID 1 (UUID1)</b> : This field contains the first UUID List Entry in the list.
95:64	<b>UUID 2 (UUID2)</b> : This field contains the second UUID List Entry in the list, if present, otherwise cleared to 0h.
...	...
4063:4032	<b>UUID 126 (UUID126)</b> : This field contains the last non-zero UUID List Entry in the list, if present, otherwise cleared to 0h.
4095:4064	<b>UUID 127 (UUID127)</b> : This field shall be cleared to 0h.

The format of a UUID List Entry is defined in Figure 321.

**Figure 321: UUID List Entry**

Bytes	Description																	
00	<b>UUID Lists Entry Header (ULEH):</b>																	
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:2</td> <td>Reserved</td> </tr> <tr> <td rowspan="4">1:0</td> <td><b>Identifier Association (IDASSOC)</b>: This field indicates whether the UUID is associated with a vendor.</td> </tr> <tr> <td> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>No association reported.</td> </tr> <tr> <td>01b</td> <td>The UUID is associated with the vendor reported in the PCI Vendor ID field of the Identify Controller data structure (refer to Figure 312).</td> </tr> <tr> <td>10b</td> <td>The UUID is associated with the vendor reported in the PCI Subsystem Vendor ID field of the Identify Controller data structure.</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table> </td> </tr> </tbody> </table>	Bits	Description	7:2	Reserved	1:0	<b>Identifier Association (IDASSOC)</b> : This field indicates whether the UUID is associated with a vendor.	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>No association reported.</td> </tr> <tr> <td>01b</td> <td>The UUID is associated with the vendor reported in the PCI Vendor ID field of the Identify Controller data structure (refer to Figure 312).</td> </tr> <tr> <td>10b</td> <td>The UUID is associated with the vendor reported in the PCI Subsystem Vendor ID field of the Identify Controller data structure.</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	No association reported.	01b	The UUID is associated with the vendor reported in the PCI Vendor ID field of the Identify Controller data structure (refer to Figure 312).	10b	The UUID is associated with the vendor reported in the PCI Subsystem Vendor ID field of the Identify Controller data structure.	11b	Reserved
	Bits	Description																
	7:2	Reserved																
	1:0	<b>Identifier Association (IDASSOC)</b> : This field indicates whether the UUID is associated with a vendor.																
<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>No association reported.</td> </tr> <tr> <td>01b</td> <td>The UUID is associated with the vendor reported in the PCI Vendor ID field of the Identify Controller data structure (refer to Figure 312).</td> </tr> <tr> <td>10b</td> <td>The UUID is associated with the vendor reported in the PCI Subsystem Vendor ID field of the Identify Controller data structure.</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table>		Value	Definition	00b	No association reported.	01b	The UUID is associated with the vendor reported in the PCI Vendor ID field of the Identify Controller data structure (refer to Figure 312).	10b	The UUID is associated with the vendor reported in the PCI Subsystem Vendor ID field of the Identify Controller data structure.	11b	Reserved							
Value		Definition																
00b		No association reported.																
01b	The UUID is associated with the vendor reported in the PCI Vendor ID field of the Identify Controller data structure (refer to Figure 312).																	
10b	The UUID is associated with the vendor reported in the PCI Subsystem Vendor ID field of the Identify Controller data structure.																	
11b	Reserved																	
15:01	Reserved																	
31:16	<b>Universally Unique Identifier (UUID)</b> : This field contains a 128-bit Universally Unique Identifier (UUID) as specified in RFC 9562. Refer to section 4.5.6.																	

#### 5.1.13.2.15 Domain List (CNS 18h)

Figure 323 defines a Domain List. The data structure is an ordered list by Domain Identifier, starting with the first Domain Identifier that is greater than or equal to the Domain Identifier specified by the CNS Specific Identifier field as defined in Figure 322 and is accessible by the controller processing the command. The Domain List describes the attributes for each Domain in the list based on the Domain Attributes Entry in Figure 324.

**Figure 322: Command Dword 11 - CNS Specific Identifier**

Bits	Description
15:0	<b>Domain Identifier (DID)</b> : This field specifies the Domain Identifier (refer to section 3.2.5.3) of the first Domain of the ordered list of Domain Attribute Entry data structures to be returned.

**Figure 323: Domain List**

Bytes	Description
<b>Header</b>	
00	<b>Number of Entries (NUMENT)</b> : This field indicates the number of Domain Attributes Entries in the list. There are up to 31 entries in the list. A value of 0h indicates that there are no entries in the list.
127:01	Reserved

**Figure 323: Domain List**

Bytes	Description
<b>Domain Attributes Entry List</b>	
255:128	<b>Entry 0:</b> This field contains the first Domain Attributes Entry in the list, if present.
383:256	<b>Entry 1:</b> This field contains the second Domain Attributes Entry in the list, if present.
...	...
((NUMENT-1)*128+255): ((NUMENT-1)*128+128)	<b>Entry NUMENT-1:</b> This field contains the last Domain Attributes Entry in the list, if present.

**Figure 324: Domain Attributes Entry**

Bytes	Description
01:00	<b>Domain Identifier (DID):</b> This field indicates the identifier of the Domain accessible by the controller that is described by this entry.
15:02	Reserved
31:16	<b>Total Domain Capacity (TDC):</b> This field indicates the total NVM capacity in this Domain. The value is in bytes.
47:32	<b>Unallocated Domain Capacity (UDC):</b> This field indicates the unallocated NVM capacity in this Domain. The value is in bytes.
63:48	<b>Max Endurance Group Domain Capacity (MEGDC):</b> This field indicates the maximum capacity of a single Endurance Group in this Domain. If this field is cleared to 0h, then the NVM subsystem does not report a maximum Endurance Group Domain Capacity value.
127:64	Reserved

**5.1.13.2.16 Endurance Group List (19h)**

An Endurance Group List (refer to Figure 326) of up to 2,047 Endurance Group Identifiers in increasing order is returned containing an Endurance Group Identifier greater than or equal to the Endurance Group Identifier specified by the CNS Specific Identifier field as defined in Figure 325. The list contains Endurance Group Identifiers of Endurance Groups that are accessible by the controller processing the command. If the value specified in the Endurance Group Identifier is greater than ENDGIDMAX, then the controller shall complete the command with a status code of Successful Completion and return an Endurance Group List containing no Endurance Group Identifiers.

**Figure 325: Command Dword 11 - CNS Specific Identifier**

Bits	Description
15:0	<b>Endurance Group Identifier (ENGGID):</b> This field specifies the Endurance Group Identifier (refer to section 3.2.3) of the first Endurance Group of the ordered list of Endurance Group Identifiers to be returned.

**Figure 326: Endurance Group List**

Bytes	Description
<b>Header</b>	
01:00	<b>Number of Endurance Group Identifiers (NEGIDS):</b> This field contains the number of Endurance Group Identifiers in the list. There may be up to 2,047 identifiers in the list. If this field is cleared to 0h, then no Endurance Group Identifiers are in the list.
<b>Endurance Group Identifier List</b>	
03:02	<b>Identifier 0:</b> This field contains the first Endurance Group Identifier in the list, if any.
05:04	<b>Identifier 1:</b> This field contains the second Endurance Group Identifier in the list, if any.
...	...
(NEGIDS*2+1): (NEGIDS *2)	<b>Identifier NEGIDS-1:</b> This field contains the last Endurance Group Identifier in the list.

### 5.1.13.2.17 I/O Command Set specific Allocated Namespace ID list (CNS 1Ah)

A list of up to 1,024 namespace IDs is returned to the host containing allocated NSIDs in increasing order that are greater than the value specified in the Namespace Identifier (NSID) field of the Identify command and as specified by the Command Set Identifier (CSI) field of the command. Only NSIDs for namespaces associated with the I/O Command Set specified in CSI are returned. For CSI values not supported by the controller the command is aborted with a status code of Invalid Field in Command.

The controller should abort the command with a status code of Invalid Namespace or Format if the NSID field is set to FFFFFFFEh or FFFFFFFFh. The NSID field may be cleared to 0h to retrieve a Namespace List including the namespace starting with NSID of 1h. The data structure returned is a Namespace List (refer to section 4.6.2).

### 5.1.13.2.18 I/O Command Set specific Identify Namespace data structure for an Allocated Namespace ID (CNS 1Bh)

An I/O Command Set specific Identify Namespace data structure (refer to section 5.1.13.2.5) is returned to the host for the specified namespace if the value in the Namespace Identifier (NSID) field is an allocated NSID. If the value in the NSID field is an unallocated NSID, then the controller returns a zero filled data structure.

The specific Identify Namespace data structure that is returned by this command is specified by the Command Set Identifier (CSI) field in the command (refer to Figure 311). If the I/O Command Set associated with the specified namespace does not support the specific Identify Namespace data structure specified by the CSI field, then the controller shall abort the command with a status code of Invalid Field in Command.

If the value in the NSID field is an invalid NSID, then the controller shall abort the command with a status code of Invalid Namespace or Format. If the NSID field is set to FFFFFFFFh, then the controller should abort the command with a status code of Invalid Namespace or Format.

### 5.1.13.2.19 Identify I/O Command Set data structure (CNS 1Ch)

The Identify I/O Command Set data structure (refer to Figure 327) is returned to the host for the controller specified in the Controller ID (CNTID) field of the command if the CNTID field does not have a value of FFFFh. If the CNTID field has a value of FFFFh, then the Identify I/O Command Set data structure is returned to the host for the controller processing the command.

This CNS value shall be implemented if the CAP.CSS.IOCSS bit is set to '1'.

The Identify I/O Command Set data structure consists of an array of I/O Command Set Vectors (refer to Figure 328) that describe the I/O Command Sets that the controller supports and the combination of supported I/O Command Sets that may be simultaneously used. The I/O Command Set Profile Feature value indicates the index of the I/O Command Set Combination that is currently selected (refer to section 5.1.25.1.17). I/O Command Set Combination 0 has an index value of 0h, I/O Command Set Combination 1 has an index value of 1h, and so on. Only I/O Command Sets that have a bit set to '1' in the I/O Command Set Vector of the I/O Command Set Combination selected by the I/O Command Set Profile Feature value may be used. All other I/O Command Sets are treated as unsupported I/O Command Sets.

**Figure 327: Identify I/O Command Set Data Structure**

Bytes	Description
7:0	<b>I/O Command Set Combination 0 (IOCSC0):</b> This field contains an I/O Command Set Vector indicating the first I/O Command Set or combination of I/O Command Sets that are simultaneously supported. If only one I/O Command Set is supported, then this field has only one bit set.
15:8	<b>I/O Command Set Combination 1 (IOCSC1):</b> This field contains an I/O Command Set Vector indicating the second I/O Command Set or combination of I/O Command Sets that are simultaneously supported if a second I/O Command Set combination is supported; otherwise, this field is cleared to 0h.  If this field is cleared to 0h, then no further I/O Command Set combinations are supported and subsequent I/O Command Set Combinations shall have a value of 0h.



**Figure 327: Identify I/O Command Set Data Structure**

Bytes	Description
...	...
4095:4088	<b>I/O Command Set Combination 511 (IOCSC511):</b> This field contains an I/O Command Set Vector indicating the 511th I/O Command Set or combination of I/O Command Sets that are simultaneously supported if a 511 <sup>th</sup> I/O Command Set combination is supported; otherwise, this field is cleared to 0h.

**Figure 328: I/O Command Set Vector**

Bits	Description
63:5	Reserved
4	<b>Computational Programs Namespace Command Set (CPNCS):</b> This bit is set to '1' if the Computational Programs Command Set is selected. This bit is cleared to '0' if the Zoned Namespace Command Set is not selected.
3	<b>Subsystem Local Memory Command Set (SLMCS):</b> This bit is set to '1' if the Subsystem Local Memory Command Set is selected. This bit is cleared to '0' if the Subsystem Local Memory Command Set is not selected.
2	<b>Zoned Namespace Command Set (ZNSCS):</b> This bit is set to '1' if the Zoned Namespace Command Set is selected. This bit is cleared to '0' if the Zoned Namespace Command Set is not selected.
1	<b>Key Value Command Set (KVCS):</b> This bit is set to '1' if the Key Value Command Set is selected. This bit is cleared to '0' if the Key Value Command Set is not selected.
0	<b>NVM Command Set (NVMCS):</b> This bit is set to '1' if the NVM Command Set is selected. This bit is cleared to '0' if the NVM Command Set is not selected.

**5.1.13.2.20 I/O Command Set Independent Identify Namespace data structure for an Allocated Namespace ID (CNS 1Fh)**

An I/O Command Set Independent Identify Namespace data structure (refer to section 5.1.13.2.8) is returned to the host for the namespace specified in the Namespace Identifier (NSID) field if it is an allocated NSID. If the specified NSID is an unallocated NSID, then the controller returns a zero filled data structure.

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

**5.1.13.2.21 Supported Controller State Formats (CNS 20h)**

A Supported Controller State Formats data structure (refer to Figure 329) is returned to the host identifying the supported NVMe Controller State data structures (refer to Figure 330) and a list of UUIDs that identifies the vendor specific controller state information.

A host uses an index into each list to identify the format and information returned by a Migration Receive command specifying the Get Controller State management operation (refer to section 5.1.16.1.1).

A host uses an index into each list to identify the format and information contained in a Migration Send command specifying the Set Controller State management operation of the Migration Send command (refer to section 5.1.17.1.3).

**Figure 329: Supported Controller State Formats Data Structure**

Bytes	Description
<b>Header</b>	
0	<b>Number of Versions (NV):</b> This field indicates the number of entries in the NVMe Controller State Version list. A value of 0h indicates that no entries exist in the NVMe Controller State Version list.
1	<b>Number of UUIDs (NUUID):</b> This field indicates the number of entries in the Vendor Specific Controller State UUID Supported list. A value of 0h indicates that no entries exist in the Vendor Specific Controller State UUID Supported List.

**Figure 329: Supported Controller State Formats Data Structure**

Bytes	Description
<b>NVMe Controller State Version list</b>	
3:2	<b>NVMe Controller State Data Structure Version 1:</b> This field contains a version of the NVMe Controller State data structure (refer to Version field in Figure 359) that is supported by this controller and is associated with index 1h of this list, if any. This field is the first entry of the NVMe Controller State Version list.
5:4	<b>NVMe Controller State Data Structure Version 2:</b> This field contains a version of the NVMe Controller State data structure that is supported by this controller and is associated with index 2h of this list, if any. This field is the second entry of the NVMe Controller State Version list.
...	
$(NV*2)+1:(NV*2)$	<b>NVMe Controller State Data Structure Version NV:</b> This field contains a version of the NVMe Controller State data structure that is supported by this controller and is associated with index NV of this list, if any. This field is the last entry of the NVMe Controller State Version list.
<b>Vendor Specific Controller State UUID Supported List</b>	
$((NV+1)*2)+15:$ $((NV+1)*2)$	<b>Vendor Specific Controller State UUID 1:</b> This field contains a 128-bit Universally Unique Identifier (UUID), as specified in RFC 9562 (refer to section 4.5.6), that identifies the format of the Vendor Specific Data field the Controller State data structure (refer to Figure 358) that is supported by this controller and is associated with index 1h of this list, if any. This field is the first entry of the Vendor Specific Controller State UUID Supported list.  The value in this field and the associated format of the Vendor Specific Data field the Controller State data structure is out of scope for this specification.
$((NV+1)*2)+31 :$ $((NV+1)*2)+16$	<b>Vendor Specific Controller State UUID 2:</b> This field contains a 128-bit Universally Unique Identifier (UUID), as specified in RFC 9562, that identifies the format of the Vendor Specific Data field the Controller State data structure that is supported by this controller and is associated with index 2h of this list, if any. This field is the second entry of the Vendor Specific Controller State UUID Supported list.  The value in this field and the associated format of the Vendor Specific Data field the Controller State data structure is out of scope for this specification.
...	
$((NUUID)*16)+$ $((NV+1)*2)-1 :$ $((NUUID-1)*16)+$ $((NV+1)*2)$	<b>Vendor Specific Controller State UUID NUUID:</b> This field contains a 128-bit Universally Unique Identifier (UUID), as specified in RFC 9562, that identifies the format of the Vendor Specific Data field the Controller State data structure that is supported by this controller and is associated with index NUUID of this list, if any. This field is the last entry of the Vendor Specific Controller State UUID Supported list.  The value in this field and the associated format of the Vendor Specific Data field the Controller State data structure is out of scope for this specification.

### 5.1.13.3 Memory-Based Transport Identify Data Structures (PCIe)

This section describes Identify data structures that are specific to the Memory-based transport model.

#### 5.1.13.3.1 Primary Controller Capabilities data structure (CNS 14h)

The Primary Controller Capabilities Structure (refer to Figure 330) is returned to the host for the primary controller specified.

**Figure 330: Identify – Primary Controller Capabilities Structure**

Bytes	Description
01:00	<b>Controller Identifier (CNTLID):</b> This field indicates the Controller Identifier of the primary controller.
03:02	<b>Port Identifier (PORTID):</b> This field indicates the Port Identifier of the NVM subsystem port associated with the primary controller. The Port Identifier for a PCI Express Port shall be unique within the NVM subsystem.  If the NVM subsystem supports an NVMe-MI Management Endpoint on this PCIe port, then this field shall contain the same value as the Port Identifier field in the Controller Information Data Structure (refer to the NVM Express Management Interface Specification) for this primary controller.

**Figure 330: Identify – Primary Controller Capabilities Structure**

Bytes	Description								
04	<b>Controller Resource Types (CRT):</b> This field indicates the controller resources types supported. If a primary controller supports a controller resource type, then all associated secondary controllers shall support that controller resource type.								
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:2</td> <td>Reserved</td> </tr> <tr> <td>1</td> <td><b>VI Resources Support (VIRS):</b> If this bit is set to '1', then VI Resources are supported. If this bit is cleared to '0', then VI Resources are not supported. Refer to section 8.2.6.2.</td> </tr> <tr> <td>0</td> <td><b>VQ Resources Support (VQRS):</b> If this bit is set to '1', then VQ Resources are supported. If this bit is cleared to '0', then VQ Resources are not supported. Refer to section 8.2.6.1.</td> </tr> </tbody> </table>	Bits	Description	7:2	Reserved	1	<b>VI Resources Support (VIRS):</b> If this bit is set to '1', then VI Resources are supported. If this bit is cleared to '0', then VI Resources are not supported. Refer to section 8.2.6.2.	0	<b>VQ Resources Support (VQRS):</b> If this bit is set to '1', then VQ Resources are supported. If this bit is cleared to '0', then VQ Resources are not supported. Refer to section 8.2.6.1.
	Bits	Description							
	7:2	Reserved							
1	<b>VI Resources Support (VIRS):</b> If this bit is set to '1', then VI Resources are supported. If this bit is cleared to '0', then VI Resources are not supported. Refer to section 8.2.6.2.								
0	<b>VQ Resources Support (VQRS):</b> If this bit is set to '1', then VQ Resources are supported. If this bit is cleared to '0', then VQ Resources are not supported. Refer to section 8.2.6.1.								
31:05	Reserved								
35:32	<b>VQ Resources Flexible Total (VQFRT):</b> This field indicates the total number of VQ Flexible Resources for the primary and its secondary controllers.								
39:36	<b>VQ Resources Flexible Assigned (VQRFPA):</b> This field indicates the total number of VQ Flexible Resources Assigned to the associated secondary controllers.								
41:40	<b>VQ Resources Flexible Allocated to Primary (VQRFAP):</b> This field indicates the total number of VQ Flexible Resources currently allocated to the primary controller. This value may change after a Controller Level Reset other than a Controller Reset, if a new value was set using the Virtualization Management command. The default value of this field is implementation specific.								
43:42	<b>VQ Resources Private Total (VQPRT):</b> This field indicates the total number of VQ Private Resources for the primary controller.								
45:44	<b>VQ Resources Flexible Secondary Maximum (VQFRSM):</b> This field indicates the maximum number of VQ Flexible Resources that may be assigned to a secondary controller.								
47:46	<b>VQ Flexible Resource Preferred Granularity (VQGRAN):</b> This field indicates the preferred granularity of assigning and removing VQ Flexible Resources. Assigning and removing VQ Resources in this granularity minimizes any wasted internal implementation resources.								
63:48	Reserved								
67:64	<b>VI Resources Flexible Total (VIFRT):</b> This field indicates the total number of VI Flexible Resources for the primary and its secondary controllers.								
71:68	<b>VI Resources Flexible Assigned (VIRFA):</b> This field indicates the total number of VI Flexible Resources Assigned to the associated secondary controllers.								
73:72	<b>VI Resources Flexible Allocated to Primary (VIRFAP):</b> This field indicates the total number of VI Flexible Resources currently allocated to the primary controller. This value may change after a Controller Level Reset other than a Controller Reset, if a new value was set using the Virtualization Management command. The default value of this field is implementation specific.								
75:74	<b>VI Resources Private Total (VIPRT):</b> This field indicates the total number of VI Private Resources for the primary controller.								
77:76	<b>VI Resources Flexible Secondary Maximum (VIFRSM):</b> This field indicates the maximum number of VI Flexible Resources that may be assigned to a secondary controller.								
79:78	<b>VI Flexible Resource Preferred Granularity (VIGRAN):</b> This field indicates the preferred granularity of assigning and removing VI Flexible Resources. Assigning and removing VI Resources in this granularity minimizes any wasted internal implementation resources.								
4095:80	Reserved								

### 5.1.13.3.2 Secondary Controller list (CNS 15h)

A Secondary Controller List (refer to Figure 331) is returned to the host for up to 127 secondary controllers associated with the primary controller processing this command. The list contains entries for controller identifiers greater than or equal to the value specified in the Controller Identifier (CDW10.CNTID) field.

All secondary controllers are represented, including those that are in an Offline state due to SR-IOV configuration settings (e.g., VF Enable is cleared to '0' or NumVFs specifies a value that does not enable the associated secondary controller).

**Figure 331: Secondary Controller List**

Bytes	Description
00	<b>Number of Entries (NUMENT):</b> This field indicates the number of Secondary Controller Entries in the list. There are up to 127 entries in the list. A value of 0h indicates there are no entries in the list.
31:01	Reserved
Secondary Controller Entry List	
63:32	<b>SC Entry 0:</b> This field contains the first Secondary Controller Entry in the list, if present.
95:64	<b>SC Entry 1:</b> This field contains the second Secondary Controller Entry in the list, if present.
...	...
((NUMENT-1)*32+63): ((NUMENT-1)*32+32)	<b>SC Entry NUMENT-1:</b> This field contains the last Secondary Controller Entry in the list, if present.

**Figure 332: Secondary Controller Entry**

Bytes	Description						
01:00	<b>Secondary Controller Identifier (SCID):</b> This field indicates the Controller Identifier of the secondary controller described by this entry.						
03:02	<b>Primary Controller Identifier (PCID):</b> This field indicates the Controller Identifier of the associated primary controller.						
04	<b>Secondary Controller State (SCS):</b> This field indicates the state of the secondary controller.						
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td><b>Online State (OLS):</b> If this bit is set to '1', then the controller is in the Online state. If this bit is cleared to '0', then the controller is in the Offline state.</td> </tr> </tbody> </table>	Bits	Description	7:1	Reserved	0	<b>Online State (OLS):</b> If this bit is set to '1', then the controller is in the Online state. If this bit is cleared to '0', then the controller is in the Offline state.
	Bits	Description					
7:1	Reserved						
0	<b>Online State (OLS):</b> If this bit is set to '1', then the controller is in the Online state. If this bit is cleared to '0', then the controller is in the Offline state.						
07:05	Reserved						
09:08	<b>Virtual Function Number (VFN):</b> If the secondary controller is an SR-IOV VF, this field indicates its VF Number, where VF Number > 0, and VF Number is no larger than the total number of VFs indicated by the TotalVFs register (refer to Single Root I/O Virtualization and Sharing Specification) in the PF's SR-IOV Extended Capability structure. If the secondary controller is not an SR-IOV VF, then this field is cleared to 0h.						
11:10	<b>Number of VQ Flexible Resources Assigned (NVQ):</b> This field indicates the number of VQ Flexible Resources currently assigned to the indicated secondary controller.						
13:12	<b>Number of VI Flexible Resources Assigned (NVI):</b> This field indicates the number of VI Flexible Resources currently assigned to the indicated secondary controller.						
31:14	Reserved						

#### 5.1.13.4 Message-Based Transport Identify Data Structures (Fabrics)

This section describes Identify data structures that are specific to the Message-based transport model.

##### 5.1.13.4.1 Get Underlying Namespace List (CNS 1Dh)

An Underlying Namespace List data structure (refer to Figure 333) is returned containing a list of all Underlying Namespaces spanning all NVM subsystems that are accessible through either a virtual function or a physical function.

**Figure 333: Underlying Namespace List Data Structure**

Bytes	Description
07:00	<b>Generation Counter (GENCTR):</b> This field indicates the version of Underlying Namespace information. This field shall be cleared to 0h as a result of NVM Subsystem Reset of the Underlying NVM Subsystem. For each change in the Underlying Namespace List, this counter is incremented by one. If the value of this field is FFFFFFFF_FFFFFFFFh, then the field shall be cleared to 0h when incremented (i.e., rolls over to 0h).
15:08	<b>Number Entries (NUMENT):</b> Number of Underlying Namespace Entries.

**Figure 333: Underlying Namespace List Data Structure**

Bytes	Description
<b>Underlying Namespace List</b>	
335:16	<b>Underlying Namespace Entry 1:</b> The first Underlying Namespace Entry data structure (refer to Figure 334) in the Underlying Namespace List. If any.
655:336	<b>Underlying Namespace Entry 2:</b> The second Underlying Namespace Entry data structure (refer to Figure 334) in the Underlying Namespace List. If any.
...	...
320*NUMENT+15: 320*(NUMENT-1)+16	<b>Underlying Namespace Entry NUMENT:</b> The last Underlying Namespace Entry data structure (refer to Figure 334) in the Underlying Namespace List. If any.

**Figure 334: Underlying Namespace Entry Data Structure**

Bytes	Description
255:00	<b>Underlying NVM Subsystem NQN (UNSNQN):</b> The Underlying NVM Subsystem NQN which contains the Underlying Namespace.
259:256	<b>Namespace Identifier (NSID):</b> This field indicates the NSID value of the namespace in the Underlying NVM Subsystem identified by the Underlying NVM Subsystem NQN field. If the value in this field is not an NSID reported in the Underlying Namespace List (refer to 5.1.13.4.1), then the command shall be aborted with a status code set to Invalid Field in Command.
261:260	<b>Controller ID (CNTLID):</b> Contains the Underlying NVM Subsystem unique controller identifier associated with the Underlying Namespace. Refer to section 5.1.13.2.1.
319:262	Reserved

#### 5.1.13.4.2 Get Ports List (CNS 1Eh)

A Ports List data structure (refer to Figure 335) is returned containing a list of Underlying Ports that may be used to export NVMe over Fabrics NVM subsystems.

**Figure 335: Ports List Data Structure**

Bytes	Description
07:00	<b>Generation Counter (GENCTR):</b> This field indicates the version of Port information. This field shall be cleared to 0h as a result of NVM Subsystem Reset of the Underlying NVM Subsystem. For each change in the Ports List, this counter is incremented by one. If the value of this field is FFFFFFFF_FFFFFFFFh, then the field shall be cleared to 0h when incremented (i.e., rolls over to 0h).
15:08	<b>Number Entries (NUMENT):</b> Number of Fabrics Transport Entries in the Ports List.
<b>Ports List</b>	
591:16	<b>Fabrics Transport Entry 1:</b> The first Underlying Fabrics Transport Entry data structure (refer to Figure 336) in the Ports List, if any.
1167:592	<b>Fabrics Transport Entry 2:</b> The second Underlying Fabrics Transport Entry data structure (refer to Figure 336) in the Ports List, if any..
...	...
576*NUMENT+15: 576*(NUMENT-1)+16	<b>Fabrics Transport Entry NUMENT:</b> The last Underlying Fabrics Transport Entry data structure (refer to Figure 336) in the Ports List, if any..

**Figure 336: Underlying Fabrics Transport Entry Data Structure**

Bytes	Description
255:00	<b>Transport Address (TRADDR):</b> This field indicates the address of the NVM subsystem that may be used for a Connect command as an ASCII string. The Transport Address Family field describes the reference for parsing this field. Refer to section 1.4 for ASCII string requirements. For the definition of this field, refer to the appropriate NVMe Transport specification.
511:256	<b>Transport Specific Address Subtype (TSAS):</b> This field indicates NVMe Transport specific information about the address. For the definition of this field, refer to the appropriate NVMe Transport specification.

**Figure 336: Underlying Fabrics Transport Entry Data Structure**

Bytes	Description
513:512	<b>Port ID of the Underlying Port (PIDUP):</b> Refer to the PORTID field in Figure 294.
514	<b>Transport Type (TRTYPE):</b> Specifies the NVMe Transport Type. Refer to Figure 294.
515	<b>Transport Address Family (ADRFAM):</b> Specifies the Address Family. Refer Figure 294.
516	<b>Transport Requirements (TREQ):</b> This field indicates requirements for the NVMe Transport. Refer to Figure 294.
575:517	Reserved

### 5.1.13.5 Command Completion

Upon completion of the Identify command, the controller posts a completion queue entry to the Admin Completion Queue.

### 5.1.14 Keep Alive command

The Keep Alive command and the Keep Alive Timer feature (refer to section 5.1.25.1.8) are used by the Keep Alive capability (refer to section 3.9). The controller should process the Keep Alive command as soon as the command is fetched. If the Keep Alive Timer is active (refer to section 3.9.2), then upon processing a Keep Alive command, the controller restarts the Keep Alive Timer.

All command specific fields are reserved.

#### 5.1.14.1 Command Completion

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

### 5.1.15 Lockdown command

The Lockdown command is used to control the Command and Feature Lockdown capability (refer to section 8.1.5) which configures the prohibition or allowance of execution of the specified command or Set Features command targeting a specific Feature Identifier.

After a successful completion of a Lockdown command prohibiting a command or Feature Identifier, all controllers, if applicable, and all management endpoints, if applicable, in the NVM subsystem behave as described in section 8.1.5.

The Lockdown command uses Command Dword 10 (refer to Figure 337) and Command Dword 14 (refer to Figure 338). All other command specific fields are reserved.

**Figure 337: Lockdown – Command Dword 10**

Bits	Description		
31:16	Reserved		
15:08	<b>Opcode or Feature Identifier (OFI):</b> This field specifies the command opcode or Set Features Feature Identifier identified by the Scope field.		
07	Reserved		
06:05	<b>Interface (IFC):</b> This field identifies the interfaces affected by this command. The actions of this command apply if a command is received on the specified interfaces.		
		Value	Affected Interfaces
		00b	Admin Submission Queue
		01b	Admin Submission Queue and out-of-band on a Management Endpoint
		10b	Out-of-band on a Management Endpoint
11b	Reserved		

**Figure 337: Lockdown – Command Dword 10**

Bits	Description	
04	<b>Prohibit (PRHBT):</b> This bit specifies whether to prohibit or allow the command opcode or Set Features Feature Identifier specified by this command. If this bit is set to '1', then this command prohibits the execution of the command based on other fields specified in Dword 10. If this bit is cleared to '0', then this command allows the execution of the command based on other fields specified in Dword 10.	
03:00	<b>Scope (SCP):</b> This field specifies the contents of the Opcode or Feature Identifier field.	
	<b>Value</b> <b>Opcode or Feature Identifier Definition</b>	
	0h	Admin command opcode
	1h	Reserved
	2h	A Set Features Feature Identifier
	3h	Management Interface Command Set opcode (refer to the NVM Express Management Interface Specification)
	4h	PCIe Command Set opcode (refer to the NVM Express Management Interface Specification)
5h-Fh	Reserved	

If the controller supports selection of a UUID:

- a) by the Lockdown command; and
- b) by the Set Features command (refer to section 5.1.25 and section 8.1.28) and for the vendor specific Feature Identifier specified by the Opcode or Feature Identifier field, if the Scope field is set to 2h,

then Command Dword 14 (refer to Figure 338) is used to specify a UUID Index value.

If the controller does not support selection of a UUID:

- a) by the Lockdown command;
- b) by the Set Features command; or
- c) for the vendor specific feature identifier specified by the Opcode or Feature Identifier field, if the Scope field is set to 2h,

then Command Dword 14 does not specify a UUID Index value. If the Scope field is not set to 2h, then UUID Index field is ignored.

**Figure 338: Lockdown – Command Dword 14**

Bits	Description
31:07	Reserved
06:00	<b>UUID Index (UIDX):</b> Refer to Figure 658.

If a controller processes this command specifying a command opcode or Feature Identifier that is not supported as being prohibitable, then the command shall be aborted with a status code of Prohibition of Command Execution Not Supported.

If a controller processes this command with the Interface field set to 01b or 10b and the NVM subsystem does not contain a Management Endpoint, then the command shall be aborted with a status code of Invalid Field in Command.

If a controller processes this command with the Interface field set to 00b or 01b and the Scope field is set to 4h, then the command shall be aborted with a status code of Invalid Field in Command.

It is not an error to attempt to prohibit a command or Feature Identifier that is already prohibited from execution or allow a command or Feature Identifier that is already allowed to be executed.

#### 5.1.15.1 Command Completion

Upon completion of the Lockdown command, the controller posts a completion queue entry to the Admin Completion Queue.

Lockdown command specific status values are defined in Figure 339.

**Figure 339: Lockdown – Command Specific Status Values**

Value	Description
28h	<b>Prohibition of Command Execution Not Supported:</b> The command was aborted due to the specified opcode or Feature Identifier not supporting being prohibited from execution by the command.

### 5.1.16 Migration Receive command

The Migration Receive command is used to obtain information from the controller processing the command that the host may use to manage a migratable controller (refer to section 8.1.12).

The Migration Receive command uses the Data Pointer field, Command Dword 10 field, Command Dword 12 field, Command Dword 13 field, Command Dword 14 field, and Command Dword 15 field. The use of the Command Dword 11 field is specific to the management operation specified by the Select field. All other command specific fields are reserved.

The Select field defined in Figure 340 specifies the management operation to be performed. Refer to section 5.1.16.1 for a description of each management operation.

**Figure 340: Migration Receive – Command Dword 10**

Bits	Description												
31:16	<b>Management Operation Specific (MOS):</b> The definition of this field is specific to a management operation (refer to the Select field). If a management operation does not define the use of this field, then this field is reserved.												
15:08	Reserved												
07:00	<b>Select (SEL):</b> This field specifies the type of management operation to perform.												
	<table border="1"> <thead> <tr> <th>Value</th> <th>M/O<sup>1</sup></th> <th>Management Operation</th> <th>Reference Section</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>M</td> <td>Get Controller State</td> <td>5.1.16.1.1</td> </tr> <tr> <td>1h to FFh</td> <td></td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Value	M/O <sup>1</sup>	Management Operation	Reference Section	0h	M	Get Controller State	5.1.16.1.1	1h to FFh		Reserved	
	Value	M/O <sup>1</sup>	Management Operation	Reference Section									
	0h	M	Get Controller State	5.1.16.1.1									
1h to FFh		Reserved											
Notes:													
1. O/M definition: O = Optional, M = Mandatory													

**Figure 341: Migration Receive – Command Dword 12**

Bits	Description
31:00	<b>Offset Lower (OL):</b> This field specifies the least-significant 32-bits of the offset, in bytes, within the data available to be returned and specifies the starting point for that data for what is actually returned to the host.  The offset is required to be dword aligned and if bits 1:0 are not cleared to 00b, then the controller shall abort the command with a status code of Invalid Field in Command.  If the host specifies an offset (i.e., OL field and OU field) that is greater than the size of the returned data requested (e.g., the returned data contains 100 bytes is requested starting at offset 200), then the controller shall abort the command with a status of Invalid Field in Command.

**Figure 342: Migration Receive – Command Dword 13**

Bits	Description
31:00	<b>Offset Upper (OU):</b> This field specifies the most-significant 32-bits of the offset, in bytes, within the data available to be returned and specifies the starting point for that data for what is actually returned to the host. Refer to the Offset Lower field definition in Figure 340.

If the controller supports selection of a UUID by the Migration Receive command (refer to Figure 210 and section 8.1.28), then Command Dword 14 is used to specify a UUID Index value (refer to Figure 343).



**Figure 343: Migration Receive – Command Dword 14**

Bits	Description
31:07	Reserved
06:00	<b>UUID Index (UIDX):</b> Refer to Figure 658.

**Figure 344: Migration Receive – Command Dword 15**

Bits	Description
31:00	<b>Number of Dwords (NUMDL):</b> This field specifies the number of dwords to return. If the host specifies a value that is greater than the number of dwords returned by the specified management operation of the command, then the controller returns the number of dwords defined by the specified management operation of the command with undefined results for the remaining dwords. This is a 0's based value.

### 5.1.16.1 Migration Receive Management Operations

#### 5.1.16.1.1 Get Controller State (Management Operation 0h)

The Get Controller State management operation of the Migration Receive command allows the host to retrieve from the controller processing the command state data for the controller specified in the Controller Identifier field in the Management Operation Specific field (refer to Figure 345). The data returned in the data buffer is the Controller State data structure as defined by Figure 358.

If the controller specified in the Controller Identifier field in the Management Operation Specific field (i.e., the specified controller):

- is suspended (refer to section 5.1.17.1.1); and
- remains suspended during the processing of the Migration Receive command,

then:

- the specified controller has stopped processing commands; and
- if:
  - a Controller Level Reset has not occurred on the specified controller during the processing of the Migration Receive command; or
  - the host does not modify controller properties on the specified controller during the processing of the Migration Receive command,

then the returned Controller State data structure is consistent between each field reported in that data structure.

If the specified controller:

- is not suspended during the processing of the Migration Receive command; or
- has a Controller Level Reset during the processing of the Migration Receive command,

then the specified controller state may be changing during the processing of the Migration Receive command and:

- each field in the reported Controller State data structure contains the value that reflects the state of the specified controller at the time the value was obtained by the controller processing the Migration Receive command; and
- state of the specified controller may be at a different state when a value is obtained between different fields (i.e., the returned Controller State data structure may or may not be internally consistent between each field).

The Get Controller State management operation uses the Management Operation Specific field as defined in Figure 345 and the Command Dword 11 field as defined in Figure 346.

**Figure 345: Get Controller State Management Operation – Management Operation Specific**

Bits	Description
15:08	Reserved
7:0	<p><b>Controller State Version Index (CSVI):</b> A non-zero value in this field specifies the index to a specific entry in the Controller State Version Supported list of the Supported Controller State Formats data structure (refer to section 5.1.13.2.21). The contents of that entry specify the format of the Controller State field in the returned data.</p> <p>If this field is cleared to 0h, then no Controller State field is returned in the returned data.</p>

The Get Controller State management operation of the Migration Receive command uses Command Dword 11 as defined in Figure 346.

**Figure 346: Get Controller State Management Operation – Command Dword 11**

Bits	Description
31:24	<p><b>Controller State UUID Index Parameter (CSUIDXP):</b> This field is vendor specific.</p> <p>If the Controller State UUID Index field is cleared to 0h, then the controller shall ignore this field.</p>
23:16	<p><b>Controller State UUID Index (CSUUIDI):</b> A non-zero value in this field specifies the index to a specific entry in the Vendor Specific Controller State UUID Supported list of the Supported Controller State Formats data structure (refer to section 5.1.13.2.21). The contents of that entry specify the format of the Vendor Specific field in the returned data.</p> <p>If this field is cleared to 0h, then no Vendor Specific field is returned in the returned data.</p>
15:00	<p><b>Controller Identifier (CNTLID):</b> This field specifies the identifier of the controller on which the management operation is performed.</p>

If the CSVI field (refer to Figure 346) specifies a non-zero index that is not defined by the Supported Controller State Formats data structure (refer to Figure 329), then the controller shall abort the command with a status code of Invalid Field in Command.

If the CSVI field is cleared to 0h, then the Controller State Size field in the returned data shall be cleared to 0h.

If the CSUUIDI field specifies a non-zero index that is not defined by the Supported Controller State Formats data structure, then the controller shall abort the command with a status code of Invalid Field in Command.

If the CSUUIDI field is cleared to 0h, then the Vendor Specific Size field in the returned data shall be cleared to 0h.

If a status code of Successful Completion is returned, then the completion queue entry Dword 0 contains additional information about the command as defined in Figure 347.

**Figure 347: Get Controller State Management Operation – Completion Queue Entry Dword 0**

Bits	Description
31:01	Reserved
00	<p><b>Controller Suspended (CSUP):</b> If this bit is set to '1', then the controller associated with the returned data structure was suspended for the entire duration of the processing of this command. If this bit is cleared to '0', then the controller associated with this data structure was not suspended for the entire duration of the processing of this command.</p>

#### 5.1.16.1.2 Command Completion

The Upon completion of the Migration Receive command, the controller posts a completion queue entry to the Admin Completion Queue indicating the status for the command. Section 5.1.16.1 describes completion details for each management operation.

Migration Receive command specific status values (i.e., SCT field set to 1h) are shown in Figure 348.

**Figure 348: Migration Receive – Command Specific Status Values**

Value	Definition
1Fh	<b>Invalid Controller Identifier:</b> The specified controller is not in a condition to set the controller state.

### 5.1.17 Migration Send command

The Migration Send command is used to manage the migration of a controller (refer to section 8.1.12).

The Migration Send command uses the Command Dword 10 and Command Dword 14. The use of the Data Pointer field, Command Dword 11 field, Command Dword 12 field, Command Dword 13 field, and Command Dword 15 field is specific to the management operation specified by the Select field. All other command specific fields are reserved.

The Select field defined in Figure 349 specifies the management operation to be performed. Refer to section 5.1.17.1 for a description of each management operation.

**Figure 349: Migration Send – Command Dword 10**

Bits	Description																				
31:16	<b>Management Operation Specific (MOS):</b> The definition of this field is specific to a management operation (refer to the Select field). If a management operation does not define the use of this field, then this field is reserved.																				
15:08	Reserved																				
07:00	<b>Select (SEL):</b> This field specifies the type of management operation to perform. <table border="1" data-bbox="480 921 1230 1087"> <thead> <tr> <th>Value</th> <th>M/O<sup>1</sup></th> <th>Management Operation</th> <th>Reference Section</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>M</td> <td>Suspend</td> <td>5.1.17.1.1</td> </tr> <tr> <td>1h</td> <td>M</td> <td>Resume</td> <td>5.1.17.1.2</td> </tr> <tr> <td>2h</td> <td>M</td> <td>Set Controller State</td> <td>5.1.17.1.3</td> </tr> <tr> <td>3h to FFh</td> <td></td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Value	M/O <sup>1</sup>	Management Operation	Reference Section	0h	M	Suspend	5.1.17.1.1	1h	M	Resume	5.1.17.1.2	2h	M	Set Controller State	5.1.17.1.3	3h to FFh		Reserved	
		Value	M/O <sup>1</sup>	Management Operation	Reference Section																
		0h	M	Suspend	5.1.17.1.1																
		1h	M	Resume	5.1.17.1.2																
		2h	M	Set Controller State	5.1.17.1.3																
3h to FFh		Reserved																			
Notes:																					
1. O/M definition: O = Optional, M = Mandatory																					

If the controller supports selection of a UUID by the Migration Send command (refer to Figure 210 and section 8.1.28), then Command Dword 14 is used to specify a UUID Index value (refer to Figure 350).

**Figure 350: Migration Send – Command Dword 14**

Bits	Description
31:07	Reserved
06:00	<b>UUID Index (UIDX):</b> Refer to Figure 658.

### 5.1.17.1 Migration Send Management Operations

#### 5.1.17.1.1 Suspend (Management Operation 0h)

The Suspend management operation of the Migration Send command allows a host to specify the type of suspend to be done on the specified controller.

The Suspend management operation uses the Command Dword 11 field as defined in Figure 351 and does not use the Management Operation Specific field.

**Figure 351: Suspend – Command Dword 11**

Bits	Description								
31	<b>Delete User Data Migration Queue (DUDMQ):</b> If this bit is set to '1' and the command completes successfully, then the User Data Migration Queue associated with the controller specified in the CNTLID field, if any, is deleted (refer to the applicable I/O Command Set specification). If this bit is cleared to '0', then the User Data Migration Queue associated the controller specified in the CNTLID field is retained after processing the command, if any.								
30:24	Reserved								
23:16	<b>Suspend Type (STYPE):</b> This field specifies the type of suspend.								
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td><b>Suspend Notification:</b> The specified controller is going to be suspended in the future with a subsequent Migration Send command specifying a Suspend management operation and a non-zero value in the Suspend Type field.</td> </tr> <tr> <td>1h</td> <td><b>Suspend:</b> Suspend the controller specified in the CNTLID field as defined in this section.</td> </tr> <tr> <td>All Others</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	<b>Suspend Notification:</b> The specified controller is going to be suspended in the future with a subsequent Migration Send command specifying a Suspend management operation and a non-zero value in the Suspend Type field.	1h	<b>Suspend:</b> Suspend the controller specified in the CNTLID field as defined in this section.	All Others	Reserved
	Value	Definition							
	0h	<b>Suspend Notification:</b> The specified controller is going to be suspended in the future with a subsequent Migration Send command specifying a Suspend management operation and a non-zero value in the Suspend Type field.							
1h	<b>Suspend:</b> Suspend the controller specified in the CNTLID field as defined in this section.								
All Others	Reserved								
15:00	<b>Controller Identifier (CNTLID):</b> This field specifies the identifier of the controller to which the Suspend management operation is performed.								

A Suspend Type field value of 0h (i.e., Suspend Notification) allows a host to notify a controller that the controller specified by the Controller Identifier field is going to be requested to suspend with a subsequent Migration Send command specifying a Suspend management operation and a non-zero value in the Suspend Type field.

A Suspend Type field value of 1h (i.e., Suspend) specifies that the controller specified in the Controller Identifier field (refer to Figure 351) is to be suspended. That controller shall perform the following actions and then stop initiating transport transactions:

1. Stop fetching commands until the specified controller is no longer suspended (refer to section 8.1.12).
2. Complete the processing of all previously fetched commands, if any, except any outstanding Asynchronous Event Request commands.

If:

- the command causes the specified controller to be suspended; and
- there is a User Data Migration Queue associated with that specified controller,

then:

- If the DUDMQ bit is set to '1', then the controller processing the command before posting the completion queue entry to the command shall delete that User Data Migration Queue; and
- after the specified controller completes the processing of all previously fetched commands, if any, except any outstanding Asynchronous Event Request commands, the controller processing the Migration Send command shall, at a vendor specific amount of time, posts an entry in that User Data Migration Queue that indicates the specified controller is suspended as specified by the applicable I/O Command Set specification.

It is not an error if a suspend operation occurs on a controller that is already suspended with the same Suspend Type value.

If the result of this command causes the specified controller to be suspended, then the specified controller remains suspended until:

- a Migration Send command specifying the Resume management operation (refer to section 5.1.17.1.2) on the specified controller is completed successfully; or
- a Controller Level Reset occurs on the controller processing this command.

If a controller is suspended, then:

- accesses to controller properties for that suspended controller shall behave normally except a host modification of the CC property that sets CC.EN bit from '0' to '1' (i.e., enabling that suspended controller) has implementation specific behavior (i.e., the suspended controller may or may not set the CSTS.RDY bit). A host should avoid modifications to controller properties, CMB, and PMR to a controller that is suspended;
- the Controller Memory Buffer (CMB), if supported, shall behave as defined in section 8.2.1 (i.e., CMB on a suspended controller continues to function normally); and
- the Persistent Memory Region (PMR), if supported, shall behave as defined in section 8.2.4 (i.e., PMR on a suspended controller continues to function normally).

#### 5.1.17.1.2 Resume (Management Operation 1h)

The Resume management operation of the Migration Send command is a request for the controller processing the command to resume the operation of a different controller that is suspended (refer to section 5.1.17.1.1).

The Resume management operation uses the Command Dword 11 field as defined in Figure 352 and does not use the Management Operation Specific field.

**Figure 352: Resume – Command Dword 11**

Bits	Description
31:16	Reserved
15:00	<b>Controller Identifier (CNTLID):</b> This field specifies the identifier of the controller to resume operations.

The Resume management operation shall cause the controller specified by the Controller Identifier field (refer to Figure 351) to start fetching and processing commands.

If the specified controller is not suspended (refer to section 5.1.17.1.1), then the controller shall abort the command with a status code of Controller Not Suspended.

If the controller processing the command has received controller state for the specified controller to be resumed and that controller state has not been verified and committed (refer to section 5.1.17.1.3), then the controller shall abort the command with a status code of Command Sequence Error.

#### 5.1.17.1.3 Set Controller State (Management Operation 2h)

The Set Controller State management operation of the Migration Send command allows the host to set the state of the controller specified by the Controller Identifier (CNTLID) field in Command Dword 11 (refer to Figure 354).

The data buffer contains the Controller State data structure specified in Figure 358 which specifies the state to be set in the specified controller.

To set the controller state of the specified controller, the specified controller is required to be in one or more of these conditions:

- Suspended (refer to section 5.1.17.1.1);
- enabled (i.e., CC.EN bit is set to '1'); or
- offline, if that specified controller is a secondary controller.

If the specified controller is not in one of these conditions, then the controller processing the command shall abort the command with a status code of Invalid Controller Identifier.

The host may submit one or more Migration Send commands to transfer the controller state. If more than one Migration Send command is submitted to transfer the controller state, then:

- The host first submits a Migration Send command with:
  - the Select field set to Set Controller State (i.e., 2h);

- the Sequence Indicator (SEQIND) field (refer to Figure 353) set to 01b; and
- the Controller Identifier field set to the controller whose state is being set,

to specify to the controller that the command is the first of a sequence of Migration Send commands of controller state data being transferred.

- The host may submit Migration Send commands with:
  - the Select field set to Set Controller State (i.e., 2h);
  - the SEQIND field cleared to 00b; and
  - the Controller Identifier field set to the controller whose state is being set,

to specify to the controller that the command is not the last of the sequence of Migration Send commands of controller state data being transferred, if any.

- The host is required to wait for the completion queue entries to be posted for the previous submitted Migration Send commands in the sequence of Migration Send commands.
- Finally, the host submits the last Migration Send command with:
  - the Select field set to Set Controller State (i.e., 2h);
  - the SEQIND field set to 10b; and
  - the Controller Identifier field set to the controller whose state is being set,

to specify to the controller that this is the last command of the sequence of Migration Send commands of the controller state data being transferred. This last Migration Send command may or may not have the Number of Dwords (NUMD) field cleared to 0h.

If a sequence of Migration Send commands of the controller state data being transferred and the controller processes a Migration Send command with:

- the Select field set to Set Controller State (i.e., 2h);
- the Sequence Indicator (SEQIND) field (refer to Figure 353) set to 01b; and
- the Controller Identifier field set to the controller whose state is being set,

then, the command is specifying a new sequence of Migration Send commands of the controller state data is being transferred and the previous sequence of Migration Send commands of the controller state data is discarded.

If a sequence of Migration Send commands of the controller state data is not being transferred to the specified controller and the controller processes a Migration Send command with the Sequence Indicator (SEQIND) field (refer to Figure 353) not set to 01b, clearer to 0b; the controller shall abort that command with a status code of Command Sequence Error.

If a single Migration Send command is submitted to transfer the entire controller state, then the host submits a Migration Send command with the SEQIND field set to 11b.

**Figure 353: Set Controller State – Management Operation Specific Field**

Bits	Description										
15:02	Reserved										
01:00	<b>Sequence Indicator (SEQIND):</b> This field identified the sequences of this Migration Send command in relation to other Migration Send commands.										
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>This command is not the first or last of a sequence of two or more Migration Send commands with this management operation used to transfer the controller state from the host to the controller.</td> </tr> <tr> <td>01b</td> <td>This command is the first of a sequence of two or more Migration Send commands with this management operation used to transfer the controller state from the host to the controller.</td> </tr> <tr> <td>10b</td> <td>This Migration Send command is the last command of a sequence of two or more Migration Send commands with this management operation used to transfer the controller state from the host to the controller.</td> </tr> <tr> <td>11b</td> <td>This Migration Send command is the only command and contains the entire controller state for this management operation used to transfer the controller state from the host to the controller.</td> </tr> </tbody> </table>	Value	Definition	00b	This command is not the first or last of a sequence of two or more Migration Send commands with this management operation used to transfer the controller state from the host to the controller.	01b	This command is the first of a sequence of two or more Migration Send commands with this management operation used to transfer the controller state from the host to the controller.	10b	This Migration Send command is the last command of a sequence of two or more Migration Send commands with this management operation used to transfer the controller state from the host to the controller.	11b	This Migration Send command is the only command and contains the entire controller state for this management operation used to transfer the controller state from the host to the controller.
	Value	Definition									
	00b	This command is not the first or last of a sequence of two or more Migration Send commands with this management operation used to transfer the controller state from the host to the controller.									
	01b	This command is the first of a sequence of two or more Migration Send commands with this management operation used to transfer the controller state from the host to the controller.									
10b	This Migration Send command is the last command of a sequence of two or more Migration Send commands with this management operation used to transfer the controller state from the host to the controller.										
11b	This Migration Send command is the only command and contains the entire controller state for this management operation used to transfer the controller state from the host to the controller.										
00b	This command is not the first or last of a sequence of two or more Migration Send commands with this management operation used to transfer the controller state from the host to the controller.										
01b	This command is the first of a sequence of two or more Migration Send commands with this management operation used to transfer the controller state from the host to the controller.										
10b	This Migration Send command is the last command of a sequence of two or more Migration Send commands with this management operation used to transfer the controller state from the host to the controller.										
11b	This Migration Send command is the only command and contains the entire controller state for this management operation used to transfer the controller state from the host to the controller.										

**Figure 354: Set Controller State – Command Dword 11**

Bits	Description
31:24	<b>Controller State UUID Index (CSUUDI):</b> A non-zero value in this field specifies the index to a specific entry in the Vendor Specific Controller State UUID Supported list of the Supported Controller State Formats data structure (refer to section 5.1.13.2.21). The contents of that entry specify the format of the Vendor Specific field in the specified Controller State data structure.  If this field is cleared to 0h, then no Vendor Specific field is contained in the Controller State data structure.
23:16	<b>Controller State Version Index (CSVI):</b> A non-zero value in this field specifies the index to a specific entry in the NVMe Controller State Version list of the Supported Controller State Formats data structure (refer to section 5.1.13.2.21). The contents of that entry specify the format of the Controller State field in the specified Controller State data structure.  If this field is cleared to 0h, then no NVMe Controller State field is specified in the Controller State data structure.
15:00	<b>Controller Identifier (CNTLID):</b> This field specifies the identifier of the controller whose state is being set.

**Figure 355: Set Controller State – Command Dword 12**

Bits	Description
31:00	<b>Controller State Offset Lower (CSOL):</b> This field specifies the least-significant 32-bits of starting offset within the Controller State data structure that the host is setting.  The offset is required to be dword aligned and if bits 1:0 are not cleared to 00b, then the controller shall abort the command with a status code of Invalid Field in Command.

**Figure 356: Set Controller State – Command Dword 13**

Bits	Description
31:00	<b>Controller State Offset Upper (CSOU):</b> This field specifies the most-significant 32-bits of starting offset within the Controller State data that the host is setting. Refer to the CSOL field definition in Figure 355.

**Figure 357: Set Controller State – Command Dword 15**

Bits	Description
31:00	<b>Number of Dwords (NUMD):</b> This field specifies the number of dwords being transferred.  This field is only allowed to be cleared to the value of 0h if the Sequence Indicator field is set to 10b.

If the host specifies an offset (i.e., the CSLO field and the CSUO field) that is greater than the size of the Controller State data, then the controller shall abort the command with a status code of Invalid Field in Command.

**Figure 358: Controller State Data Structure**

Bytes	Description					
<b>Header</b>						
01:00	<b>Version (VER):</b> This field indicates the version of this data structure. This field shall be cleared to 0h.					
02	<b>Controller State Attributes (CSATTR):</b> This field specifies attributes associated with the controller state.					
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>07:01</td> <td>Reserved</td> </tr> <tr> <td>00</td> <td><b>Controller Suspended (CP):</b> If this bit is set to '1', then the controller associated with this data structure was suspended for the entire duration of the processing of the Migration Receive command that reported in contents this data structure. If this bit is cleared to '0', then the controller associated with this data structure was not suspended for the entire duration of the processing of the Migration Receive command that reported the contents in this data structure.</td> </tr> </tbody> </table>	Bits	Description	07:01	Reserved	00
Bits	Description					
07:01	Reserved					
00	<b>Controller Suspended (CP):</b> If this bit is set to '1', then the controller associated with this data structure was suspended for the entire duration of the processing of the Migration Receive command that reported in contents this data structure. If this bit is cleared to '0', then the controller associated with this data structure was not suspended for the entire duration of the processing of the Migration Receive command that reported the contents in this data structure.					
15:03	Reserved					
31:16	<b>NVMe Controller State Size (NVMECSS):</b> This field indicates the number of dwords in the NVMe Controller State field.					
47:32	<b>Vendor Specific Size (VSS):</b> This field indicates the number of dwords in the Vendor Specific Data field.					
<b>Controller State Data</b>						
(NVMECSS*4)+47:48	<b>NVMe Controller State (NVMECS):</b> If the NVMECSS field is non-zero, then this field contains the controller state data as defined in Figure 359. If the NVMECSS field is cleared to 0h, then this field does not exist in the data structure.					
((VSS+NVMECSS)*4)+47: (NVMECSS*4)+48	<b>Vendor Specific Data (VSD):</b> If the VSS field is non-zero, then this field contains vendor specific data. If the VSS field is cleared to 0h, then this field does not exist in the data structure.					

If the CSVI field (refer to Figure 354) specifies a non-zero index that is not defined by the Supported Controller State Formats data structure (refer to Figure 329), then the controller shall abort the command with a status code of Invalid Field in Command.

If the CSVI field (refer to Figure 354) is cleared to 0h and the NVMe Controller State Size field is non-zero, then the controller shall abort the command with a status code of Invalid Field in Command.

If the CSUIDI field (refer to Figure 354) specifies a non-zero index that is not defined by the Supported Controller State Formats data structure, then the controller shall abort the command with a status code of Invalid Field in Command.

If the NVMe Controller State field exists (i.e., the NVMECSS field is non-zero) and any I/O Submission Queue or I/O Completion Queue exists in the controller specified by the Controller Identifier (CNTLID) field, then the controller shall abort the command with a status code of Invalid Field in Command.

Figure 359 defines the NVMe Controller State data structure that identifies the state of the all the I/O Submission Queues and I/O Completion Queues for a controller. Any controller state defined by this specification that is not included in the NVMe Controller State data structure is either included in the Vendor Specific Data field in the Controller State data structure (refer to Figure 358) or is obtained in a vendor specific manner.



**Figure 359: NVMe Controller State Data Structure**

Bytes	Description
<b>Header</b>	
01:00	<b>Version (VER):</b> This field indicates the version of this data structure. This field shall be cleared to 0h.
03:02	<b>Number of I/O Submission Queues (NIOSQ):</b> This field indicates the number I/O Submission Queues contained in this data structure.
05:04	<b>Number of I/O Completion Queues (NIOCQ):</b> This field indicates the number I/O Completion Queues contained in this data structure.
07:06	Reserved
<b>I/O Submission Queue List</b>	
31:08	<b>I/O Submission Queue State 0:</b> This field contains the state of the first I/O Submission Queue reported, if any. The contents of this field are defined in the I/O Submission Queue State data structure (refer to Figure 360).
55:32	<b>I/O Submission Queue State 1:</b> This field contains the state of the second I/O Submission Queue reported, if any. The contents of this field are defined in the I/O Submission Queue State data structure (refer to Figure 360).
...	
$((\text{NIOSQ}) * 24) + 7:$ $((\text{NIOSQ}-1) * 24) + 8$	<b>I/O Submission Queue State NIOSQ-1:</b> This field contains the state of the last I/O Submission Queue reported, if any. The contents of this field are defined in the I/O Submission Queue State data structure (refer to Figure 360).
<b>I/O Completion Queue List</b>	
$((\text{NIOCQ}+1) * 24) + 7:$ $((\text{NIOCQ}) * 24) + 8$	<b>I/O Completion Queue State 0:</b> This field contains the state of the first I/O Completion Queue reported, if any. The contents of this field are defined in the I/O Completion Queue State data structure (refer to Figure 361).
$((\text{NIOCQ}+2) * 24) + 7:$ $((\text{NIOCQ}+1) * 24) + 8$	<b>I/O Completion Queue State 1:</b> This field contains the state of the second I/O Completion Queue reported, if any. The contents of this field are defined in the I/O Completion Queue State data structure (refer to Figure 361).
...	
$((\text{NIOCQ}) * 24) +$ $((\text{NIOSQ}) * 24) + 15:$ $((\text{NIOCQ}-1) * 24) +$ $((\text{NIOSQ}) * 24) + 8$	<b>I/O Completion Queue State NIOSQ-1:</b> This field contains the state of the last I/O Completion Queue reported, if any. The contents of this field are defined in the I/O Completion Queue State data structure (refer to Figure 361).

The I/O Submission Queue list shall be listed in ascending order by I/O Submission Queue Identifier.

The I/O Completion Queue list shall be listed in ascending order by I/O Completion Queue Identifier.

**Figure 360: I/O Submission Queue State Data Structure**

Bytes	Description
07:00	<b>I/O Submission PRP Entry 1 (IOSQPRP1):</b> This field contains the contents of the PRP1 field (refer to Figure 477) from the Create I/O Submission Queue command that created this I/O Submission Queue.
09:08	<b>I/O Submission Queue Size (IOSQSIZE):</b> This field contains the contents of the QSIZE field (refer to Figure 478) from the Create I/O Submission Queue command that created this I/O Submission Queue.
11:10	<b>I/O Submission Queue Identifier (IOSQQID):</b> This field contains the contents of the QID field (refer to Figure 478) from the Create I/O Submission Queue command that created this I/O Submission Queue.
13:12	<b>I/O Completion Queue Identifier (IOSQCQID):</b> This field contains the contents of the CQID field (refer to Figure 479) from the Create I/O Submission Queue command that created this I/O Submission Queue.

Figure 360: I/O Submission Queue State Data Structure

Bytes	Description								
15:14	<b>I/O Submission Queue Attributes (IOSQA):</b> This field contains various attributes associated with the I/O Submission Queue.								
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:03</td> <td>Reserved</td> </tr> <tr> <td>02:01</td> <td><b>I/O Submission Queue Priority (IOSQQPRIO):</b> This field contains the contents of the QPRIO field (refer to Figure 479) from the Create I/O Submission Queue command that created this I/O Submission Queue.</td> </tr> <tr> <td>00</td> <td><b>I/O Submission Queue Physically Contiguous (IOSQPC):</b> This bit contains the contents of the PC bit (refer to Figure 479) from the Create I/O Submission Queue command that created this I/O Submission Queue.</td> </tr> </tbody> </table>	Bits	Description	15:03	Reserved	02:01	<b>I/O Submission Queue Priority (IOSQQPRIO):</b> This field contains the contents of the QPRIO field (refer to Figure 479) from the Create I/O Submission Queue command that created this I/O Submission Queue.	00	<b>I/O Submission Queue Physically Contiguous (IOSQPC):</b> This bit contains the contents of the PC bit (refer to Figure 479) from the Create I/O Submission Queue command that created this I/O Submission Queue.
	Bits	Description							
	15:03	Reserved							
02:01	<b>I/O Submission Queue Priority (IOSQQPRIO):</b> This field contains the contents of the QPRIO field (refer to Figure 479) from the Create I/O Submission Queue command that created this I/O Submission Queue.								
00	<b>I/O Submission Queue Physically Contiguous (IOSQPC):</b> This bit contains the contents of the PC bit (refer to Figure 479) from the Create I/O Submission Queue command that created this I/O Submission Queue.								
17:16	<b>I/O Submission Queue Head Pointer (IOSQHP):</b> This field indicates the value of the I/O Submission Queue head pointer.								
19:18	<b>I/O Submission Queue Tail Pointer (IOSQTP):</b> This field indicates the value of the I/O Submission Queue tail pointer.								
23:20	Reserved								

Figure 361: I/O Completion Queue State Data Structure

Bytes	Description												
07:00	<b>I/O Completion Queue Size PRP Entry 1 (IOCQPRP1):</b> This field contains the contents of the PRP1 field (refer to Figure 473) from the Create I/O Completion Queue command that created this I/O Completion Queue.												
09:08	<b>I/O Completion Queue Size (IOCQSIZE):</b> This field contains the contents of the QSIZE field (refer to Figure 474) from the Create I/O Completion Queue command that created this I/O Completion Queue.												
11:10	<b>I/O Completion Queue Identifier (IOCQQID):</b> This field contains the contents of the PRP1 field (refer to Figure 474) from the Create I/O Completion Queue command that created this I/O Completion Queue.												
13:12	<b>I/O Completion Queue Head Pointer (IOCQHP):</b> This field indicates the value of the I/O Completion Queue head pointer.												
15:14	<b>I/O Completion Queue Tail Pointer (IOCQTP):</b> This field indicates the value of the I/O Completion Queue tail pointer.												
19:16	<b>I/O Completion Queue Attributes (IOCQA):</b> This field contains various attributes associated with the I/O Completion Queue.												
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>31:16</td> <td><b>I/O Completion Queue Interrupt Vector (IOCQIV):</b> This field contains the contents of the IV field (refer to Figure 475) from the Create I/O Completion Queue command that created this I/O Completion Queue.</td> </tr> <tr> <td>15:03</td> <td>Reserved</td> </tr> <tr> <td>02</td> <td><b>Slot 0 Phase Tag (S0PT):</b> This bit states the value of the Phase Tag bit for slot 0 of this I/O Completion Queue.</td> </tr> <tr> <td>01</td> <td><b>I/O Completion Queue Interrupts Enabled (IOCQIEN):</b> This bit contains the contents of the IEN bit (refer to Figure 475) from the Create I/O Completion Queue command that created this I/O Completion Queue.</td> </tr> <tr> <td>00</td> <td><b>I/O Completion Queue Physically Contiguous (IOCQPC):</b> This bit contains the contents of the PC bit (refer to Figure 475) from the Create I/O Completion Queue command that created this I/O Completion Queue.</td> </tr> </tbody> </table>	Bits	Description	31:16	<b>I/O Completion Queue Interrupt Vector (IOCQIV):</b> This field contains the contents of the IV field (refer to Figure 475) from the Create I/O Completion Queue command that created this I/O Completion Queue.	15:03	Reserved	02	<b>Slot 0 Phase Tag (S0PT):</b> This bit states the value of the Phase Tag bit for slot 0 of this I/O Completion Queue.	01	<b>I/O Completion Queue Interrupts Enabled (IOCQIEN):</b> This bit contains the contents of the IEN bit (refer to Figure 475) from the Create I/O Completion Queue command that created this I/O Completion Queue.	00	<b>I/O Completion Queue Physically Contiguous (IOCQPC):</b> This bit contains the contents of the PC bit (refer to Figure 475) from the Create I/O Completion Queue command that created this I/O Completion Queue.
	Bits	Description											
	31:16	<b>I/O Completion Queue Interrupt Vector (IOCQIV):</b> This field contains the contents of the IV field (refer to Figure 475) from the Create I/O Completion Queue command that created this I/O Completion Queue.											
	15:03	Reserved											
02	<b>Slot 0 Phase Tag (S0PT):</b> This bit states the value of the Phase Tag bit for slot 0 of this I/O Completion Queue.												
01	<b>I/O Completion Queue Interrupts Enabled (IOCQIEN):</b> This bit contains the contents of the IEN bit (refer to Figure 475) from the Create I/O Completion Queue command that created this I/O Completion Queue.												
00	<b>I/O Completion Queue Physically Contiguous (IOCQPC):</b> This bit contains the contents of the PC bit (refer to Figure 475) from the Create I/O Completion Queue command that created this I/O Completion Queue.												
23:20	Reserved												

If the CSVI field (refer to Figure 354) is cleared to 0h and the NVMe Controller State Size field (refer to Figure 358) is not cleared to 0h, then the controller shall abort the command with a status code of Invalid Field in Command.

If the CSUIDI field (refer to Figure 354) is cleared to 0h and the Vendor Specific Size field (refer to Figure 358) is not cleared to 0h, then the controller shall abort the command with a status code of Invalid Field in Command.

If the CSVI field (refer to Figure 354) is cleared to 0h and the CSUIDI field (refer to Figure 354) is cleared to 0h, then the controller shall abort the command with a status code of Invalid Field in Command.

If the Migration Send command for this management operation specifies the SEQIND field set to 10b and that command completes successfully, then the controller state shall be verified and committed to the specified controller which includes:

- Creating any I/O Submission Queues and I/O Completion Queues specified by the NVMe Controller State field; and
- setting the queue state for each I/O Submission Queues and I/O Completion Queues specified by the NVMe Controller State field.

If any Migration Send command for this management operation is not successful, then the host should attempt to re-transfer the entire controller state data so that any existing controller state is overwritten with the desired controller state.

### 5.1.17.2 Command Completion

Upon completion of the Migration Send command, the controller posts a completion queue entry to the Admin Completion Queue indicating the status for the command. Section 5.1.17.1 describes completion details for each management operation.

Migration Send command specific status values (i.e., SCT field set to 1h) are shown in Figure 362.

**Figure 362: Migration Send – Command Specific Status Values**

Value	Definition
1Fh	<b>Invalid Controller Identifier:</b> The specified controller is not in a condition to set the controller state.
37h	<b>Invalid Controller Data Queue:</b> This error indicates that the specified Controller Data Queue Identifier is invalid for the controller processing the command.
38h	<b>Not Enough Resources:</b> This error indicates that there is not enough resources in the controller to process the command.
3Ah	<b>Controller Not Suspended:</b> The operation requested is not allowed if the specified controller is not suspended (refer to section 5.1.17.1.1).

### 5.1.18 NVMe-MI Receive command

Refer to the NVM Express Management Interface Specification for details on the NVMe-MI Receive command.

### 5.1.19 NVMe-MI Send command

Refer to the NVM Express Management Interface Specification for details on the NVMe-MI Send command.

### 5.1.20 Namespace Attachment command

The Namespace Attachment command is used to attach and detach controllers from a namespace. The attach and detach operations (refer to section 8.1.15) are persistent across all reset events. Namespace attach and detach operations are persistent across Virtualization Management commands that set a secondary controller offline.

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

The Select field determines the data structure used as part of the command. The data structure is 4,096 bytes in size. The data structure used for Controller Attach and Controller Detach is a Controller List (refer to section 4.6.1). The controllers that are to be attached or detached, respectively, are described in the data structure.

If the SEL field specifies the Controller Attach value, then

- If the Maximum Domain Namespace Attachments (MAXDNA) field in the Identify Controller data structure (refer to Figure 312) is non-zero, then:

- For each controller specified in the controller list, if attaching the namespace to that I/O controller causes the sum of the number of namespaces attached to each I/O controller in the Domain to be greater than the value specified in the MAXDNA field, then the controller shall abort the command with a status code of Namespace Attachment Limit Exceeded;

and

- For each I/O controller specified in the controller list, if the Maximum I/O Controller Namespace Attachments (MAXCNA) field in the Identify Controller data structure for that controller is non-zero, then:
  - If attaching the namespace to that I/O controller causes that I/O controller to have the number of attached namespaces to be greater than the value specified in the MAXCNA field, then the controller shall abort the command with a status code of Namespace Attachment Limit Exceeded.

If an attempt is made to attach a namespace to a controller that does not support the corresponding I/O Command Set, then the command shall be aborted with a status code of I/O Command Set Not Supported.

If an attempt is made to attach a namespace to a controller that supports the corresponding I/O Command Set and the corresponding I/O Command Set is not enabled by the I/O Command Set profile feature, then the command shall be aborted with a status code of I/O Command Set Not Enabled.

**Figure 363: Namespace Attachment – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field. If using PRPs, this field shall not be a pointer to a PRP List as the data buffer may not cross more than one page boundary.

**Figure 364: Namespace Attachment – Command Dword 10**

Bits	Description								
31:04	Reserved								
03:00	<p><b>Select (SEL):</b> This field selects the type of attachment to perform.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Controller Attach</td> </tr> <tr> <td>1h</td> <td>Controller Detach</td> </tr> <tr> <td>2h to Fh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	Controller Attach	1h	Controller Detach	2h to Fh	Reserved
Value	Definition								
0h	Controller Attach								
1h	Controller Detach								
2h to Fh	Reserved								

### 5.1.20.1 Command Completion

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

Command specific status values associated with the Namespace Attachment command are defined in Figure 365. For failures, the byte offset of the first failing entry is reported in the Command Specific Information field of the Error Information Log Entry. The controller does not process further entries in the Controller List after an error is encountered.

**Figure 365: Namespace Attachment – Command Specific Status Values**

Value	Definition
18h	<b>Namespace Already Attached:</b> The controller is already attached to the specified namespace.
19h	<b>Namespace is Private:</b> The controller is not attached to the namespace. The request to attach the controller could not be completed because the namespace is private and is already attached to one controller.
1Ah	<b>Namespace Not Attached:</b> The controller is not attached to the namespace. The request to detach the controller could not be completed.

**Figure 365: Namespace Attachment – Command Specific Status Values**

Value	Definition
1Ch	<b>Controller List Invalid:</b> The controller list provided is invalid or the controller list contains an Administrative controller.
25h	<b>ANA Attach Failed:</b> The controller is not attached to the namespace as a result of an ANA condition (e.g., attaching the controller would result in an ANA Persistent Loss state (refer to section 8.1.1.7)).
27h	<b>Namespace Attachment Limit Exceeded:</b> Attaching the namespace to a controller causes maximum number of namespace attachments allowed to be exceeded.
29h	<b>I/O Command Set Not Supported:</b> The request to attach the controller could not be completed due to the I/O Command Set corresponding to the namespace is not supported by the controller.
2Ah	<b>I/O Command Set Not Enabled:</b> The request to attach the controller could not be completed due to the I/O Command Set corresponding to the namespace is restricted by the I/O Command Set profile feature.

### 5.1.21 Namespace Management command

The Namespace Management command is used to manage namespaces (refer to section 8.1.15), including create and delete operations.

Note: The controller continues to execute commands submitted to I/O Submission Queues while this operation is in progress.

If the Namespace Management command supported, then the Namespace Attachment command (refer to section 5.1.20) shall also be supported. The Namespace Management command shall not be supported by controllers in an Exported NVM subsystem.

The host uses the Namespace Attachment command to attach or detach a namespace to or from a controller. The create operation does not attach the namespace to a controller. As a side effect of the delete operation, the namespace is detached from all controllers as the namespace is no longer present in the system. It is recommended that host software detach all controllers from a namespace prior to deleting the namespace. If the namespace is not detached from all controllers prior to being deleted, then Attached Namespace Attribute Changed asynchronous events are reported as specified in section 8.1.15.2. If a namespace is deleted, then Allocated Namespace Attribute Changed asynchronous events are reported as specified in section 8.1.15.2.

The data structure used for the create operation is defined in Figure 369 and the CSI field specifies the I/O Command Set specification. There is no data structure transferred for the delete operation.

The Namespace Management command uses the Data Pointer, Command Dword 10, and Command Dword 11 fields. All other command specific fields are reserved.

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

- **Create:** The NSID field is reserved for this operation; host software clears this field to a value of 0h. The controller shall select an available Namespace Identifier to use for the operation; or
- **Delete:** This field specifies the previously created namespace to delete in this operation. Specifying a value of FFFFFFFFh is used to delete all namespaces in the NVM subsystem. If the value of FFFFFFFFh is specified and there are zero valid namespaces, the command completes successfully.

**Figure 366: Namespace Management – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field. If using PRPs, this field shall not be a pointer to a PRP List as the data buffer may not cross more than one page boundary.

**Figure 367: Namespace Management – Command Dword 10**

Bits	Description								
31:04	Reserved								
03:00	<p><b>Select (SEL):</b> This field selects the type of management operation to perform.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Create</td> </tr> <tr> <td>1h</td> <td>Delete</td> </tr> <tr> <td>2h to Fh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	Create	1h	Delete	2h to Fh	Reserved
Value	Definition								
0h	Create								
1h	Delete								
2h to Fh	Reserved								

**Figure 368: Namespace Management – Command Dword 11**

Bits	Description
31:24	<p><b>Command Set Identifier (CSI):</b> For a create operation (i.e., SEL 0h), this field specifies the I/O Command Set for the created namespace. A CSI value of 0h creates a namespace using the NVM Command Set. For all other operations this field is reserved.</p> <p>Values for this field are defined by Figure 311.</p>
23:0	Reserved

**Figure 369: Namespace Management – Data Structure for Create**

Bytes	Description
511:00	<b>Specific to the I/O Command Set (SIOCS):</b> Refer to the Namespace Management command section of the applicable I/O Command Set specification.
1023:512	Reserved
4095:1024	<b>Vendor Specific (VS)</b>

### 5.1.21.1 Command Completion

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

Namespace Management command specific status values (i.e., SCT field set to 1h) are shown in Figure 370.

**Figure 370: Namespace Management – Command Specific Status Values**

Value	Definition
0Ah	<p><b>Invalid Format:</b> The User Data Format specified is not supported. This may be due to various conditions, including:</p> <ol style="list-style-type: none"> <li>1) specifying an invalid User Data Format number;</li> <li>2) enabling protection information when there are not sufficient resources (e.g., metadata per LBA); or</li> <li>3) the specified format is not available in the current configuration.</li> </ol>
15h	<b>Namespace Insufficient Capacity:</b> Creating the namespace requires more unallocated capacity than is currently available. The Command Specific Information field in the Error Information Log Entry data structure (refer to Figure 205) specifies the total amount of unallocated NVM capacity required to create the namespace in bytes.
16h	<b>Namespace Identifier Unavailable:</b> The number of namespaces supported has been exceeded.
1Bh	<b>Thin Provisioning Not Supported:</b> Thin provisioning is not supported by the controller.
24h	<p><b>ANA Group Identifier Invalid:</b> The specified ANA Group Identifier (ANAGRPID) is not supported in the submitted command. This may be due to various conditions, including:</p> <ol style="list-style-type: none"> <li>a) specifying an ANAGRPID that does not exist;</li> <li>b) the controller does not allow an ANAGRPID to be specified (i.e., the ANA Group ID Support (ANAGIDS) bit in the ANACAP field is cleared to '0'); or</li> </ol>

**Figure 370: Namespace Management – Command Specific Status Values**

Value	Definition
	<p>c) the specified ANAGRPID is not supported by the controller processing the command (e.g., the specified value exceeds ANAGRPMAX (refer to Figure 312)).</p> <p>If the host specified a non-zero ANAGRPID, retrying the command with the ANAGRPID field cleared to 0h may succeed.</p>
29h	<b>I/O Command Set Not Supported:</b> The I/O Command Set specified for a create operation is not supported by the controller.

Dword 0 of the completion queue entry contains the Namespace Identifier created. The definition of Dword 0 of the completion queue entry is in Figure 371.

**Figure 371: Namespace Management – Completion Queue Entry Dword 0**

Bits	Description
31:00	<b>Namespace Identifier (NSID):</b> This field specifies the namespace identifier created in a Create operation. This field is reserved for all other operations.

### 5.1.22 Sanitize command

The Sanitize command is used to start a sanitize operation that targets the NVM subsystem or to recover from a previously failed sanitize operation that targeted the NVM subsystem. The sanitize operation types that may be supported are Block Erase, Crypto Erase, and Overwrite.

A sanitize operation consists of:

- sanitize processing (refer to section 8.1.24), which may include:
  - deallocation of all media allocated for user data; and
  - additional media modification;
- optional verification of media allocated for user data; and
- post-verification deallocation of all media allocated for user data following media verification, if any, as described in section 8.1.24.

All sanitize operations are performed in the background (i.e., completion of the Sanitize command that starts a sanitize operation does not indicate completion of that sanitize operation). Refer to section 8.1.24 for details on the sanitize operation.

If the NVM subsystem supports multiple domains and the Sanitize command is not able to start a sanitize operation as a result of the NVM subsystem being divided (refer to section 3.2.5), then the controller shall abort the Sanitize command shall be aborted with a status code of Asymmetric Access Inaccessible or Asymmetric Access Persistent Loss.

The Sanitize command shall not be supported by Exported NVM Subsystems (refer to section 8.1.24.5).

The Sanitize Capabilities (SANICAP) field in the Identify Controller data structure (refer to Figure 312) indicates:

- a) the sanitize operation types supported;
- b) whether setting the No-Deallocate After Sanitize (NDAS) bit (refer to Figure 372) causes media to be modified as part of sanitize processing;
- c) whether the controller inhibits the functionality of the No-Deallocation After Sanitize bit in the Sanitize command; and
- d) whether the controller supports the Media Verification state and the Post-Verification Deallocation state (refer to the Verification Support (VERS) bit in the SANICAP field in Figure 312).

If a Sanitize command specifies an unsupported value in the SANACT field (refer to Figure 372), then the controller shall abort the command with a status code of Invalid Field in Command.

If the Verification Support (VERS) bit is cleared to '0' and a Sanitize command specifies the Enter Media Verification State (EMVS) bit set to '1' (refer to Figure 372), then the controller shall abort the command with a status code of Invalid Field in Command.

If the Verification Support (VERS) bit is set to '1' and a Sanitize command is processed that specifies:

- a) the Enter Media Verification State (EMVS) bit set to '1', the SANACT field set to a value of 010b (i.e., Block Erase) or a value of 100b (i.e., Crypto Erase), and the No-Deallocate After Sanitize (NDAS) bit cleared to '0', then successful sanitize processing is followed by entry to the Media Verification state;
- b) the Enter Media Verification State (EMVS) bit set to '1' and:
  - a) the SANACT field set to 011b (i.e., Overwrite); or
  - b) the No-Deallocate After Sanitize (NDAS) bit set to '1';
 then the controller shall abort the command with a status code of Invalid Field in Command;
- c) the SANACT field set to 101b (i.e., Exit Media Verification State) and the sanitization target is in the Media Verification state, then:
  - the controller does not start a new sanitize operation; and
  - the sanitization target transitions from the Media Verification state to the Post-Verification Deallocation state, in which all media allocated for user data in the NVM subsystem is deallocated;
 or
- d) the SANACT field set to 101b (i.e., Exit Media Verification State) and the sanitization target is not in the Media Verification state, then the controller shall abort the command with a status code of Invalid Field in Command.

If any Persistent Memory Region is enabled in an NVM subsystem, then the controller shall abort any Sanitize command with a status code of Sanitize Prohibited While Persistent Memory Region is Enabled.

If any namespace is write protected in an NVM subsystem (refer to section 8.1.16), then the controller aborts any Sanitize command with a status code of Namespace is Write Protected.

If a firmware activation with reset is pending, then the controller shall abort any Sanitize command (refer to section 5.1.8.1 and section 3.11).

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

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

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

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

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

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

Activation of new firmware is prohibited during a sanitize operation (refer to section 8.1.24.4).

If one or more controllers in the NVM subsystem is suspended (refer to section 5.1.17.1.1), then the controller shall abort the command with a status code of Controller Suspended.



Support for Sanitize commands in a Controller Memory Buffer (i.e., submitted to an Admin Submission Queue in a Controller Memory Buffer or specifying an Admin Completion Queue in a Controller Memory Buffer) is implementation specific. If an implementation does not support Sanitize commands in a Controller Memory Buffer and a controller’s Admin Submission Queue or Admin Completion Queue is in the Controller Memory Buffer, then the controller shall abort all Sanitize commands with a status code of Command Not Supported for Queue in CMB.

All sanitize operations (i.e., Block Erase, Crypto Erase, and Overwrite) are performed in the background (i.e., Sanitize command completion does not indicate sanitize operation completion). If a sanitize operation starts as a result of a Sanitize command, then the controller shall complete that Sanitize command with a status code of Successful Completion. If the controller completes a Sanitize command with any status code other than Successful Completion, then the controller:

- shall not start the sanitize operation for that command;
- shall not modify the Sanitize Status log page; and
- shall not alter any user data.

The Sanitize command uses Command Dword 10 and Command Dword 11. All other command specific fields are reserved.

**Figure 372: Sanitize – Command Dword 10**

Bits	Description
31:11	Reserved
10	<p><b>Enter Media Verification State (EMVS):</b> If this bit is set to ‘1’, then the Media Verification state shall be entered if sanitize processing completes successfully (i.e., the Global Data Erased (GDE) bit is set to ‘1’ (refer to Figure 291)).</p> <p>If this bit is cleared to ‘0’, then this bit shall have no effect.</p> <p>If the SANACT field does not specify starting a sanitize operation (i.e., is set to any value other than 010b, 011b, or 100b), then this bit shall be ignored by the controller.</p>
09	<p><b>No-Deallocate After Sanitize (NDAS):</b> If this bit is set to ‘1’ and the No-Deallocate Inhibited bit (refer to Figure 312) is cleared to ‘0’, then the controller shall not deallocate any media allocated for user data as a result of successfully completing the sanitize operation. If this bit is:</p> <ul style="list-style-type: none"> <li>a) cleared to ‘0’; or</li> <li>b) set to ‘1’ and the No-Deallocate Inhibited bit is set to ‘1’,</li> </ul> <p>then the controller should deallocate all media allocated for user data as a result of successfully completing the sanitize operation.</p> <p>If the SANACT field does not specify starting a sanitize operation (i.e., is set to any value other than 010b, 011b, or 100b), then this bit shall be ignored by the controller.</p>
08	<p><b>Overwrite Invert Pattern Between Passes (OIPBP):</b> If this bit is set to ‘1’, then the Overwrite Pattern shall be inverted between passes. If this bit is cleared to ‘0’, then the overwrite pattern shall not be inverted between passes. If the Sanitize Action field is set to a value other than 011b (i.e., Overwrite), then this bit shall be ignored by the controller.</p>
07:04	<p><b>Overwrite Pass Count (OWPASS):</b> This field specifies the number of overwrite passes (i.e., how many times the media is to be overwritten) using the data from the Overwrite Pattern field of this command. A value of 0h specifies 16 overwrite passes. If the Sanitize Action field is set to a value other than 011b (i.e., Overwrite), then this field shall be ignored by the controller.</p>
03	<p><b>Allow Unrestricted Sanitize Exit (AUSE):</b> If this bit is set to ‘1’, then the sanitize processing is performed in unrestricted completion mode (i.e., in the Unrestricted Processing state; refer to section 8.1.24.3.4). If this bit is cleared to ‘0’, then the sanitize processing is performed in restricted completion mode (i.e., in the Restricted Processing state; refer to section 8.1.24.3.2).</p> <p>If the SANACT field does not specify starting a sanitize operation (i.e., is set to any value other than 010b, 011b, or 100b), then this bit shall be ignored by the controller.</p>

**Figure 372: Sanitize – Command Dword 10**

Bits	Description																
02:00	<b>Sanitize Action (SANACT):</b> This field specifies the sanitize action to perform.																
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>Reserved</td> </tr> <tr> <td>001b</td> <td>Exit Failure Mode</td> </tr> <tr> <td>010b</td> <td>Start a Block Erase sanitize operation</td> </tr> <tr> <td>011b</td> <td>Start an Overwrite sanitize operation</td> </tr> <tr> <td>100b</td> <td>Start a Crypto Erase sanitize operation</td> </tr> <tr> <td>101b</td> <td>Exit Media Verification State</td> </tr> <tr> <td>110b to 111b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	000b	Reserved	001b	Exit Failure Mode	010b	Start a Block Erase sanitize operation	011b	Start an Overwrite sanitize operation	100b	Start a Crypto Erase sanitize operation	101b	Exit Media Verification State	110b to 111b	Reserved
	Value	Definition															
	000b	Reserved															
	001b	Exit Failure Mode															
	010b	Start a Block Erase sanitize operation															
	011b	Start an Overwrite sanitize operation															
	100b	Start a Crypto Erase sanitize operation															
101b	Exit Media Verification State																
110b to 111b	Reserved																

**Figure 373: Sanitize – Command Dword 11**

Bits	Description
31:00	<p><b>Overwrite Pattern (OVRPAT):</b> This field specifies a 32-bit pattern that is used for the Overwrite sanitize operation. Refer to section 8.1.24.</p> <p>If the Sanitize Action field is set to a value other than 011b (i.e., Overwrite), then this field shall be ignored by the controller.</p>

### 5.1.22.1 Command Completion

When the command is complete, the controller shall post a completion queue entry to the Admin Completion Queue indicating the status for the command. All sanitize operations are performed in the background (i.e., completion of the Sanitize command that started that sanitize operation does not indicate completion of the sanitize operation). If a sanitize operation starts (refer to section 8.1.24.3), then the Sanitize Status log page shall be updated before posting the completion queue entry for the command that started that sanitize operation.

Sanitize command specific status values (i.e., SCT field set to 1h) are shown in Figure 374.

**Figure 374: Sanitize – Command Specific Status Values**

Value	Definition
0Bh	<b>Firmware Activation Requires Conventional Reset:</b> 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.
10h	<b>Firmware Activation Requires NVM Subsystem Reset:</b> The sanitize operation could not be started because a firmware activation is pending and an NVM Subsystem Reset is required.
11h	<b>Firmware Activation Requires Controller Level Reset:</b> The sanitize operation could not be started because a firmware activation is pending and a Controller Level Reset is required.
23h	<b>Sanitize Prohibited While Persistent Memory Region is Enabled:</b> A sanitize operation is prohibited while the Persistent Memory Region is enabled.
39h	<b>Controller Suspended:</b> One or more controllers in the NVM subsystem have been suspended by a Migration Send command.

### 5.1.23 Security Receive command

The Security Receive command transfers the status and data result of one or more Security Send commands that were previously submitted to the controller.

The association between a Security Receive command and previous Security Send commands is dependent on the Security Protocol. The format of the data to be transferred is dependent on the Security Protocol. Refer to SPC-5 for Security Protocol details.

Each Security Receive command returns the appropriate data corresponding to a Security Send command as defined by the rules of the Security Protocol. The Security Receive command data may not be retained if there is a loss of communication between the controller and host, or if a Controller Level Reset occurs.

The fields used are Data Pointer, Command Dword 10, and Command Dword 11 fields. All other command specific fields are reserved.

**Figure 375: Security Receive – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field.

**Figure 376: Security Receive – Command Dword 10**

Bits	Description
31:24	<b>Security Protocol (SECP):</b> This field specifies the security protocol as defined in SPC-5. The controller shall abort the command with a status code of Invalid Field in Command if an unsupported value of the Security Protocol is specified.
23:16	<b>SP Specific 1 (SPSP1):</b> The value of this field contains bits 15:08 of the Security Protocol Specific field as defined in SPC-5.
15:08	<b>SP Specific 0 (SPSP0):</b> The value of this field contains bits 07:00 of the Security Protocol Specific field as defined in SPC-5.
07:00	<b>NVMe Security Specific Field (NSSF):</b> Refer to Figure 378 for definition of this field for Security Protocol EAh. For all other Security Protocols this field is reserved.

**Figure 377: Security Receive – Command Dword 11**

Bits	Description
31:00	<b>Allocation Length (AL):</b> The value of this field is specific to the Security Protocol In command with the INC_512 field cleared to 0h as defined in SPC-5.

### 5.1.23.1 Command Completion

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

### 5.1.23.2 Security Protocol 00h

A Security Receive command with the Security Protocol field cleared to 00h shall return information about the security protocols supported by the controller. This command is used in the security discovery process and is not associated with a Security Send command. Refer to SPC-5 for the details of Security Protocol 00h and the SP Specific field.

### 5.1.23.3 Security Protocol EAh

Security Protocol EAh is assigned for NVMe interface use (refer to ACS-4). This protocol may be used in Security Receive and Security Send commands. The specific usage type is defined by the Security Protocol Specific Field defined in Figure 378.

**Figure 378: Security Protocol EAh – Security Protocol Specific Field Values**

SP Specific (SPSP) Value	Definition	NVMe Security Specific Field (NSSF) Definition
0001h	Replay Protected Memory Block	RPMB Target
0002h to FFFFh	Reserved	Reserved

### 5.1.24 Security Send command

The Security Send command is used to transfer security protocol data to the controller. The data structure transferred to the controller as part of this command contains security protocol specific commands to be performed by the controller. The data structure transferred may also contain data or parameters associated with the security protocol commands. Status and data that is to be returned to the host for the security

protocol commands submitted by a Security Send command are retrieved with the Security Receive command defined in section 5.1.23.

The association between a Security Send command and subsequent Security Receive command is Security Protocol field dependent as defined in SPC-5.

The fields used are Data Pointer, Command Dword 10, and Command Dword 11 fields. All other command specific fields are reserved.

**Figure 379: Security Send – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field.

**Figure 380: Security Send – Command Dword 10**

Bits	Description
31:24	<b>Security Protocol (SECP):</b> This field specifies the security protocol as defined in SPC-5. The controller shall abort the command with a status code of Invalid Field in Command if a reserved value of the Security Protocol is specified.
23:16	<b>SP Specific 1 (SPSP1):</b> The value of this field contains bits 15:08 of the Security Protocol Specific field as defined in SPC-5.
15:08	<b>SP Specific 0 (SPSP0):</b> The value of this field contains bits 07:00 of the Security Protocol Specific field as defined in SPC-5.
07:00	<b>NVMe Security Specific Field (NSSF):</b> Refer to Figure 378 for definition of this field for Security Protocol EAh. For all other Security Protocols this field is reserved.

**Figure 381: Security Send – Command Dword 11**

Bits	Description
31:00	<b>Transfer Length (TL):</b> The value of this field is specific to the Security Protocol Out command with the INC_512 field cleared to 0h as defined in SPC-5.

#### 5.1.24.1 Command Completion

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

#### 5.1.25 Set Features command

The Set Features command specifies the attributes of the Feature indicated.

The Set Features command uses the Data Pointer, Command Dword 10, and Command Dword 14. The use of Command Dword 11, Command Dword 12, Command Dword 13, and Command Dword 15 fields is Feature specific. If Command Dword 11, Command Dword 12, Command Dword 13, or Command Dword 15 fields are not used, then the Command Dwords are reserved.

**Figure 382: Set Features – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field. If using PRPs, this field shall not be a pointer to a PRP List as the data buffer may not cross more than one page boundary. If no data structure is used as part of the specified feature, then this field is not used.

**Figure 383: Set Features – Command Dword 10**

Bits	Description
31	<p><b>Save (SV):</b> This bit specifies that the controller shall save the attribute so that the attribute persists through all power states and resets.</p> <p>The controller indicates in the Save and Select Feature Support (SSFS) bit in the Optional NVM Command Support field of the Identify Controller data structure in Figure 312 whether this bit is supported.</p> <p>If the Feature Identifier specified in the Set Features command is not saveable by the controller and the controller receives a Set Features command with this bit set to '1', then the command shall be aborted with a status code of Feature Identifier Not Saveable.</p>
30:08	Reserved
07:00	<b>Feature Identifier (FID):</b> This field indicates the identifier of the Feature that attributes are being specified for.

If the controller supports selection of a UUID by the Set Features command (refer to Figure 385 and section 8.1.28) and the controller supports selection of a UUID for the specified vendor specific feature identifier (refer to Figure 385), then Command Dword 14 is used to specify a UUID Index value (refer to Figure 384). If the controller does not support selection of a UUID by the Set Features command or the controller does not support selection of a UUID for the specified vendor specific feature identifier, then Command Dword 14 does not specify a UUID Index value.

**Figure 384: Set Features – Command Dword 14**

Bits	Description
31:07	Reserved
06:00	<b>UUID Index (UIDX):</b> Refer to Figure 658.

Figure 385 defines the Features that are able to be configured with a Set Features command and retrieved with a Get Features command.

Section 5.1.25.1 describes features that are common to all transport models. Section 5.1.25.2 describes features that are specific to the Memory-based transport model. Section 5.1.25.3 describes features that are specific to the Message-based transport model.

**Figure 385: Set Features – Feature Identifiers**

Feature Identifier	Current Setting Persists Across Power Cycle and Reset <sup>2</sup>	Uses Memory Buffer for Attributes	Feature Name	Scope <sup>6</sup>
00h	Reserved			
01h	No	No	Arbitration	Controller
02h	No	No	Power Management	Controller <sup>7</sup>
03h	Refer to the NVM Command Set Specification			
04h	No	No	Temperature Threshold	Controller
05h	Refer to the NVM Command Set Specification			
06h	No	No	Volatile Write Cache	Controller
07h	No	No	Number of Queues	Controller
08h	No	No	Interrupt Coalescing	Controller
09h	No	No	Interrupt Vector Configuration	Controller
0Ah	Refer to the NVM Command Set Specification			
0Bh	No	No	Asynchronous Event Configuration	Controller
0Ch	No	Yes	Autonomous Power State Transition	Controller <sup>7</sup>

Figure 385: Set Features – Feature Identifiers

Feature Identifier	Current Setting Persists Across Power Cycle and Reset <sup>2</sup>	Uses Memory Buffer for Attributes	Feature Name	Scope <sup>6</sup>
0Dh	No <sup>3</sup>	No <sup>4</sup>	Host Memory Buffer	Controller
0Eh	No	Yes	Timestamp	Controller
0Fh	No	No	Keep Alive Timer	Controller
10h	Yes	No	Host Controlled Thermal Management	Controller
11h	No	No	Non-Operational Power State Config	Controller
12h	Yes	No	Read Recovery Level Config	NVM Set NVM subsystem
13h	No	Yes	Predictable Latency Mode Config	NVM Set
14h	No	No	Predictable Latency Mode Window	NVM Set
15h	Refer to the NVM Command Set Specification			
16h	No	Yes	Host Behavior Support	Controller
17h	Yes	No	Sanitize Config	NVM subsystem
18h	No	No	Endurance Group Event Configuration	Endurance Group
19h	Yes	No	I/O Command Set Profile	Controller
1Ah	Yes	No	Spinup Control	NVM subsystem
1Bh	Yes	No	Power Loss Signaling Config	Domain
1Ch	Refer to the NVM Command Set Specification			
1Dh	Yes	No	Flexible Data Placement	Endurance Group
1Eh	Yes	Yes	Flexible Data Placement Events	Reclaim Unit Handle
1Fh	Yes	Yes	Namespace Admin Label	Namespace
20h	Refer to the Key Value Command Set Specification			
21h	No	Yes	Controller Data Queue	Controller Data Queue per controller
22h to 77h	Reserved			
78h	Yes	Yes	Embedded Management Controller Address	NVM subsystem
79h	Yes	Yes	Host Management Agent Address	NVM subsystem
7Ah to 7Ch	Reserved for Management Features.			
7Dh	No	Yes	Enhanced Controller Metadata	Controller
7Eh	No	Yes	Controller Metadata	Controller
7Fh	No	Yes	Namespace Metadata	Namespace per controller
80h	Yes	No	Software Progress Marker	Controller
81h	No	Yes	Host Identifier	Controller
82h	No	No	Reservation Notification Mask	Namespace
83h	Yes	No	Reservation Persistence	Namespace
84h	No	No	Namespace Write Protection Config	Namespace
85h	No	No	Boot Partition Write Protection Config	Controller
86h to BFh	Reserved			

**Figure 385: Set Features – Feature Identifiers**

Feature Identifier	Current Setting Persists Across Power Cycle and Reset <sup>2</sup>	Uses Memory Buffer for Attributes	Feature Name	Scope <sup>6</sup>
C0h to FFh	Vendor Specific <sup>1, 5</sup>			
<p>Notes:</p> <ol style="list-style-type: none"> <li>1. The behavior of a controller in response to an inactive namespace ID to a vendor specific Feature Identifier is vendor specific.</li> <li>2. This column is only valid if the feature is not saveable (refer to section 4.4). If the feature is saveable, then this column is not used.</li> <li>3. The controller does not save settings for the Host Memory Buffer feature across power states and reset events, however, host software may restore the previous values. Refer to section 8.2.3.</li> <li>4. The feature does not use a memory buffer for Set Features commands and does use a memory buffer for Get Features commands. Refer to section 5.1.25.2.4.</li> <li>5. Selection of a UUID may be supported. Refer to section 8.1.28.</li> <li>6. Refer to Feature Identifiers Supported and Effects log page in section 5.1.12.1.18 for how scope is reported to the host.</li> <li>7. For NVM Subsystems with multiple controllers in the same domain, specifying different power states results in an unspecified power state for that domain.</li> </ol>				

### 5.1.25.1 Common Feature Specific Information

Refer to section 3.1.3.6 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. For more information on Features, including default value definitions, saved value definitions, and current value definitions, refer to section 4.4.

Upon completion of a Set Features command for a feature, the host should rediscover, re-enumerate and/or re-initialize all capabilities associated with that feature. For example, if a namespace capability change may occur for a feature, then host software should pause the use of any associated namespace, submit the Set Features command for that feature and wait for that command to complete, and then re-issue commands to all namespaces affected by that Set Features command.

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

If the controller does not support a changeable value for a Feature (i.e., the Feature is not changeable, refer to section 5.1.11.2), and a Set Features command for that Feature is processed, then if that command specifies a Feature value that:

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

Refer to each feature description in this section for any additional requirements associated with that feature.

#### 5.1.25.1.1 Arbitration (Feature Identifier 01h)

This Feature controls command arbitration within the controller. Refer to section 3.4.4 for command arbitration details. The attributes are specified in Command Dword 11.

If a Get Features command is submitted for this Feature, the attributes specified in Figure 386 are returned in Dword 0 of the completion queue entry for that command.

**Figure 386: Arbitration & Command Processing – Command Dword 11**

Bits	Description
31:24	<b>High Priority Weight (HPW):</b> This field defines the number of commands that may be executed from the high priority service class in each arbitration round. This is a 0's based value.
23:16	<b>Medium Priority Weight (MPW):</b> This field defines the number of commands that may be executed from the medium priority service class in each arbitration round. This is a 0's based value.
15:08	<b>Low Priority Weight (LPW):</b> This field defines the number of commands that may be executed from the low priority service class in each arbitration round. This is a 0's based value.
07:03	Reserved
02:00	<b>Arbitration Burst (AB):</b> Indicates the maximum number of commands that the controller may fetch at one time from a particular Submission Queue. The value is expressed as a power of two (e.g., 000b indicates one, 011b indicates eight). A value of 111b indicates no limit.

#### 5.1.25.1.2 Power Management (Feature Identifier 02h)

This Feature allows the host to configure the controller power state. The attributes are specified in Command Dword 11 (refer to Figure 387).

Upon successful completion of a Set Features command for this Feature, the controller shall be in the Power State specified. For a transition to a non-operational power state, the device may exceed the power indicated for that non-operational power state as defined in section 8.1.17.1 (e.g., while completing this command). If enabled, autonomous power state transitions continue to occur from the new state.

If a Get Features command is submitted for this Feature, the attributes described in Figure 388 are returned in Dword 0 of the completion queue entry for that command.

**Figure 387: Power Management – Command Dword 11**

Bits	Description
31:08	Reserved
07:05	<b>Workload Hint (WH):</b> This field indicates the type of workload expected. This hint may be used to optimize performance. Refer to section 8.1.17.3 for more details.
04:00	<b>Power State (PS):</b> This field indicates the new power state into which the controller is requested to transition. This power state shall be one supported by the controller as indicated in the Number of Power States Supported (NPSS) field in the Identify Controller data structure. If the power state specified is not supported, the controller shall abort the command and should return an error of Invalid Field in Command.

**Figure 388: Power Management – Completion Queue Entry Dword 0**

Bits	Description
31:08	Reserved
07:05	<b>Workload Hint (WH):</b> This field indicates the type of workload. Refer to section 8.1.17.3 for more details.
04:00	<b>Power State (PS):</b> This field indicates the current power state of the controller, or the power state into which the controller is transitioning.

#### 5.1.25.1.3 Temperature Threshold (Feature Identifier 04h)

A controller may report up to nine temperature values in the SMART / Health Information log page (i.e., the Composite Temperature and Temperature Sensor 1 through Temperature Sensor 8; refer to Figure 206). Associated with each implemented temperature sensor is an over temperature threshold and an under temperature threshold. When a temperature is greater than or equal to its corresponding over temperature threshold or less than or equal to its corresponding under temperature threshold, then the Temperature Threshold Condition (TTC) bit is set to '1' of the Critical Warning field in the SMART / Health Information log page (refer to section 5.1.12.1.3). This may trigger an asynchronous event.



The over temperature threshold feature shall be implemented for Composite Temperature. The under temperature threshold Feature shall be implemented for Composite Temperature if a non-zero Warning Composite Temperature Threshold (WCTEMP) field value is reported in the Identify Controller data structure (refer to Figure 312). The over temperature threshold and under temperature threshold features shall be implemented for all implemented temperature sensors (i.e., all Temperature Sensor fields that report a non-zero value).

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

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

#### **5.1.25.1.3.1 Temperature Threshold Hysteresis**

This feature allows the host to specify the temperature hysteresis that a controller shall use to determine the end of a temperature threshold event.

If the controller supports the use of the Temperature Threshold Hysteresis feature as indicated by a non-zero value of the TMPTHMH field (refer to Figure 312), then the controller shall:

- support this feature for the Composite Temperature value (refer to Figure 389); and
- support the Temperature Threshold Hysteresis Recovery event (refer to Figure 312).

If hysteresis is supported for Threshold Temperature Select values other than Composite Temperature (i.e., 0h) (refer to Figure 389); then, associated with each implemented temperature sensor is an over-temperature threshold and an under-temperature threshold with a hysteresis value. If the host specifies hysteresis for a Threshold Temperature Select that is not supported, the controller shall abort the command with a status of Invalid Field in Command.

An over-temperature threshold hysteresis event begins when the value of a temperature field (refer to the temperature values reported as described in section 5.1.25.1.3) transitions from less than its corresponding temperature threshold to greater than or equal to that temperature threshold. That over-temperature threshold hysteresis event ends when the value of that temperature field transitions from greater than or equal to that temperature threshold to less than that temperature threshold minus the value of the Temperature Threshold Hysteresis (TMPTHH) field.

An under-temperature threshold hysteresis event begins when the value of a temperature field transitions from greater than its corresponding temperature threshold to less than or equal to that temperature threshold. That under-temperature threshold hysteresis event ends when the value of that temperature field transitions from less than or equal to that temperature threshold to greater than that temperature threshold plus the value of the Temperature Threshold Hysteresis (TMPTHH) field.

When a temperature threshold hysteresis event has started, the Temperature Threshold Condition (TTC) is set to '1' of the Critical Warning field in the SMART / Health Information log page (refer to section 5.1.12.1.3) is set to '1'. This may trigger an initial asynchronous event.

At the end of a temperature threshold hysteresis event, a Temperature Threshold Hysteresis Recovery event shall be triggered (refer to Figure 391), the TTC bit (refer to section 5.1.12.1.3) shall be cleared to '0' and the Warning Composite Temperature Time field shall stop accumulating the number of minutes.

If the value of the Temperature Threshold Hysteresis field is larger than the value supported by the controller as specified by the TMPTHMH field in Figure 312, the controller shall abort the command with a status code of Invalid Field in Command.

Figure 389: Temperature Threshold – Command Dword 11

Bits	Description																								
31:25	Reserved																								
24:22	<p><b>Temperature Threshold Hysteresis (TMPTHH):</b> This field indicates the temperature hysteresis (i.e., the number of kelvins the controller uses to determine the end of the over temperature or under temperature event). If this field is cleared to 000b, then the hysteresis as defined for this Feature does not apply.</p> <p>If the Temperature Threshold Maximum Hysteresis (TMPTHMH) field in Figure 312 is cleared to 0h, this field shall be cleared to 000b.</p>																								
21:20	<p><b>Threshold Type Select (THSEL):</b> This field selects the threshold type that is modified by a Set Features command and whose threshold value is returned by a Get Features command.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Over Temperature Threshold</td> </tr> <tr> <td>01b</td> <td>Under Temperature Threshold</td> </tr> <tr> <td>10b to 11b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00b	Over Temperature Threshold	01b	Under Temperature Threshold	10b to 11b	Reserved																
Value	Definition																								
00b	Over Temperature Threshold																								
01b	Under Temperature Threshold																								
10b to 11b	Reserved																								
19:16	<p><b>Threshold Temperature Select (TMPSEL):</b> This field selects the temperature whose threshold is modified by a Set Features command and whose threshold value is returned by a Get Features command.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Composite Temperature</td> </tr> <tr> <td>1h</td> <td>Temperature Sensor 1</td> </tr> <tr> <td>2h</td> <td>Temperature Sensor 2</td> </tr> <tr> <td>3h</td> <td>Temperature Sensor 3</td> </tr> <tr> <td>4h</td> <td>Temperature Sensor 4</td> </tr> <tr> <td>5h</td> <td>Temperature Sensor 5</td> </tr> <tr> <td>6h</td> <td>Temperature Sensor 6</td> </tr> <tr> <td>7h</td> <td>Temperature Sensor 7</td> </tr> <tr> <td>8h</td> <td>Temperature Sensor 8</td> </tr> <tr> <td>9h to Eh</td> <td>Reserved</td> </tr> <tr> <td>Fh</td> <td>All implemented temperature sensors in a Set Features command. Reserved in a Get Features command.</td> </tr> </tbody> </table>	Value	Definition	0h	Composite Temperature	1h	Temperature Sensor 1	2h	Temperature Sensor 2	3h	Temperature Sensor 3	4h	Temperature Sensor 4	5h	Temperature Sensor 5	6h	Temperature Sensor 6	7h	Temperature Sensor 7	8h	Temperature Sensor 8	9h to Eh	Reserved	Fh	All implemented temperature sensors in a Set Features command. Reserved in a Get Features command.
Value	Definition																								
0h	Composite Temperature																								
1h	Temperature Sensor 1																								
2h	Temperature Sensor 2																								
3h	Temperature Sensor 3																								
4h	Temperature Sensor 4																								
5h	Temperature Sensor 5																								
6h	Temperature Sensor 6																								
7h	Temperature Sensor 7																								
8h	Temperature Sensor 8																								
9h to Eh	Reserved																								
Fh	All implemented temperature sensors in a Set Features command. Reserved in a Get Features command.																								
15:00	<p><b>Temperature Threshold (TMPTH):</b> Indicates the threshold value for the temperature sensor and threshold type specified in Kelvins.</p>																								

#### 5.1.25.1.4 Volatile Write Cache (Feature Identifier 06h)

This Feature controls the volatile write cache, if present, on the controller. If a volatile write cache is present (refer to the VWC field in Figure 312), then this Feature shall be supported. The attributes are specified in Command Dword 11.

Note: If the controller is able to guarantee that data present in a write cache is written to non-volatile storage media on loss of power, then that write cache is considered non-volatile and this Feature does not apply to that write cache.

If a Get Features command is submitted for this Feature, the attributes specified in Figure 390 are returned in Dword 0 of the completion queue entry for that command.

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

If a volatile write cache is present and the volatile write cache is disabled (i.e., the WCE bit is cleared to '0'), then the user data written by any command to a namespace shall be persistent. Refer to the Volatile Write Cache Presence field contained in the Common Namespace Features field in Figure 319.

**Figure 390: Volatile Write Cache – Command Dword 11**

Bits	Description
31:01	Reserved
00	<b>Volatile Write Cache Enable (WCE):</b> If this bit is set to '1', then the volatile write cache is enabled. If this bit is cleared to '0', then the volatile write cache is disabled.

**5.1.25.1.5 Asynchronous Event Configuration (Feature Identifier 0Bh)**

This Feature controls the events that trigger an asynchronous event notification to the host. This Feature may be used to disable reporting events by the controller in the case of a persistent condition (refer to section 5.1.2). If the condition for an event is true when the corresponding notice is enabled, then an event is sent to the host. The attributes are specified in Command Dword 11.

If a Get Features command is submitted for this Feature, the attributes specified in Figure 391 are returned in Dword 0 of the completion queue entry for that command.

**Figure 391: Asynchronous Event Configuration – Command Dword 11**

Bits	Description
<b>Fabrics Specific</b>	
31	<b>Discovery Log Page Change Notification (DLPCN):</b> This bit indicates that the Discovery controller reports Discovery Log Page Change Notifications. If this bit is set to '1', then the Discovery controller shall send a notification if Discovery log page changes occur.
30	<b>Host Discovery Log Page Change Notification (HDLPCN):</b> This bit indicates that the Discovery controller reports Host Discovery Log Page Change Notifications. If this bit is set to '1', then the Discovery controller shall send a notification if Host Discovery log page changes occur.
29	<b>AVE Discovery Log Page Change Notification (ADLPCN):</b> This bit indicates that the Discovery controller reports AVE Discovery Log Page Change Notifications. If this bit is set to '1', then the Discovery controller shall send a notification if AVE Discovery log page changes occur.
28	<b>Pull Model DDC Request Log Page Change Notification (PMDRLPCN):</b> This bit indicates that the Discovery controller reports Pull Model DDC Request log page Change Notifications. If this bit is set to '1', then the Discovery controller shall send a notification if Pull Model DDC Request log page changes occur.
<b>Non-Fabrics Specific</b>	
27	<b>Zone Descriptor Changed Notices<sup>2</sup> (ZDCN):</b> I/O Command Set specific definition.
26:20	Reserved
19	<b>Allocated Namespace Attribute Notices (ANSAN):</b> This bit determines whether an asynchronous event notification is sent to the host for an Allocated Namespace Attribute Changed asynchronous event (refer to Figure 151). If this bit is set to '1', then the Allocated Namespace Attribute Changed asynchronous event is sent to the host when this condition occurs. If this bit is cleared to '0', then the controller shall not send the Allocated Namespace Attribute Changed asynchronous event to the host.
18	<b>Reachability Group (RGRP0):</b> This bit determines whether an asynchronous event notification is sent to the host when a Reachability Group change occurs (i.e., the contents of the Reachability Groups log page (refer to section 5.1.12.1.25) changed). If this bit is set to '1', then the Reachability Group Change Notices event is sent to the host when this condition occurs. If this bit is cleared to '0', then the controller shall not send the Reachability Group Change Notices event to the host.
17	<b>Reachability Association (RASSN):</b> This bit determines whether an asynchronous event notification is sent to the host when a Reachability Association change occurs (i.e., the contents of the Reachability Associations log page (refer to section 5.1.12.1.26) changed). If this bit is set to '1', then the Reachability Association Change Notices event is sent to the host when this condition occurs. If this bit is cleared to '0', then the controller shall not send the Reachability Association Change Notices event to the host.
16	<b>Temperature Threshold Hysteresis Recovery (TTHRY):</b> This bit determines whether an asynchronous event notification is sent to the host at the end of a temperature threshold hysteresis event (refer to section 5.1.25.1.3.1). If this bit is set to '1', then the Temperature Threshold Hysteresis Recovery event is sent to the host if an outstanding Asynchronous Event Request command exists at the time this condition occurs. If this bit is cleared to '0', then the controller shall not send the Temperature Threshold Hysteresis Recovery event to the host.

**Figure 391: Asynchronous Event Configuration – Command Dword 11**

Bits	Description
15	<b>Normal NVM Subsystem Shutdown (NNSSDN):</b> This bit determines whether an asynchronous event notification is sent to the host when the NVM subsystem has started performing a normal shutdown due to an NVM Subsystem Shutdown (refer to Figure 153). If this bit is set to '1', then the Normal NVM Subsystem Shutdown event is sent to the host if an outstanding Asynchronous Event Request command exists at the time this condition occurs. If this bit is cleared to '0', then the controller shall not send the Normal NVM Subsystem Shutdown event to the host.
14	<b>Endurance Group Event Aggregate Log Change Notices (EGEALCN):</b> This bit determines whether an asynchronous event notification is sent to the host when an event entry for an Endurance Group (refer to section 3.2.3) has been added to the Endurance Group Event Aggregate log (refer to section 5.1.12.1.15). If this bit is set to '1', then the Endurance Group Event Aggregate Log Change event is sent to the host when this condition occurs. If this bit is cleared to '0', then the controller shall not send the Endurance Group Event Aggregate Log Change event to the host.  If Endurance Groups are not supported and this bit is set to '1', then the Set Features command shall be aborted with a status of Invalid Field in Command.
13	<b>LBA Status Information Alert Notices<sup>1</sup> (LSIAN):</b> I/O Command Set specific definition.
12	<b>Predictable Latency Event Aggregate Log Change Notices (PLEALCN):</b> This bit determines whether an asynchronous event notification is sent to the host when an event pending entry for an NVM Set (refer to section 5.1.12.1.12) has been added to the Predictable Latency Event Aggregate Log. If this bit is set to '1', then the Predictable Latency Event Aggregate Log Change event is sent to the host when this condition occurs. If this bit is cleared to '0', then the controller shall not send the Predictable Latency Event Aggregate Log Change event to the host.
11	<b>Asymmetric Namespace Access Change Notices (ANACN):</b> This bit determines whether an asynchronous event notification is sent to the host when an asymmetric namespace access change occurs (i.e., the contents of the Asymmetric Namespace Access log page (refer to section 5.1.12.1.13) change). If this bit is set to '1', then the Asymmetric Namespace Access Change Notices event is sent to the host when this condition occurs. If this bit is cleared to '0', then the controller shall not send the Asymmetric Namespace Access Change Notices event to the host.
10	<b>Telemetry Log Notices (TLN):</b> This bit determines whether an asynchronous event notification is sent to the host when the Telemetry Controller-Initiated Data Available field transitions from 0h to 1h in the Telemetry Controller-Initiated log page. If this bit is set to '1', then the Telemetry Log Changed event is sent to the host when this condition occurs. If this bit is cleared to '0', then the controller shall not send the Telemetry Log Changed event to the host.
09	<b>Firmware Activation Notices (FAN):</b> This bit determines whether an asynchronous event notification is sent to the host for a Firmware Activation Starting event (refer to Figure 151). If this bit is set to '1', then the Firmware Activation Starting event is sent to the host when this condition occurs. If this bit is cleared to '0', then the controller shall not send the Firmware Activation Starting event to the host.
08	<b>Attached Namespace Attribute Notices (NAN):</b> This bit determines whether an asynchronous event notification is sent to the host for an Attached Namespace Attribute Changed asynchronous event (refer to Figure 151). If this bit is set to '1', then the Attached Namespace Attribute Changed asynchronous event is sent to the host when this condition occurs. If this bit is cleared to '0', then the controller shall not send the Attached Namespace Attribute Changed asynchronous event to the host.
07:00	<b>SMART / Health Critical Warnings (SHCW):</b> This field determines whether an asynchronous event notification is sent to the host for the corresponding Critical Warning specified in the SMART / Health Information log (refer to Figure 206). If a bit is set to '1', then an asynchronous event notification is sent when the corresponding critical warning bit is set to '1' in the SMART / Health Information log. If a bit is cleared to '0', then an asynchronous event notification is not sent when the corresponding critical warning bit is set to '1' in the SMART / Health Information log.
Notes: 1. Refer to the NVM Command Set Specification. 2. Refer to the Zoned Namespace Command Set Specification.	

**5.1.25.1.6 Autonomous Power State Transition (Feature Identifier 0Ch)**

This Feature configures the settings for autonomous controller power state transitions, refer to section 8.1.17.2.

The Autonomous Power State Transition uses Command Dword 11 and specifies the attribute information in the data structure indicated in Figure 392 and the Autonomous Power State Transition data structure consisting of 32 of the entries defined in Figure 393.

If a Get Features command is issued for this Feature, the attributes specified in Figure 392 are returned in Dword 0 of the completion queue entry and the Autonomous Power State Transition data structure, whose entry structure is defined in Figure 393, is returned in the data buffer for that command.

**Figure 392: Autonomous Power State Transition – Command Dword 11**

Bits	Description
31:01	Reserved
00	<b>Autonomous Power State Transition Enable (APSTE):</b> This bit specifies whether autonomous power state transition is enabled. If this bit is set to '1', then autonomous power state transitions are enabled. If this bit is cleared to '0', then autonomous power state transitions are disabled. This bit is cleared to '0' by default.

Each entry in the Autonomous Power State Transition data structure is defined in Figure 393. Each entry is 64 bits in size. There is an entry for each of the allowable 32 power states. For power states that are not supported, the unused Autonomous Power State Transition data structure entries shall be cleared to all zeroes. The entries begin with power state 0 and then increase sequentially (i.e., power state 0 is described in bytes 7:0, power state 1 is described in bytes 15:8, etc.). The data structure is 256 bytes in size and shall be physically contiguous.

**Figure 393: Autonomous Power State Transition – Data Structure Entry**

Bits	Description
63:32	Reserved
31:08	<b>Idle Time Prior to Transition (ITPT):</b> This field specifies the amount of idle time that occurs in this power state prior to transitioning to the Idle Transition Power State. The time is specified in milliseconds. A value of 0h disables the autonomous power state transition feature for this power state.
07:03	<b>Idle Transition Power State (ITPS):</b> This field specifies the power state to which the controller autonomously transitions, after there is a continuous period of idle time in the current power state that exceeds the time specified in the Idle Time Prior to Transition (ITPT) field. If the ITPT field is set to a non-zero value, then the state specified in this field shall be a non-operational state as described in Figure 313. This field should not specify a power state with higher reported idle power than the current power state. If the ITPT field is cleared to 0h, then this field should be cleared to 0h.
02:00	Reserved

The Autonomous Power State Transition feature may interact with the Non-Operational Power State Config feature (refer to section 5.1.25.1.10). Figure 394 shows these interactions.

**Figure 394: Interactions between APSTE and NOPPME**

APSTE <sup>1</sup>	NOPPME <sup>2</sup>	Non-operational power state entry	Background operations during non-operational power states
1	1	Entered by host request <sup>3</sup> or by ITPT idle timer <sup>4</sup>	Allowed
0	1	Entered by host request <sup>3</sup>	Allowed
1	0	Entered by host request <sup>3</sup> or by ITPT idle timer <sup>4</sup>	Not allowed
0	0	Entered by host request <sup>3</sup>	Not allowed
Notes:			
1. Defined in Figure 392.			
2. Defined in Figure 399.			
3. Refer to section 5.1.25.1.2.			
4. Refer to Figure 393.			

### 5.1.25.1.7 Timestamp (Feature Identifier 0Eh)

This Feature enables the host to set a timestamp value in the controller. A controller indicates support for the Timestamp feature through the Optional NVM Command Support (ONCS) field in the Identify Controller data structure. The Timestamp field value (refer to Figure 395) in a Set Features command sets a timestamp value in the controller. After the current value for this Feature is set, the controller updates that value as time passes. A Get Features command that requests the current value reports the timestamp value in the controller at the time the Get Features command is processed (e.g., the value set with a Set Features command for the current value plus the elapsed time since being set).

Note: If the Timestamp feature is saveable (refer to Figure 195) and the host saves a value, then the timestamp value restored after a subsequent power on or reset event is the value that was saved (refer to section 4.4). As a result, the timestamp may appear to move backwards in time.

The accuracy of a Timestamp value after initialization may be affected by vendor specific factors, such as whether the controller continuously counts after the timestamp is initialized, or whether the controller stops counting during certain intervals (e.g., non-operational power states). If the controller stops counting during such intervals, then the Synch bit in the Timestamp – Data Structure for Get Features (refer to Figure 396) shall be set to '1'.

If the controller maintains (i.e., continues to update) the timestamp value across any type of Controller Level Reset (e.g., across a Controller Reset), then the controller shall also preserve the Timestamp Origin field (refer to Figure 396) across that type of Controller Level Reset.

If the controller does not maintain the value of the timestamp across the most recent Controller Level Reset, then the Timestamp field is cleared to 0h due to that Controller Level Reset.

Timestamp values should not be used for security applications. Other application use of the Timestamp feature is outside the scope of this specification.

If a Set Features command is issued for this Feature, the data structure specified in Figure 395 is transferred in the data buffer for that command, specifying the Timestamp value.

**Figure 395: Timestamp – Data Structure for Set Features**

Bytes	Description
05:00	<b>Timestamp (TSTMP):</b> Number of milliseconds that have elapsed since midnight, 01-Jan-1970, UTC. Refer to ISO 8601 for format requirements.
07:06	Reserved

If a Get Features command is issued for this Feature, the data structure specified in Figure 396 is returned in the data buffer for that command.

**Figure 396: Timestamp – Data Structure for Get Features**

Bytes	Description
05:00	<p><b>Timestamp (TSTMP):</b></p> <p>If the Timestamp Origin field cleared to 000b, then this field is set to the time in milliseconds since the last Controller Level Reset.</p> <p>If the Timestamp Origin field is set to 001b, then this field is set to the last Timestamp value set by the host, plus the time in milliseconds since the Timestamp was set. If the sum of the Timestamp value set by the host and the elapsed time exceeds <math>2^{48}</math>, the value returned should be reduced modulo <math>2^{48}</math>.</p> <p>If the Synch bit is set to '1', then the Timestamp value may be reduced by vendor specific time intervals not counted by the controller.</p>

**Figure 396: Timestamp – Data Structure for Get Features**

Bytes	Description								
06	<b>Timestamp Attribute (TSTMPS):</b> This field indicates attributes associated with the timestamp.								
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>07:04</td> <td>Reserved</td> </tr> </tbody> </table>	Bits	Description	07:04	Reserved				
	Bits	Description							
	07:04	Reserved							
	03:01	<b>Timestamp Origin (TSTMPO):</b>							
		<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>The Timestamp field was initialized to 0h by a Controller Level Reset.</td> </tr> <tr> <td>001b</td> <td>The Timestamp field was initialized with a Timestamp value using a Set Features command.</td> </tr> <tr> <td>010b to 111b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	000b	The Timestamp field was initialized to 0h by a Controller Level Reset.	001b	The Timestamp field was initialized with a Timestamp value using a Set Features command.	010b to 111b
Value		Definition							
000b		The Timestamp field was initialized to 0h by a Controller Level Reset.							
001b	The Timestamp field was initialized with a Timestamp value using a Set Features command.								
010b to 111b	Reserved								
00	<b>Synch (SYNC):</b>								
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0b</td> <td>The controller counted time in milliseconds continuously since the Timestamp value was initialized.</td> </tr> <tr> <td>1b</td> <td>The controller may have stopped counting during vendor specific intervals after the Timestamp value was initialized (e.g., non-operational power states).</td> </tr> </tbody> </table>	Value	Definition	0b	The controller counted time in milliseconds continuously since the Timestamp value was initialized.	1b	The controller may have stopped counting during vendor specific intervals after the Timestamp value was initialized (e.g., non-operational power states).		
	Value	Definition							
0b	The controller counted time in milliseconds continuously since the Timestamp value was initialized.								
1b	The controller may have stopped counting during vendor specific intervals after the Timestamp value was initialized (e.g., non-operational power states).								
07	Reserved								

#### 5.1.25.1.8 Keep Alive Timer (Feature Identifier 0Fh)

This Feature configures the controller Keep Alive Timer. Refer to section 3.9 for Keep Alive details. The attributes are specified in Command Dword 11.

If a Get Features command is submitted for this Feature, the attributes specified in Figure 397 are returned in Dword 0 of the completion queue entry for that command.

**Figure 397: Keep Alive Timer – Command Dword 11**

Bits	Description
31:00	<p><b>Keep Alive Timeout (KATO):</b> This field specifies the timeout value for the Keep Alive Timer feature in milliseconds. The controller rounds up the value specified to the granularity indicated in the KAS field in the Identify Controller data structure. If this field is cleared to 0h, then the Keep Alive Timer is disabled.</p> <p>The default value for this field is 0h for NVMe transports that do not require use of the Keep Alive Timer feature (e.g., NVMe over PCIe). For NVMe transports that require use of the Keep Alive Timer feature (e.g., RDMA and TCP), the default value for this field is 1D4C0h (i.e., 120,000 milliseconds or 2 minutes) rounded up to the granularity indicated in the KAS field.</p> <p>Refer to the applicable NVMe Transport Binding specification for details.</p>

#### 5.1.25.1.9 Host Controlled Thermal Management (Feature Identifier 10h)

This Feature configures the controller settings for the host controlled thermal management feature, refer to section 8.1.17.5. The host controlled thermal management feature uses Command Dword 11 with the attributes shown in Figure 398.

If a Get Features command is submitted for this Feature, then the attributes shown in Figure 398 are returned in Dword 0 of the completion queue entry for that command.

**Figure 398: HCTM – Command Dword 11**

Bits	Description
31:16	<p><b>Thermal Management Temperature 1 (TMT1):</b> This field specifies the temperature, in Kelvins, when the controller begins to transition to lower power active power states or performs vendor specific thermal management actions while minimizing the impact on performance (e.g., light throttling) in order to attempt to reduce the Composite Temperature.</p> <p>A value cleared to 0h, specifies that this part of the Feature shall be disabled.</p> <p>The range of values that are supported by the controller are indicated in the Minimum Thermal Management Temperature field and Maximum Thermal Management Temperature field in the Identify Controller data structure in Figure 312.</p> <p>If the host attempts to set this field to a value less than the value contained in the Minimum Thermal Management Temperature field or greater than the value contained in the Maximum Thermal Management Temperature field in the Identify Controller data structure in Figure 312, then the command shall abort with a status code of Invalid Field in Command.</p> <p>If the host attempts to set this field to a value greater than or equal to the value contained in the Thermal Management Temperature 2 field, if non-zero, then the command shall abort with a status code of Invalid Field in Command.</p>
15:00	<p><b>Thermal Management Temperature 2 (TMT2):</b> This field specifies the temperature, in Kelvins, when the controller begins to transition to lower power active power states or perform vendor specific thermal management actions regardless of the impact on performance (e.g., heavy throttling) in order to attempt to reduce the Composite Temperature.</p> <p>A value cleared to 0h, specifies that this part of the Feature shall be disabled.</p> <p>The range of values that are supported by the controller are indicated in the Minimum Thermal Management Temperature field and Maximum Thermal Management Temperature field in the Identify Controller data structure in Figure 312.</p> <p>If the host attempts to set this field to a value less than the value contained in the Minimum Thermal Management Temperature field or greater than the value contained in the Maximum Thermal Management Temperature field in the Identify Controller data structure in Figure 312, then the command shall abort with a status code of Invalid Field in Command.</p> <p>If the host attempts to set this field to a non-zero value less than or equal to the value contained in the Thermal Management Temperature 1 field, then the command shall abort with a status code of Invalid Field in Command.</p>

#### 5.1.25.1.10 Non-Operational Power State Config (Feature Identifier 11h)

This Feature configures non-operational power state settings for the controller. The settings are specified in Command Dword 11.

If a Get Features command is submitted for this Feature, the values in Figure 399 are returned in Dword 0 of the completion queue entry for that command.

**Figure 399: Non-Operational Power State Config – Command Dword 11**

Bits	Description
31:01	Reserved



**Figure 399: Non-Operational Power State Config – Command Dword 11**

Bits	Description
00	<p><b>Non-Operational Power State Permissive Mode Enable (NOPPME):</b> If this bit is set to '1', then the controller may temporarily exceed the power limits of any non-operational power state, up to the limits of the last operational power state, to run controller-initiated background operations in that state (i.e., Non-Operational Power State Permissive Mode is enabled). If this bit is cleared to '0', then the controller shall not exceed the limits of any non-operational state while running controller-initiated background operations in that state (i.e., Non-Operational Power State Permissive Mode is disabled).</p> <p>If Non-Operational Power State Permissive Mode is disabled, then:</p> <ul style="list-style-type: none"> <li>a) thermal management that requires power (e.g., cooling fans) may be disabled; and</li> <li>b) performance after resuming from the non-operational power state may be degraded until background activity that was not allowed while in that non-operational power state has completed.</li> </ul> <p>If the host attempts to set this bit to '1' and the controller does not support Non-Operational Power State Permissive Mode as indicated in the Controller Attributes (CTRATT) field of the Identify Controller data structure, then the controller shall abort the command with a status code of Invalid Field in Command.</p>

The Non-Operational Power State Config feature may interact with the Autonomous Power State Transition feature (refer to section 5.1.25.1.6). Figure 394 shows these interactions.

#### 5.1.25.1.11 Read Recovery Level Config (Feature Identifier 12h)

This Feature is used to configure the Read Recovery Level (refer to section 8.1.20). The attributes are specified in Command Dword 11 and Command Dword 12. Modifying the Read Recovery Level has no effect on the data contained in any associated namespace.

The scope of this Feature is:

- NVM Set if NVM Sets are supported; or
- NVM Subsystem if NVM Sets are not supported.

If a Get Features command is submitted for this Feature, the attributes specified in Figure 401 are returned in Dword 0 of the completion queue entry for that command.

**Figure 400: Read Recovery Level Config – Command Dword 11**

Bits	Description
31:16	Reserved
15:00	<b>NVM Set Identifier (NVMSETID):</b> This field specifies the NVM Set to be modified. If NVM Sets are not supported, then this field is ignored and the command applies to all namespaces in the NVM subsystem.

**Figure 401: Read Recovery Level Config – Command Dword 12**

Bits	Description
31:04	Reserved
03:00	<b>Read Recovery Level (RRL):</b> This field sets the Read Recovery Level for the NVM Set specified.

#### 5.1.25.1.12 Predictable Latency Mode Config (Feature Identifier 13h)

This Feature configures an NVM Set to use Predictable Latency Mode, including warning event thresholds. Predictable Latency Mode and events are disabled by default. The attributes are specified in Command Dword 11, Command Dword 12, and the Deterministic Threshold Configuration data structure.

The NVM Set has transitioned to Predictable Latency Mode when the controller completes a Set Features command successfully with the Predictable Latency Enable bit in Command Dword 12 set to '1'. A transition to the Predictable Latency Mode may be delayed (i.e., the Set Features command completion is delayed) if the NVM subsystem needs to perform background operations on the NVM in order to operate in

Predictable Latency Mode. Upon successful completion of this command, the controller shall be in the Non-Deterministic Window.

If a Get Features command is submitted for this Feature, the attributes specified in Figure 403 are returned in Dword 0 of the completion queue entry for that command and the Deterministic Threshold Configuration data structure is returned.

**Figure 402: Predictable Latency Mode Config – Command Dword 11**

Bits	Description
31:16	Reserved
15:00	<b>NVM Set Identifier (NVMSETID):</b> This field specifies the NVM Set to be modified.

**Figure 403: Predictable Latency Mode Config – Command Dword 12**

Bits	Description
31:01	Reserved
00	<b>Predictable Latency Enable (LPE):</b> If this bit is set to '1', then Predictable Latency Mode (refer to section 8.1.18) is enabled for the NVM Set specified. If this bit is cleared to '0', then Predictable Latency Mode is disabled for the NVM Set specified.

Predictable Latency Events (refer to section 5.1.12.1.12) are configured as described in Figure 404.

**Figure 404: Predictable Latency Mode – Deterministic Threshold Configuration Data Structure**

Bytes	Description														
01:00	<b>Enable Event (ENEV):</b> This field specifies whether an entry shall be added to the Predictable Latency Event Aggregate log page for the associated event. If a bit is set to '1', then an entry shall be added if the specified event occurs. If a bit is cleared to '0', then an entry shall not be added if the specified event occurs.														
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15</td> <td><b>Deterministic Excursion (DTE):</b> Autonomous transition from DTWIN to NDWIN due to Deterministic Excursion.</td> </tr> <tr> <td>14</td> <td><b>Exceeded Value (EXV):</b> Autonomous transition from DTWIN to NDWIN due to typical or maximum value exceeded.</td> </tr> <tr> <td>13:03</td> <td>Reserved</td> </tr> <tr> <td>02</td> <td><b>DTWIN Time Warning (DTTW)</b></td> </tr> <tr> <td>01</td> <td><b>DTWIN Writes Warning (DTWW)</b></td> </tr> <tr> <td>00</td> <td><b>DTWIN Reads Warning (DTRW)</b></td> </tr> </tbody> </table>	Bits	Description	15	<b>Deterministic Excursion (DTE):</b> Autonomous transition from DTWIN to NDWIN due to Deterministic Excursion.	14	<b>Exceeded Value (EXV):</b> Autonomous transition from DTWIN to NDWIN due to typical or maximum value exceeded.	13:03	Reserved	02	<b>DTWIN Time Warning (DTTW)</b>	01	<b>DTWIN Writes Warning (DTWW)</b>	00	<b>DTWIN Reads Warning (DTRW)</b>
	Bits	Description													
	15	<b>Deterministic Excursion (DTE):</b> Autonomous transition from DTWIN to NDWIN due to Deterministic Excursion.													
	14	<b>Exceeded Value (EXV):</b> Autonomous transition from DTWIN to NDWIN due to typical or maximum value exceeded.													
	13:03	Reserved													
	02	<b>DTWIN Time Warning (DTTW)</b>													
01	<b>DTWIN Writes Warning (DTWW)</b>														
00	<b>DTWIN Reads Warning (DTRW)</b>														
31:02	Reserved														
39:32	<b>DTWIN Reads Threshold (DTRT):</b> If the value of DTWIN Reads Estimate falls below this value and the DTWIN Reads Warning is enabled, then the 'DTWIN Reads Warning' event is set in the Predictable Latency Per NVM Set log page for the affected NVM Set.														
47:40	<b>DTWIN Writes Threshold (DTWT):</b> If the value of DTWIN Writes Estimate falls below this value and the DTWIN Writes Warning is enabled, then the 'DTWIN Writes Warning' event is set in the Predictable Latency Per NVM Set log page for the affected NVM Set.														
55:48	<b>DTWIN Time Threshold (DTTT):</b> If the value of DTWIN Time Estimate falls below this value and the DTWIN Time Warning is enabled, then the 'DTWIN Time Warning' event is set in the Predictable Latency Per NVM Set log page for the affected NVM Set.														
511:56	Reserved														

### 5.1.25.1.13 Predictable Latency Mode Window (Feature Identifier 14h)

This Feature is used to set the window for the specified NVM Set and its associated namespaces if the NVM Set is configured in Predictable Latency Mode (refer to section 8.1.18). The attributes are specified in Command Dword 11 and Command Dword 12. If Predictable Latency Mode is not enabled, then the controller shall abort the command with a status code of Invalid Field in Command.

The transition to the window selected is complete when the Set Features command completes successfully. A transition to the Deterministic Window may be delayed (i.e., the Set Features command completion is delayed) if the minimum time has not been spent in the Non-Deterministic Window.

If a Get Features command is submitted for this Feature, the attributes specified in Figure 406 are returned in Dword 0 of the completion queue entry for that command. If Predictable Latency Mode is not enabled, then the controller shall abort the command with a status code of Invalid Field in Command.

**Figure 405: Predictable Latency Mode Window – Command Dword 11**

Bits	Description
31:16	Reserved
15:00	<b>NVM Set Identifier (NVMSETID):</b> This field specifies the NVM Set to be modified.

**Figure 406: Predictable Latency Mode Window – Command Dword 12**

Bits	Description		
31:03	Reserved		
02:00	<b>Window Select (WSEL):</b> This field selects or indicates the window used by all namespaces in the NVM Set.		
		<b>Value</b>	<b>Definition</b>
		000b	Reserved
		001b	Deterministic Window (DTWIN)
		010b	Non-Deterministic Window (NDWIN)
	011b to 111b	Reserved	

#### 5.1.25.1.14 Host Behavior Support (Feature Identifier 16h)

This Feature enables use of controller functionality that is associated with and depends upon specific host behavior that may or may not be supported by all hosts. A controller does not use such functionality unless the host has indicated that the host supports the specific host behavior upon which the functionality depends. The host indicates that support to the controller by setting a field in this Feature. That host action enables controller use of the associated functionality with that host. A controller shall not use functionality with a host that has not indicated support for the associated specific host behavior upon which that controller functionality depends. The attributes in Figure 407 are transferred in the data buffer.

For example, the Command Interrupted status code is associated with and depends upon the specific host behavior that the host is expected to retry commands that are aborted with that status code. That command retry behavior may or may not be supported by all hosts (e.g., hosts compliant with versions 1.3 and earlier of the NVM Express Base Specification are unlikely to retry commands aborted with the Command Interrupted status code as that status code was introduced after NVM Express Base Specification, Revision 1.3). A host that supports that command retry behavior indicates its support to the controller by setting a field to 1h in the Host Behavior Support Feature. Setting that field to 1h enables controller use of the Command Interrupted status code, with the result that this status code is used only with hosts that have indicated support for the associated command retry behavior.

This Feature is not saveable (refer to Figure 195). The default value of this Feature shall be all bytes cleared to 0h.

After a successful completion of a Set Features command for this Feature, the controller may use controller-to-host functionality that depends on specific host behavior as indicated by the attributes. If multiple Set Features commands for this Feature are processed by the controller, only information from the most recent successful command is retained (i.e., subsequent commands replace information provided by previous commands).

If a Get Features command is submitted for this Feature, the attributes specified in Figure 407 are returned in the data buffer for that command.

**Figure 407: Host Behavior Support – Data Structure**

Bytes	Description
00	<p><b>Advanced Command Retry Enable (ACRE):</b> If this bit is set to 1h, then the Command Interrupted status code is enabled (refer to Figure 102) and command retry delays are enabled. The controller may use the Command Interrupted status code and may indicate a command retry delay by setting the Command Retry Delay (CRD) field to a non-zero value in the Status field of a completion queue entry, refer to Figure 100. A host that sets this field to 1h indicates host support for the command retry behaviors that are specified for both the Command Interrupted status code and non-zero values in the CRD field.</p> <p>If this bit is cleared to 0h, then both the Command Interrupted status code and command retry delays are disabled. The controller shall not use the Command Interrupted status code, and shall clear the CRD field to 0h in all CQEs.</p> <p>All values other than 0h and 1h are reserved.</p>
01	<p><b>Extended Telemetry Data Area 4 Supported (ETDAS):</b> If this bit is set to 1h, then Telemetry Host-Initiated Data Area 4 and Telemetry Controller-Initiated Data Area 4 are supported by the host. If the Data Area 4 Support (DA4S) bit is set to '1' of the Log Page Attributes field, then the controller may populate Telemetry Host-Initiated Data Area 4 (refer to section 5.1.12.1.8) and the Telemetry Controller-Initiated Data Area 4 (refer to section 5.1.12.1.9).</p> <p>If this field is cleared to 0h, then Telemetry Host-Initiated Data Area 4 and Telemetry Controller-Initiated Data Area 4 are not supported by the host.</p> <p>All values other than 0h and 1h are reserved.</p>
02	<p><b>LBA Format Extension Enable (LBAFEE):</b> I/O Command Set specific definition. Refer to the applicable I/O Command Set specification for details.</p> <p>All values other than 0h and 1h are reserved.</p>
03	<p><b>Host Dispersed Namespace Support (HDISNS):</b> If this bit is set to 1h, then dispersed namespaces are enabled. A host that sets this field to 1h specifies that globally unique namespace identifiers (refer to section 8.1.9.2) are used for identifying namespaces (e.g., multi-path I/O software does not use the NSID of a namespace as the sole method for detection of multiple paths to that namespace).</p> <p>A host that supports reservations (refer to section 8.1.22) and sets this field to 1h specifies a Host Identifier that is unique across all hosts that connect to any participating NVM subsystem.</p> <p>If this bit is cleared to 0h, then dispersed namespaces are disabled. If the participating NVM subsystem prohibits host access to dispersed namespaces (refer to section 8.1.9.4) when this field is cleared to 0h, then the controller aborts any I/O commands (refer to section 7 in this specification and the I/O Commands section in the appropriate I/O Command Set specification) or any of the Admin commands listed in Figure 615 that the host submits to dispersed namespaces with a status code of Host Dispersed Namespace Support Not Enabled, as described in section 8.1.9.4.</p> <p>All values other than 0h and 1h are reserved.</p>

**Figure 407: Host Behavior Support – Data Structure**

Bytes	Description												
05:04	<b>Copy Descriptor Formats Enable (CDFE):</b> The bits in this field that are used (i.e., are neither reserved nor unused) enable the corresponding Copy Descriptor Formats (e.g., the Copy Descriptor Format 2h Enable (CDF2E) bit enables Copy Descriptor Format 2h). Bits in this field that this specification defines as unused (i.e., bits 1:0) correspond to Copy Descriptor Formats (refer to the NVM Command Set Specification) that are always enabled if they are supported.												
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:5</td> <td>Reserved</td> </tr> <tr> <td>4</td> <td><b>Copy Descriptor Format 4h Enable (CDF4E):</b> If this bit is set to '1' and the controller supports Copy Descriptor Format 4h, then Copy Descriptor Format 4h is enabled. If this bit is cleared to '0', or the controller does not support Copy Descriptor Format 4h, then Copy Descriptor Format 4h is disabled.</td> </tr> <tr> <td>3</td> <td><b>Copy Descriptor Format 3h Enable (CDF3E):</b> If this bit is set to '1' and the controller supports Copy Descriptor Format 3h, then Copy Descriptor Format 3h is enabled. If this bit is cleared to '0', or the controller does not support Copy Descriptor Format 3h, then Copy Descriptor Format 3h is disabled.</td> </tr> <tr> <td>2</td> <td><b>Copy Descriptor Format 2h Enable (CDF2E):</b> If this bit is set to '1' and the controller supports Copy Descriptor Format 2h, then Copy Descriptor Format 2h is enabled. If this bit is cleared to '0', or the controller does not support Copy Descriptor Format 2h, then Copy Descriptor Format 2h is disabled.</td> </tr> <tr> <td>1:0</td> <td><b>Not Used (NUSED):</b> The controller shall ignore these bits when processing a Set Features command that specifies this Feature and shall clear these bits to '0' in the attributes returned in the data buffer for a Get Features command that specifies this Feature.</td> </tr> </tbody> </table>	Bits	Description	15:5	Reserved	4	<b>Copy Descriptor Format 4h Enable (CDF4E):</b> If this bit is set to '1' and the controller supports Copy Descriptor Format 4h, then Copy Descriptor Format 4h is enabled. If this bit is cleared to '0', or the controller does not support Copy Descriptor Format 4h, then Copy Descriptor Format 4h is disabled.	3	<b>Copy Descriptor Format 3h Enable (CDF3E):</b> If this bit is set to '1' and the controller supports Copy Descriptor Format 3h, then Copy Descriptor Format 3h is enabled. If this bit is cleared to '0', or the controller does not support Copy Descriptor Format 3h, then Copy Descriptor Format 3h is disabled.	2	<b>Copy Descriptor Format 2h Enable (CDF2E):</b> If this bit is set to '1' and the controller supports Copy Descriptor Format 2h, then Copy Descriptor Format 2h is enabled. If this bit is cleared to '0', or the controller does not support Copy Descriptor Format 2h, then Copy Descriptor Format 2h is disabled.	1:0	<b>Not Used (NUSED):</b> The controller shall ignore these bits when processing a Set Features command that specifies this Feature and shall clear these bits to '0' in the attributes returned in the data buffer for a Get Features command that specifies this Feature.
	Bits	Description											
	15:5	Reserved											
	4	<b>Copy Descriptor Format 4h Enable (CDF4E):</b> If this bit is set to '1' and the controller supports Copy Descriptor Format 4h, then Copy Descriptor Format 4h is enabled. If this bit is cleared to '0', or the controller does not support Copy Descriptor Format 4h, then Copy Descriptor Format 4h is disabled.											
	3	<b>Copy Descriptor Format 3h Enable (CDF3E):</b> If this bit is set to '1' and the controller supports Copy Descriptor Format 3h, then Copy Descriptor Format 3h is enabled. If this bit is cleared to '0', or the controller does not support Copy Descriptor Format 3h, then Copy Descriptor Format 3h is disabled.											
2	<b>Copy Descriptor Format 2h Enable (CDF2E):</b> If this bit is set to '1' and the controller supports Copy Descriptor Format 2h, then Copy Descriptor Format 2h is enabled. If this bit is cleared to '0', or the controller does not support Copy Descriptor Format 2h, then Copy Descriptor Format 2h is disabled.												
1:0	<b>Not Used (NUSED):</b> The controller shall ignore these bits when processing a Set Features command that specifies this Feature and shall clear these bits to '0' in the attributes returned in the data buffer for a Get Features command that specifies this Feature.												
511:06	Reserved												

**5.1.25.1.15 Sanitize Config (Feature Identifier 17h)**

This Feature controls behavior of the Sanitize command and sanitize operations. The scope of this Feature is the NVM subsystem.

The attributes are specified in Command Dword 11.

If a Get Features command is submitted for this Feature, the attributes specified in Figure 408 are returned in Dword 0 of the completion queue entry for that command.

If this Feature is not saveable (refer to Figure 195), then the default value of the NODRM attribute shall be cleared to '0' (i.e., No-Deallocate Error Response Mode).

If the capabilities of the Sanitize Config Feature Identifier are both changeable and saveable (refer to section 4.4), then the host is able to configure this Feature when initially provisioning a device.

**Figure 408: Sanitize Config – Command Dword 11**

Bits	Description
31:01	Reserved
00	<p><b>No-Deallocate Response Mode (NODRM):</b> If the No-Deallocate Inhibited bit is set to '1' in the Sanitize Capabilities field of the Identify Controller data structure (refer to Figure 312), then this bit defines the response of the controller to a Sanitize command processed with the No-Deallocate After Sanitize (NDAS) bit set to '1' (refer to Figure 372).</p> <p>If this bit is set to '1' (i.e., No-Deallocate Warning Response Mode), then the controller shall process such Sanitize commands, and if the resulting sanitize operation is completed successfully, then the SOS field shall be set to 100b (i.e., Sanitized Unexpected Deallocate) in the Sanitize Status log page (refer to Figure 291).</p> <p>If this bit is cleared to '0' (i.e., No-Deallocate Error Response Mode), then the controller shall abort such Sanitize commands with a status code of Invalid Field in Command.</p> <p>If the No-Deallocate Inhibited bit in the Sanitize Capabilities field of the Identify Controller data structure (refer to Figure 312) is cleared to '0', then this bit has no effect.</p>

### 5.1.25.1.16 Endurance Group Event Configuration (Feature Identifier 18h)

This Feature controls the events that trigger adding an Endurance Group Event Aggregate Log Change Notices event to the Endurance Group Event Aggregate log for the specified Endurance Group. This Feature may be used to disable reporting events in the case of a persistent condition (refer to section 5.1.2). If the condition for an event is true when the corresponding notice is enabled, then an event is sent to the host. The attributes are specified in Command Dword 11.

If a Get Features command is submitted for this Feature, the Endurance Group Critical Warnings field in Command Dword 11 is not used and the attributes specified in Figure 409 are returned in Dword 0 of the completion queue entry for that command.

**Figure 409: Asynchronous Event Configuration – Command Dword 11**

Bits	Description
31:24	Reserved
23:16	<b>Endurance Group Critical Warnings (EGCW):</b> This field determines whether an event entry for an Endurance Group (refer to section 3.2.3) is added to the Endurance Group Event Aggregate log page (refer to section 5.1.12.1.15) for the corresponding Critical Warning specified in the Endurance Group Information log page (refer to Figure 218). If a bit is set to '1', then an entry is added when the corresponding critical warning bit is set to '1' in the Endurance Group Information log page. If a bit is cleared to '0', then an entry is not added when the corresponding critical warning bit is set to '1' in the Endurance Group Information log page.
15:00	<b>Endurance Group Identifier (ENDGID):</b> This field indicates the Endurance Group for which asynchronous events are being configured. If this field is cleared to 0h, then the Endurance Group Critical Warnings field is not used.

If a bit is set to '1' in the Endurance Group Critical Warnings field which corresponds to a reserved bit in the Critical Warning field of the Endurance Group Information log page (refer to Figure 218), then the Set Features command shall be aborted with a status code of Invalid Field in Command.

If the Endurance Group Identifier specifies an Endurance Group that does not exist, then the Set Features or Get Features command shall be aborted with a status code of Invalid Field in Command.

### 5.1.25.1.17 I/O Command Set Profile (Feature Identifier 19h)

This Feature specifies the I/O Command Sets that may be used by the controller when all supported I/O Command Sets (110b) are selected in CC.CSS. This Feature shall be implemented if the CAP.CSS.IOCSS bit is set to '1'. When CC.CSS is set to any value other than 110b, then this Feature has no effect and the I/O Command Sets that may be used by the controller are specified by CC.CSS. If CC.CSS is set to any value other than 110b and the controller receives a Set Features command for this Feature, then this command has no effect and returns a status code of Successful Completion.

When all supported I/O Command Sets (110b) is selected in CC.CSS, the value of this Feature specifies the index of the I/O Command Set Combination in the Identify I/O Command Set data structure that is used. Refer to section 5.1.13.2.19 for more information. The Index is specified in the I/O Command Set Combination Index field of Command Dword 11 (refer to Figure 410). If any namespace attached to the controller uses an I/O Command Set that is not supported by the specified I/O Command Set combination, then the controller shall abort the command with a status code of I/O Command Set Combination Rejected. Upon successful completion of a Set Features command for this Feature, the controller transitions to using the specified I/O Command Set Combination.

**Figure 410: I/O Command Set Profile – Command Dword 11**

Bits	Description
31:09	Reserved

**Figure 410: I/O Command Set Profile – Command Dword 11**

Bits	Description
08:00	<p><b>I/O Command Set Combination Index (IOCSCI):</b> This field specifies the index of the I/O Command Set Combination that is to be used. This field is used for the Set Features command only and is ignored for the Get Features command for this Feature.</p> <p>The controller shall abort a command that specifies an index that corresponds to an I/O Command Set Combination that has a value of 0h with a status code of I/O Command Set Combination Rejected.</p>

If a Get Features command is submitted for this Feature, then the attributes described in Figure 411 are returned in Dword 0 of the completion queue entry for that command.

**Figure 411: I/O Command Set Profile – Completion Queue Entry Dword 0**

Bits	Description
31:09	Reserved
08:00	<p><b>I/O Command Set Combination Index (IOCSCI):</b> This field returns the index of the currently selected I/O Command Set Combination.</p>

#### 5.1.25.1.18 Spinup Control (Feature Identifier 1Ah)

This Feature allows the host to configure the method for initial spinup for Endurance Groups that store data on rotational media (refer to section 8.1.23).

If the NVM subsystem does not contain any Endurance Groups that store data on rotational media, then the controller shall abort the Set Features command and the Get Features command for this Feature with status code of Invalid Field in Command.

The method is specified in Command Dword 11 (refer to Figure 412).

**Figure 412: Spinup Control – Command Dword 11**

Bits	Description
31:01	Reserved
0	<p><b>Spinup Control Enable (SCE):</b> If this bit is set to '1', then the Spinup Control feature is enabled. If this bit is cleared to '0', then the Spinup Control feature is disabled. The setting is persistent.</p>

If a Get Features command is submitted for this Feature, the attributes described in (refer to Figure 413) are returned in Dword 0 of the completion queue entry for that command.

**Figure 413: Completion Queue Entry Dword 0**

Bits	Description
31:01	Reserved
0	<p><b>Spinup Control State (SCS):</b> If this bit is set to '1', then the Spinup Control feature is enabled. If this bit is cleared to '0', then the Spinup Control feature is disabled.</p>

#### 5.1.25.1.19 Power Loss Signaling Config (Feature Identifier 1Bh)

This Feature configures the behavior of Power Loss Signaling (refer to section 8.2.5). The scope of this Feature is as described in Figure 385. The attributes are specified in Command Dword 11 (refer to Figure 414).

If a Get Features command is successfully completed for this Feature, then the attribute described in Figure 414 is returned in Dword 0 of the completion queue entry for that command.

If a Set Features command is submitted for this Feature and the Power Loss Signaling Mode field specifies a Power Loss Signaling mode that is not supported (refer to the Power Loss Signaling Information field in Figure 312), then that command shall be aborted with a status code of Invalid Field in Command.

If a Set Features command is processed while the controller is in the FQ Processing state, then that command is aborted as described in section 8.2.5.2.

**Figure 414: Power Loss Signaling Config – Command Dword 11**

Bits	Description		
31:02	Reserved		
01:00	<b>Power Loss Signaling Mode (PLSM):</b> Specifies the Power Loss Signaling mode of operation.		
		<b>Value</b>	<b>Definition</b>
		00b	Power Loss Signaling not enabled
		01b	Power Loss Signaling with Emergency Power Fail enabled
		10b	Power Loss Signaling with Forced Quiescence enabled
11b	Reserved		

#### 5.1.25.1.20 Flexible Data Placement (Feature Identifier 1Dh)

This Feature controls operation of the Flexible Data Placement capability (refer to section 8.1.10) in the specified Endurance Group (refer to Figure 415).

The attribute is specified in Command Dword 12. Effects of enabling and disabling the Flexible Data Placement are described in section 8.1.10.

If a Get Features command specifying this Feature is successfully completed, then the attributes described in Figure 416 are returned in Dword 0 of the completion queue entry for that command.

**Figure 415: Flexible Data Placement – Command Dword 11**

Bits	Description
31:16	Reserved
15:00	<b>Endurance Group Identifier (ENDGID):</b> If Endurance Groups are supported, then this field specifies the identifier for the Endurance Group (refer to section 3.2.3) used for this Feature.

**Figure 416: Flexible Data Placement – Command Dword 12**

Bits	Description
31:16	Reserved
15:08	<b>Flexible Data Placement Configuration Index (FDPCIDX):</b> This field specifies the index into the FDP Configuration Descriptor List (refer to Figure 279) identifying a valid FDP configuration to be applied to the Endurance Group when Flexible Data Placement is enabled (refer to section 8.1.10).
07:01	Reserved
00	<b>Flexible Data Placement Enable (FDPE):</b> This bit specifies if the Flexible Data Placement (i.e., capability) is enabled or disabled. If this bit is set to '1', then Flexible Data Placement is enabled. If this bit is cleared to '0', then Flexible Data Placement is disabled.

This Feature shall be saveable (refer to section 4.4). The default value of this Feature shall be 0h.

The value of this Feature is only allowed to change if the SV bit is set to '1' in the Set Features command. Therefore, if the Save and Select Feature Support (SSFS) bit is set to '1' in the ONCS field in the Identify Controller data structure (refer to Figure 312) and the SV bit is cleared to '0', then the controller shall abort the command with a status code of Invalid Field in Command.

If the value of this Feature changes, then the controller is allowed to modify any fields associated with specifying any User Data Format information accessible using the Identify command (i.e., any field associated with the Format Index (refer to section 1.6.2) that includes CNS values 00h, 05h, 08h, 09h, 0Ah). The host should issue Identify commands in this condition to access the values that may have been modified.



### 5.1.25.1.21 Flexible Data Placement Events (Feature Identifier 1Eh)

This Feature controls if a controller generates Flexible Data Placement (FDP) events associated with a specific Reclaim Unit Handle. The Reclaim Unit Handle is the Reclaim Unit Handle associated with the specified Placement Handle for the specified namespace. For a Set Features command, the data buffer contains the list of FDP Event Types that are to be enabled or disabled as defined in Figure 420.

The attribute is specified in Command Dword 12. Effects of enabling and disabling FDP events on the Reclaim Unit Handles are described in section 8.1.10.

If Flexible Data Placement is disabled, then the controller shall abort any Set Feature command or Get Feature command specifying this Feature with a status code of FDP Disabled.

If the NSID field is set to FFFFFFFFh, then the controller shall abort the command with a status code of Invalid Field in Command.

If a Get Features command specifying this Feature is successfully completed, then the attributes for the supported list of FDP Event Types described in Figure 421 is returned in the data buffer and Dword 0 of the completion queue entry contains the value of NOET (i.e., the number of supported FDP Event Types that are contained in the data buffer). Completion queue entry Dword 0 is defined in Figure 417. The supported list of FDP Event Types shall be listed in ascending order of FDP Event Type.

**Figure 417: FDP Events – Completion Queue Entry Dword 0**

Bits	Description
31:08	Reserved
07:00	<b>Number of FDP Event Types (NOET):</b> This field specifies the number of FDP Event Types that are contained in the data buffer (refer to Figure 420).

**Figure 418: FDP Events – Command Dword 11**

Bits	Description
31:24	Reserved
23:16	<b>Number of FDP Event Types (NOET):</b> This field specifies the number of FDP Event Types that are contained in the data buffer (refer to Figure 420).  This field is ignored by the controller on a Get Features command.
15:00	<b>Placement Handle (PHNDL):</b> This field specifies the Placement Handle associated with the Reclaim Unit Handle affected by this command.  If the specified Placement Handle is not valid for the namespace, then the controller shall abort a Set Feature command or a Get Feature command with a status code of Invalid Field in Command.

**Figure 419: Flexible Data Placement – Command Dword 12**

Bits	Description
31:01	Reserved
00	<b>FDP Event Enable (FDPEE):</b> If this bit is set to '1', then FDP events are enabled on the Reclaim Unit Handle associated with the Placement Handle. If this bit is cleared to '0', then FDP events are disabled on the Reclaim Unit Handle associated with the Placement Handle.  This field is ignored by the controller on a Get Features command.

This Feature shall be saveable (refer to section 4.4).

If:

- a Reclaim Unit Handle is shared by more than one namespace that exists in the same Endurance Group; and
- a Set Features command specifies this Feature and one of those namespaces,

then a modification to this Feature occurs to that shared Reclaim Unit Handle and the Placement Handle that is associated with the shared Reclaim Unit Handle.

**Figure 420: FDP Events – Set Feature Data Structure**

Bytes	Description
<b>FDP Event Type List</b>	
00	<b>FDP Event Type 0:</b> This field contains the first FDP Event Type (refer to Figure 289).
01	<b>FDP Event Type 1:</b> This field contains the second FDP Event Type.
...	
NOET-1	<b>FDP Event Type NOET-1:</b> This field contains the last FDP Event Type.

**Figure 421: FDP Events – Get Feature Data Structure**

Bytes	Description
<b>Supported FDP Event Type List</b>	
01:00	<b>Supported FDP Event Type 0:</b> This field contains the Supported FDP Event Descriptor for the first Supported FDP Event Type.
03:02	<b>Supported FDP Event Type 1:</b> This field contains the Supported FDP Event Descriptor for the second Supported FDP Event Type.
...	
(NOET-1)*2+1: (NOET-1)*2	<b>Supported FDP Event Type NOET-1:</b> This field contains the Supported FDP Event Descriptor for the last Supported FDP Event Type.

**Figure 422: Supported FDP Event Descriptor**

Bytes	Description
00	<b>FDP Event Type (FDPET):</b> This field contains the FDP Event Type (refer to Figure 289).
01	<b>FDP Event Type Attributes (FDPETA):</b> This field contains attributes for the FDP Event Type.
	<b>Bits</b> <b>Description</b>
	07:01
00	<b>FDP Event Enable (FDPEE):</b> If this bit is set to '1' then this FDP Event Type is enabled. If this bit is cleared to '0', then this FDP Event Type is disabled.

#### 5.1.25.1.22 Namespace Admin Label (Feature Identifier 1Fh)

The Namespace Admin Label feature provides the ability to set and get the Namespace Admin Label for a namespace. This Feature shall be saveable and therefore shall not be supported if the Save and Select Feature Support (SSFS) bit is cleared to '0' in the Optional NVM Command Support (ONCS) field of the Identify Controller data structure (refer to Figure 312). The attributes in Figure 423 are transferred in the data buffer.

The saved value and current value of this Feature shall be identical. If the SV bit is cleared to '0' in a Set Features command that specifies this Feature, the controller shall abort the command with a status code of Invalid Field in the Command. If the value of the Namespace Admin Label is changed by means outside the scope of this standard, then that change shall affect the results of any subsequent Get Features command that specifies this Feature.

The default value of this Feature is all nulls (i.e., all bytes cleared to 0h). Sanitize operations (refer to section 8.1.24) affect the values of this Feature; any successful sanitize operation shall modify this Feature by resetting both the saved value and the current value to the default value.

If a Get Features command is submitted for this Feature, the attributes specified in Figure 423 are returned in the data buffer for that command.

**Figure 423: Namespace Admin Label – Data Structure**

Bytes	Description
255:0	<b>Namespace Admin Label (NSAL):</b> This field contains the Namespace Admin Label for the namespace as a null-terminated UTF-8 string. A Namespace Admin Label is intended to assist a human administrator in identifying a namespace (e.g., based on the contents of the data stored in the namespace (e.g., “Q4 2018 Financial Records”) or the application intended to use the namespace (e.g., “CRM data”).

### 5.1.25.1.23 Controller Data Queue (Feature Identifier 21h)

This Feature allows a host to update the status of the head pointer of a Controller Data Queue (CDQ) and specify the configuration of a Controller Data Queue Tail event. The CDQ is specified by the Controller Data Queue Identifier (CDQID) field in Command Dword 11 (refer to Figure 424).

The Head Pointer field specifies the current slot of the head pointer in the queue (refer to section 8.1.6). The controller uses this value to determine if CDQ entries have been freed by the host.

If the Enable Tail Pointer Trigger (ETPT) bit is set to ‘1’, then when the slot specified by the Tail Pointer Trigger (TPT) field for the specified CDQ is posted with an entry, the controller shall generate a Controller Data Queue Tail Pointer event to the host (refer to section 8.1.6).

If the Set Features command is successful and there is a pending Controller Data Queue Tail Pointer event for the specified Controller Data Queue specified by the CDQID field, then the controller shall clear that pending event.

A controller should report Controller Data Queue Tail Pointer events in the order of occurrence to avoid reporting the same Controller Data Queue when a Controller Data Queue Tail Pointer event for that Controller Data Queue is being repeatedly triggered.

If the Enable Tail Pointer Trigger (ETPT) bit is set to ‘1’ in the current value of this Feature (refer to section 4.4) and the controller processes a Set Features command for this Feature that specifies:

- the Enable Tail Pointer Trigger (ETPT) bit set to ‘1’; and
- a slot in the Tail Pointer Trigger (TPT) field,

then prior to posting the completion for that Set Features command, the controller may post an entry into the specified CDQ in the slot specified by the value of the TPT field that had been set prior to processing that Set Features command. To detect this condition, after receiving the completion of that Set Features command, a host should examine the CDQ to determine if the requested tail pointer trigger has already occurred. If the requested tail pointer trigger has occurred, then a subsequent Set Features command should be submitted by the host to disable the Tail Pointer Trigger event or request a different tail pointer trigger.

If a Get Features command is submitted for this Feature, the attributes described in Figure 424 are returned in Dword 0 and the attributes described in Figure 426 are returned in the data buffer for that command.

**Figure 424: Controller Data Queue – Command Dword 11**

Bits	Description
31	<b>Enable Tail Pointer Trigger (ETPT):</b> If this bit is set to ‘1’, then the controller is to generate a Controller Data Queue Tail Pointer event when that controller posts the entry into the slot specified by the TPT field for the specified CDQ. If this bit is cleared to ‘0’, then there is no request for a Controller Data Queue Tail Pointer event to be sent by the controller.  If a Controller Data Queue Tail Pointer event is generated for the specified Controller Data Queue, then the current value of this bit for this Feature shall be cleared to ‘0’ (i.e., the current value for this bit only enables a single occurrence of a Controller Data Queue Tail Pointer event for the specified Controller Data Queue).  For a Get Features command, this field shall be ignored by the controller.
30:16	Reserved
15:00	<b>Controller Data Queue Identifier (CDQID):</b> This field contains the identifier associated with the CDQ.

**Figure 425: Controller Data Queue – Command Dword 12**

Bits	Description
31:00	<b>Head Pointer (HP):</b> This field specifies the slot of the head pointer for the specified CDQ.

**Figure 426: Controller Data Queue – Command Dword 13**

Bits	Description
31:00	<p><b>Tail Pointer Trigger (TPT):</b> If the ETPT bit is set to '1', then this field specifies a slot in the CDQ that when posted with an entry causes the controller to issue a Controller Data Queue Tail Pointer event.</p> <p>For a Set Features command, if the ETPT bit is cleared to '0', then this field shall be ignored by the controller.</p> <p>For a Get Features command, this field shall be ignored by the controller.</p>

If the CDQ is empty (refer to section 3.3.1.4) and the Head Pointer field specifies a value that is not the same value as the current value, then the controller shall abort the command with a status code of Invalid Field in Command.

If the CDQ is not empty and the Head Pointer field specifies a slot not associated with an entry that was posted by the controller within the specified CDQ, then the controller shall abort the command with a status code of Invalid Field in Command.

If the Enable Tail Pointer Trigger bit is set to '1' and the Tail Pointer Trigger field specifies a slot not associated with an entry within the specified CDQ, then the controller shall abort the command with a status code of Invalid Field in Command.

If the value in the Controller Data Queue Identifier field specifies a CDQ that does not exist in the controller processing the command, then the controller shall abort the command with a status code of Invalid Controller Data Queue.

**Figure 427: Controller Data Queue – Data Structure**

Bytes	Description
03:00	<b>Head Pointer (HP):</b> This field indicates the slot of the head pointer for the specified CDQ.
07:04	<p><b>Tail Pointer Trigger (TPT):</b> If the ETPT bit is set to '1', then this field indicates the slot in the CDQ that when posted with an entry causes the controller to issue a Controller Data Queue Tail Pointer event.</p> <p>If the ETPT bit is cleared to '0', then this field shall be cleared to 0h and should be ignored by the host.</p>
511:08	Reserved

#### 5.1.25.1.24 Embedded Management Controller Address (Feature Identifier 78h)

This Feature configures a URI (refer to RFC 3986) containing the address of a management agent provided by the system (e.g., a Management Controller (refer to the NVM Express Management Interface Specification) or an enclosure manager) for management of the NVM subsystem.

If a Set Features command is issued for this Feature, the data structure specified in Figure 428 is transferred in the data buffer for that command.

If a Get Features command is issued for this Feature, the data structure specified in Figure 428 is returned in the data buffer for that command.

**Figure 428: Set Features – Command Specific Status Values**

Bytes	Description
03:00	Reserved
511:04	<b>Embedded Management Controller Address (EMCA):</b> A URI (refer to RFC 3986), containing the address of a management agent provided by the system.

The Command and Feature Lockdown log page (refer to section 5.1.12.1.20), if supported, should indicate that this Feature is supported to be prohibited from execution.

#### 5.1.25.1.25 Host Management Agent Address (Feature Identifier 79h)

This Feature configures a URI (refer to RFC 3986) containing the address of a management agent provided by host software for management of the NVM subsystem.

If a Set Features command is issued for this Feature, the data structure specified in Figure 429 is transferred in the data buffer for that command.

If a Get Features command is issued for this Feature, the data structure specified in Figure 429 is returned in the data buffer for that command.

**Figure 429: Host Management Agent Address**

Bytes	Description
03:00	Reserved
511:04	<b>Host Management Agent Address (HMAA):</b> A URI (refer to RFC 3986), containing the address of a management agent residing in host software.

The Command and Feature Lockdown log page (refer to section 5.1.12.1.20), if supported, should indicate that this Feature is supported to be prohibited from execution.

#### 5.1.25.1.26 Host Metadata (Feature Identifier 7Dh), (Feature Identifier 7Eh), (Feature Identifier 7Fh)

The Host Metadata features are the Enhanced Controller Metadata feature (Feature Identifier 7Dh), the Controller Metadata feature (Feature Identifier 7Eh), and the Namespace Metadata feature (Feature Identifier 7Fh).

If a Get Features command specifying one of the Host Metadata features with the SEL field set to 011b (i.e., Supported Capabilities) is submitted, then the Saveable bit in Dword 0 of the corresponding completion queue entry shall be cleared to '0' (i.e., refer to section 4.4), and the Changeable bit in Dword 0 of the corresponding completion queue entry shall be set to '1'.

If a Get Features command specifying one of the Host Metadata features, the controller shall perform additional actions specified in Figure 430.

**Figure 430: Get Features – Command Dword 11**

Bits	Description
31:01	Reserved

**Figure 430: Get Features – Command Dword 11**

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

The host issues a Set Features command specifying one of the Host Metadata features containing a Host Metadata data structure (refer to Figure 432). The host receives a Host Metadata data structure via the Get Features command. The content of the strings in the Host Metadata data structure are vendor specific.

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

The Action is specified in Command Dword 11 as shown in Figure 431.

**Figure 431: Set Features – Command Dword 11**

Bits	Description
31:15	Reserved

**Figure 431: Set Features – Command Dword 11**

Bits	Description										
14:13	<p><b>Element Action (EA):</b> This field specifies the action to perform on the specified Host Metadata Feature value for each Metadata Element Descriptor data structure contained in the Host Metadata data structure.</p> <table border="1" data-bbox="699 405 1045 548" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Add or Replace Entry</td> </tr> <tr> <td>01b</td> <td>Delete Entry Multiple</td> </tr> <tr> <td>10b</td> <td>Add Entry Multiple</td> </tr> <tr> <td>11b</td> <td>Reserved</td> </tr> </tbody> </table> <p>If the Element Action field is cleared to 00b (Add/Replace Entry) and the Metadata Element Descriptor with the specified Element Type (refer to Figure 433) does not exist in the specified Host Metadata Feature value, then the Controller shall create the descriptor in the specified Host Metadata Feature value with the value in the Host Metadata data structure.</p> <p>If the Element Action field is cleared to 00b (Add/Replace Entry) and one Metadata Element Descriptor with the specified Element Type exists in the specified Host Metadata Feature value, then the Controller shall replace with the value in the specified Host Metadata data structure.</p> <p>If the Element Action field is cleared to 00b (Add/Replace Entry) and the Feature Identifier field is set to Enhanced Controller Metadata, then the controller shall abort the Set Features command with a status code of Invalid Field in Command and shall not change any Host Metadata Feature value.</p> <p>If the Element Action field is set to 01b (Delete Entry Multiple), then the Controller shall delete all the specified Metadata Element Descriptors from the specified Host Metadata Feature value, if any. If none of the specified Metadata Element Descriptors are present in the specified Host Metadata Feature value, then the controller shall complete the Set Features command with a status code of Successful Completion and shall not change any Host Metadata Feature value.</p> <p>If the Element Action field is set to 10b (Add Entry Multiple), the Feature Identifier field is set to Enhanced Controller Metadata, and no Metadata Element Descriptor with the specified Element Type exists in the Enhanced Controller Metadata Feature value, then the controller shall create new Metadata Element Descriptors in the Enhanced Controller Metadata Feature value with the Element Type and the value specified in the Host Metadata data structure.</p> <p>If the Element Action field is set to 10b (Add Entry Multiple), the Feature Identifier field is set to Enhanced Controller Metadata, and one or more Metadata Element Descriptors with the specified Element Type exists in the Enhanced Controller Metadata Feature value, then the controller shall add the specified Metadata Element to the Enhanced Controller Metadata Feature value and shall not modify any existing Metadata Element Descriptors.</p> <p>If the Element Action field is set to 10b (Add Entry Multiple) and the Feature Identifier field is not set to Enhanced Controller Metadata, then the controller shall abort the Set Features command with a status code of Invalid Field in Command and shall not change the Host Metadata Feature value.</p>	Value	Definition	00b	Add or Replace Entry	01b	Delete Entry Multiple	10b	Add Entry Multiple	11b	Reserved
Value	Definition										
00b	Add or Replace Entry										
01b	Delete Entry Multiple										
10b	Add Entry Multiple										
11b	Reserved										
12:00	Reserved										

Metadata Element Descriptors may be added, replaced, or deleted based on the action specified in the Element Action field. Modification of the Host Metadata Feature value shall be performed by the controller in an atomic manner.

If a Set Features command is submitted for a Host Metadata Feature, a Host Metadata data structure, defined in Figure 432, is transferred in the data buffer for the command. The Host Metadata data structure is 4 KiB in size and contains zero or more Metadata Element Descriptors. If host software attempts to add or replace a Metadata Element that causes the Host Metadata Feature value of the specified feature to grow larger than 4 KiB, then the controller shall abort the command with a status code of Invalid Field in Command.

If the host receives a Host Metadata data structure via the Get Features command, then all of the Metadata Element Descriptors present for the specified feature are added to a Host Metadata data structure (refer to

Figure 432) and returned in the data buffer for that command. The data buffer size is equal to the size of the Host Metadata data structure that is 4 KiB in size.

**Figure 432: Host Metadata Data Structure**

Bytes	Description
<b>Header</b>	
00	<b>Number of Metadata Element Descriptors (NMED):</b> This field contains the number of Metadata Element descriptors in the data structure.
01	Reserved
<b>Metadata Element Descriptor List</b>	
x:02	<b>Metadata Element Descriptor 0:</b> This field contains the first Metadata Element descriptor or 0h if there are no entries.
y:x+1	<b>Metadata Element Descriptor 1:</b> This field contains the second Metadata Element descriptor or 0h if there is only 1 entry.
...	...
4095:z	<b>Metadata Element Descriptor N:</b> This field contains the last Metadata Element descriptor or 0h if there are less than N+1 entries where N is the value of the NMED field.

If the Feature Identifier field specifies Controller Metadata or Namespace Metadata, then the Host Metadata data structure may contain at most one Metadata Element Descriptor of each Element Type. If the Feature Identifier field specifies Enhanced Controller Metadata, then a Host Metadata data structure may contain more than one Metadata Element Descriptor of each Element Type. Each Metadata Element Descriptor contains the data structure shown in Figure 433.

**Figure 433: Metadata Element Descriptor**

Bits	Description								
31 + (Element Length*8) :32	<b>Element Value (EVAL):</b> This field specifies the value for the element.								
31:16	<b>Element Length (ELEN):</b> This field specifies the length of the Element Value field in bytes. This field shall be cleared to 0h when deleting an entry (i.e., the EA field is set to 01b in Command Dword 11). This field should be non-zero when adding/updating an entry (i.e., the EA field is cleared to 00b). If this field is cleared to 0h when adding/updating an entry, then the controller behavior is undefined.								
15:12	Reserved								
11:08	<b>Element Revision (ER):</b> This field specifies the revision of this element value. Unless specified otherwise elsewhere in this specification, all Metadata Element Descriptors shall clear this field to 0h.								
07:05	Reserved								
04:00	<b>Element Type (ET):</b> This field specifies the type of metadata stored in the descriptor.								
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00h</td> <td>Reserved</td> </tr> <tr> <td>01h to 17h</td> <td>Element Types defined by this specification. Enhanced Controller Metadata Element and Controller Metadata Element types are defined in Figure 434. Namespace Metadata Element types are defined in Figure 435.</td> </tr> <tr> <td>18h to 1Fh</td> <td>Vendor Specific</td> </tr> </tbody> </table>	Value	Definition	00h	Reserved	01h to 17h	Element Types defined by this specification. Enhanced Controller Metadata Element and Controller Metadata Element types are defined in Figure 434. Namespace Metadata Element types are defined in Figure 435.	18h to 1Fh	Vendor Specific
	Value	Definition							
	00h	Reserved							
01h to 17h	Element Types defined by this specification. Enhanced Controller Metadata Element and Controller Metadata Element types are defined in Figure 434. Namespace Metadata Element types are defined in Figure 435.								
18h to 1Fh	Vendor Specific								

#### 5.1.25.1.26.1 Enhanced Controller Metadata (Feature Identifier 7Dh)

This Feature is used to store enhanced controller metadata about the host platform for later retrieval.

The metadata element types defined in Figure 434 are used by this Feature.

**Figure 434: Controller Metadata Element Types**

Value	Definition
00h	Reserved
01h	<b>Operating System Controller Name:</b> The name of the controller in the operating system as a UTF-8 string.



**Figure 434: Controller Metadata Element Types**

Value	Definition
02h	<b>Operating System Driver Name:</b> The name of the driver in the operating system as a UTF-8 string.
03h	<b>Operating System Driver Version:</b> The version of the driver in the operating system as a UTF-8 string.
04h	<b>Pre-boot Controller Name:</b> The name of the controller in the pre-boot environment as a UTF-8 string.
05h	<b>Pre-boot Driver Name:</b> The name of the driver in the pre-boot environment as a UTF-8 string.
06h	<b>Pre-boot Driver Version:</b> The version of the driver in the pre-boot environment as a UTF-8 string.
07h	<b>System Processor Model:</b> The model of the processor as a UTF-8 string.
08h	<b>Chipset Driver Name:</b> The chipset driver name as a UTF-8 string.
09h	<b>Chipset Driver Version:</b> The chipset driver version as a UTF-8 string.
0Ah	<b>Operating System Name and Build:</b> The operating system name and build as a UTF-8 string.
0Bh	<b>System Product Name:</b> The system product name as a UTF-8 string.
0Ch	<b>Firmware Version:</b> The host firmware (e.g., UEFI) version as a UTF-8 string.
0Dh	<b>Operating System Driver Filename:</b> The operating system driver filename as a UTF-8 string.
0Eh	<b>Display Driver Name:</b> The display driver name as a UTF-8 string.
0Fh	<b>Display Driver Version:</b> The display driver version as a UTF-8 string.
10h	<b>Host-Determined Failure Record:</b> A failure record (e.g., the reason the host has flagged a failure for an NVMe Storage Device (refer to the NVM Express Management Interface Specification) FRU which may be used for failure analysis) as a UTF-8 string.
11h to 17h	Reserved
18h to 1Fh	Vendor Specific

Refer to section 5.1.25.1.26 for the definitions of Command Dword 11 and the Host Metadata data structure.

The default value for the Number of Metadata Element Descriptors of the Enhanced Controller Metadata Feature shall be 0h on a Controller Level Reset.

If a Get Features command with the SEL field set to 011b (i.e., Supported Capabilities) with the Enhanced Controller Metadata Feature value is submitted, then the NS Specific bit in Dword 0 of the corresponding completion queue entry shall be cleared to '0'.

#### 5.1.25.1.26.2 Controller Metadata (Feature Identifier 7Eh)

This Feature is used to store controller metadata about the host platform for later retrieval.

The Controller Metadata Feature provides backward compatibility with Management Controllers (refer to the NVM Express Management Interface Specification) compliant with version 1.1 and earlier versions of the NVM Express Management Interface Specification.

If a controller supports both the Enhanced Controller Metadata Feature and the Controller Metadata Feature, then the Controller Metadata Feature should not be used by the host.

The metadata element types defined in Figure 434 are used by this Feature.

Refer to section 5.1.25.1.26 for the definitions of Command Dword 11 and the Host Metadata data structure.

If a Set Features command's Element Action field of Command Dword 11 is set to 10b (Add Entry Multiple), then the controller shall abort the command with a status code of Invalid Field in Command and shall not change the Host Metadata Feature value.

The default value for the Number of Metadata Element Descriptors of the Controller Metadata Feature shall be 0h on a Controller Level Reset.

If a Get Features command with the SEL field set to 011b (i.e., Supported Capabilities) with the Controller Metadata Feature value is submitted, then the NS Specific bit in Dword 0 of the corresponding completion queue entry shall be cleared to '0'.

### 5.1.25.1.26.3 Namespace Metadata (Feature Identifier 7Fh)

This Feature is used to store metadata about a namespace associated with a controller for later retrieval. This Feature is namespace specific and controller specific. The Add Entry Multiple action is prohibited for this Feature.

**Figure 435: Namespace Metadata Element Types**

Value	Definition
00h	Reserved
01h	<b>Operating System Namespace Name:</b> The name of the namespace in the operating system as a UTF-8 string.
02h	<b>Pre-boot Namespace Name:</b> The name of the namespace in the pre-boot environment as a UTF-8 string.
03h	<b>Operating System Namespace Name Qualifier 1:</b> The first qualifier of the Operating System Namespace Name as a UTF-8 string.
04h	<b>Operating System Namespace Name Qualifier 2:</b> The second qualifier of the Operating System Namespace Name as a UTF-8 string.
05h to 17h	Reserved
18h to 1Fh	Vendor Specific

Refer to section 5.1.25.1.26 for the definitions of Command Dword 11 and the Host Metadata data structure.

If a Get Features command with the SEL field set to 011b (i.e., Supported Capabilities) with the Namespace Metadata Feature value is submitted, then the NS Specific bit in Dword 0 of the corresponding completion queue entry shall be set to '1'.

### 5.1.25.1.27 Software Progress Marker (Feature Identifier 80h)

This Feature is a controller software progress marker. The software progress marker is persistent across power states. This information may be used to indicate to an OS software driver whether there have been issues with the OS successfully loading. The attributes are specified in Command Dword 11.

If a Get Features command is submitted for this Feature, the attributes specified in Figure 436 are returned in Dword 0 of the completion queue entry for that command.

**Figure 436: Software Progress Marker – Command Dword 11**

Bits	Description
31:08	Reserved
07:00	<b>Pre-boot Software Load Count (PBSLC):</b> Indicates the load count of pre-boot software. After successfully loading and initializing the controller, pre-boot software should set this field to one more than the previous value of the Pre-boot Software Load Count. If the previous value is 255, then the value should not be updated by pre-boot software (i.e., the value does not wrap to 0). OS driver software should set this field to 0h after the OS has successfully been initialized.

### 5.1.25.1.28 Host Identifier (Feature Identifier 81h)

This Feature allows the host to register a Host Identifier with the controller. The Host Identifier is used by the controller to determine whether other controllers in the NVM subsystem are associated with the same host. The Host Identifier may be used to designate host elements that access an NVM subsystem independently of each other or for reservations.

The Host Identifier is contained in the data structure indicated in Figure 438. The attributes are specified in Command Dword 11. If a Get Features command is issued for this Feature, the data structure specified in Figure 438 is returned in the data buffer for that command.

A Host Identifier value of 0h indicates that the host associated with the controller is not associated with any other controller in the NVM subsystem. For example, two controllers in an NVM subsystem that both have a Host Identifier value of 0h indicates that the controllers are associated with different hosts. NVMe over PCIe implementations may support using a Host Identifier value of 0h for the reservations feature (refer to

section 8.1.22). However, reservations and registrations associated with a Host Identifier value of 0h do not persist across a Controller Level Reset since a host that uses a Host Identifier value of 0h is treated as a different host after a Controller Level Reset.

A Set Features command should be used to change a Host Identifier value of 0h to a non-zero value before using streams (refer to section 8.1.8.3) or using reservations (refer to section 8.1.22). Information (i.e., streams or reservations) associated with a Host Identifier value of 0h retain the association to that Host Identifier if the Host Identifier value is changed and are not associated with the host that has the non-zero Host Identifier.

The NVM subsystem indicates if reservations are supported with a Host Identifier value of 0h with the RHII bit in the Controller Attributes field of the Identify Controller data structure (refer to Figure 312). The NVM subsystem indicates if streams are supported with a Host Identifier value of 0h with the SRNZID bit in the NVM Subsystem Stream Capability field of the streams directive return parameters (refer to Figure 607).

The Host Identifier feature shall not be a saveable feature.

The requirements and use of the Host Identifier feature are dependent on whether the NVMe over PCIe implementation or the NVMe over Fabrics implementation is supported. Refer to section 5.1.25.1.28.1 and section 5.1.25.1.28.2.

**Figure 437: Host Identifier – Command Dword 11**

Bits	Description
31:01	Reserved
00	<p><b>Enable Extended Host Identifier (EXHID):</b> If this bit is set to '1', then the host is requesting the use of an extended 128-bit Host Identifier. If this bit is cleared to '0', then the host is requesting the use of a 64-bit Host Identifier. NVMe over Fabrics implementations shall use an extended 128-bit Host Identifier.</p> <p>If the controller does not support a 128-bit Host Identifier as indicated in the Controller Attributes field in the Identify Controller data structure and the host sets this bit to '1', then a status code of Invalid Field in Command shall be returned.</p> <p>If the controller does not support a 64-bit Host Identifier (e.g., the device is an NVMe over Fabrics device) and the host clears this bit to '0', then a status code of Invalid Field in Command shall be returned.</p> <p>If the NVM subsystem supports a 64-bit Host Identifier, supports a 128-bit Host Identifier and detects that another controller in the NVM subsystem is already using a non-zero Host Identifier of a different size than the size requested in this command, then a status code of Host Identifier Inconsistent Format shall be returned.</p>

**Figure 438: Host Identifier – Data Structure Entry**

Bytes	Description
15:00	<p><b>Host Identifier (HOSTID):</b> This field specifies a 64-bit or 128-bit identifier that uniquely identifies the host associated with the controller within the NVM subsystem. The host provides an 8 byte or 16 byte data structure depending on the value specified in the Enable Extended Host Identifier bit. The value of the Host Identifier used by a host, the method used to select this value, and the method used to ensure uniqueness are outside the scope of this specification. Controllers in an NVM subsystem that have the same Host Identifier are assumed to be associated with the same host and have the same reservation and registration rights.</p> <p>If the current Host Identifier value is cleared to 0h, then the current Host Identifier shall be set to the value specified in this field.</p> <p>If the current Host Identifier value is set to a non-zero value, then the controller shall abort the command with a status code of Command Sequence Error.</p> <p>A Host Identifier value of 0h indicates that the host is not associated with any other controller in the NVM subsystem.</p>

### 5.1.25.1.28.1 PCIe Transport Implementations

The Host Identifier is an optional feature when implemented on a controller using a PCIe transport. The controller may support a 64-bit Host Identifier and/or an extended 128-bit Host Identifier. It is recommended that implementations support the extended 128-bit Host Identifier as indicated in the Controller Attributes field in the Identify Controller data structure.

The Host Identifier for PCIe transport implementations shall have a default value of 0h and shall not have a saved value. NVM Express Base Specification, Revision 2.0 and earlier allowed saving a Host Identifier value, if a non-zero Host Identifier value had been saved then that saved value shall be reset to the default value.

### 5.1.25.1.28.2 NVMe over Fabrics Implementations

The Host Identifier is a mandatory feature in NVMe over Fabrics implementations. The Host Identifier shall be an extended 128-bit Host Identifier.

### 5.1.25.1.29 Reservation Notification Mask (Feature Identifier 82h)

This Feature controls the masking of reservation notifications on a per namespace basis. A Reservation Notification log page is created whenever a reservation notification occurs on a namespace and the corresponding reservation notification type is not masked on that namespace by this Feature. If reservations are supported by the controller, then this Feature shall be supported. The attributes are specified in Command Dword 11.

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

If a Get Features command successfully completes for this Feature, the attributes specified in Figure 439 are returned in Dword 0 of the completion queue entry for that command.

**Figure 439: Reservation Notification Configuration – Command Dword 11**

Bits	Description
31:04	Reserved
03	<b>Mask Reservation Preempted Notification (RESPRE):</b> If this bit is set to '1', then mask the reporting of reservation preempted notification by the controller. If this bit is cleared to '0', then the notification is not masked and a Reservation Notification log page is created whenever notification occurs.
02	<b>Mask Reservation Released Notification (RESREL):</b> If this bit is set to '1', then mask the reporting of reservation released notification by the controller. If this bit is cleared to '0', then the notification is not masked and a Reservation Notification log page is created whenever the notification occurs.
01	<b>Mask Registration Preempted Notification (REGPRE):</b> If this bit is set to '1', then mask the reporting of registration preempted notification by the controller. If this bit is cleared to '0', then the notification is not masked and a Reservation Notification log page is created whenever the notification occurs.
00	Reserved

### 5.1.25.1.30 Reservation Persistence (Feature Identifier 83h)

Each namespace that supports reservations has a Persist Through Power Loss (PTPL) state that may be modified using either a Set Features command or a Reservation Register command (refer to section 7.6). The Reservation Persistence feature attributes are specified in Command Dword 11.

The PTPL state is contained in the Reservation Persistence feature that is namespace specific. A Set Features command that uses the namespace ID FFFFFFFFh modifies the PTPL state associated with all

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

This feature shall not be saveable (refer to Figure 195).

If a Get Features command successfully completes for this Feature Identifier, the attributes specified in Figure 440 are returned in Dword 0 of the completion queue entry for that command

**Figure 440: Reservation Persistence Configuration – Command Dword 11**

Bits	Description
31:01	Reserved
00	<b>Persist Through Power Loss (PTPL):</b> If this bit is set to '1', then reservations and registrants persist across a power loss. If this bit is cleared to '0', then reservations are released and registrants are cleared on a power loss.

**5.1.25.1.31 Namespace Write Protection Config (Feature Identifier 84h)**

This Feature is used by the host to configure the namespace write protection state or to determine the write protection state of a namespace. Refer to section 8.1.16 for definition and behaviors of the namespace write protection states. The settings are specified in Command Dword 11.

This Feature is not saveable (refer to Figure 195). There is no default value for this Feature; the value of the Feature after a power cycle or a Controller Level Reset is determined by the write protection state of the namespace prior to the power cycle or Controller Level Reset, except for the Write Protect Until Power Cycle write protection state (refer to section 8.1.16).

If a Get Features command is submitted for this Feature, the attributes specified in Figure 441 are returned in Dword 0 of the completion queue entry for that command.

**Figure 441: Write Protection – Command Dword 11**

Bits	Description		
31:03	Reserved		
02:00	<b>Write Protection State (WPS):</b> This field specifies the write protection state of the specified namespace.		
		<b>Value</b>	<b>Definition</b>
		000b	No Write Protect
		001b	Write Protect
		010b	Write Protect Until Power Cycle
		011b	Permanent Write Protect
100b to 111b	Reserved		

If a Set Features command attempts to change the namespace write protection state of a namespace that is in the Write Protect Until Power Cycle state or in the Permanent Write Protect state, then the controller shall abort the command with a status code of Feature Not Changeable.

If a Set Features command attempts to change the namespace write protection state of a namespace to the Write Protect Until Power Cycle state and the Write Protect Until Power Cycle Control (WPUPCC) bit of the of the Write Protection Control field is cleared to '0', then the controller shall abort the command with a status code of Feature Not Changeable.

If a Set Features command attempts to change the namespace write protection state of a namespace to the Write Protect Until Power Cycle state in a multi-domain NVM subsystem (i.e., the MDS bit is set to '1'

in the CTRATT field of the Identify Controller data structure (refer to Figure 312)), then the controller shall abort the command with a status code of Feature Not Changeable.

If a Set Features command changes the namespace to a write protected state, then the controller shall commit all volatile write cache data and metadata associated with the specified namespace to non-volatile storage media as part of transitioning to the write protected state.

#### 5.1.25.1.32 Boot Partition Write Protection Config (Feature Identifier 85h)

This Feature is used by the host to configure Boot Partition write protection states and to determine the write protection state of both Boot Partitions supported by a controller. Refer to section 8.1.3.3.1 for definition and behaviors of the Boot Partition write protection states and state transitions. The settings are specified in Command Dword 11.

This Feature is not saveable (refer to Figure 195). The default value of both the Boot Partition 0 Write Protection State field and the Boot Partition 1 Write Protection State field after a power cycle is Write Locked.

If a Get Features command is submitted for this Feature, the attributes specified in Figure 442 are returned in Dword 0 of the completion queue entry for that command. A controller shall not return a value of 000b for either the Boot Partition 0 Write Protection State field or the Boot Partition 1 Write Protection State field as a result of the Get Features command for the Boot Partition Write Protection Config feature.

If a Set Features command is submitted for this Feature with either the Boot Partition 0 Write Protection State field or the Boot Partition 1 Write Protection State field cleared to 000b, then the controller shall not change the Boot Partition write protection state for that Boot Partition as part of the Set Features command completion.

If the Boot Partition Write Protection Enable bit is set to '1' in the RPMB Device Configuration Block data structure (refer to section 8.1.21), then the controller shall return a value of 100b for both the Boot Partition 0 Write Protection State field and the Boot Partition 1 Write Protection State field as a result of the Get Features command for the Boot Partition Write Protection Config feature.

If a Set Features command is submitted for this Feature with either the Boot Partition 0 Write Protection State field or the Boot Partition 1 Write Protection State field set to 100b, then the controller shall abort the command with a status code of Invalid Field in Command.

**Figure 442: Boot Partition Write Protection Config - Command Dword 11**

Bits	Description														
31:06	Reserved														
05:03	<p><b>Boot Partition 1 Write Protection State (BP1WPS):</b> This field specifies the write protection state of Boot Partition 1.</p> <p>The default value of this field is Write Locked.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b<sup>1</sup></td> <td>Change in state not requested</td> </tr> <tr> <td>001b</td> <td>Write Unlocked</td> </tr> <tr> <td>010b</td> <td>Write Locked</td> </tr> <tr> <td>011b</td> <td>Write Locked Until Power Cycle</td> </tr> <tr> <td>100b<sup>2</sup></td> <td>Write Protection controlled by RPMB</td> </tr> <tr> <td>101b to 111b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	000b <sup>1</sup>	Change in state not requested	001b	Write Unlocked	010b	Write Locked	011b	Write Locked Until Power Cycle	100b <sup>2</sup>	Write Protection controlled by RPMB	101b to 111b	Reserved
Value	Definition														
000b <sup>1</sup>	Change in state not requested														
001b	Write Unlocked														
010b	Write Locked														
011b	Write Locked Until Power Cycle														
100b <sup>2</sup>	Write Protection controlled by RPMB														
101b to 111b	Reserved														

**Figure 442: Boot Partition Write Protection Config - Command Dword 11**

Bits	Description														
02:00	<p><b>Boot Partition 0 Write Protection State (BP0WPS):</b> This field specifies the write protection state of Boot Partition 0.</p> <p>The default value of this field is Write Locked.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b<sup>1</sup></td> <td>Change in state not requested</td> </tr> <tr> <td>001b</td> <td>Write Unlocked</td> </tr> <tr> <td>010b</td> <td>Write Locked</td> </tr> <tr> <td>011b</td> <td>Write Locked Until Power Cycle</td> </tr> <tr> <td>100b<sup>2</sup></td> <td>Write Protection controlled by RPMB</td> </tr> <tr> <td>101b to 111b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	000b <sup>1</sup>	Change in state not requested	001b	Write Unlocked	010b	Write Locked	011b	Write Locked Until Power Cycle	100b <sup>2</sup>	Write Protection controlled by RPMB	101b to 111b	Reserved
Value	Definition														
000b <sup>1</sup>	Change in state not requested														
001b	Write Unlocked														
010b	Write Locked														
011b	Write Locked Until Power Cycle														
100b <sup>2</sup>	Write Protection controlled by RPMB														
101b to 111b	Reserved														
<p>Notes:</p> <ol style="list-style-type: none"> <li>1. A value of 000b is not returned by a controller via a Get Features command. This value is only used in the Set Features command.</li> <li>2. A value of 100b is reserved for the Set Features command for both the Boot Partition 0 Write Protection State field and the Boot Partition 1 Write Protection State field. The Boot Partition write protection states transition to this state when RPMB Boot Partition Write Protection is enabled (refer to section 8.1.3.3.2 and section 8.1.21).</li> </ol>															

If a Set Features command is submitted that attempts to change the Boot Partition write protection state of a Boot Partition that is in the Write Locked Until Power Cycle state, then the controller shall abort the command with a status code of Feature Not Changeable.

If a Set Features command is submitted that attempts to change either Boot Partition write protection state from a value of 100b (i.e., Boot Partition write protection is controlled by RPMB), then the controller shall abort the command with a status code of Feature Not Changeable.

If a Set Features command attempts to change the Boot Partition write protection state of a Boot Partition shared across multiple controllers to the Write Locked Until Power cycle state in a multi-domain NVM subsystem (i.e., the MDS bit is set to '1' in the CTRATT field of the Identify Controller data structure (refer to Figure 312), then the controller shall abort the command with a status code of Feature Not Changeable.

### 5.1.25.2 Memory-Based Transport Feature Specific Information (PCIe)

#### 5.1.25.2.1 Number of Queues (Feature Identifier 07h)

This Feature indicates the number of queues that the host requests for the controller processing the command. This Feature shall only be issued during initialization prior to creation of any I/O Submission and/or Completion Queues. If a Set Features command is issued for this Feature after creation of any I/O Submission and/or I/O Completion Queues, then the Set Features command shall abort with status code of Command Sequence Error. The controller shall not change the value allocated between resets. For a Set Features command, the attributes are specified in Command Dword 11 (refer to Figure 443). For a Get Features command, Dword 11 is ignored.

If a Set Features or Get Features command is submitted for this Feature, the attributes specified in Figure 444 are returned in Dword 0 of the completion queue entry for that command.

**Figure 443: Number of Queues – Command Dword 11**

Bits	Description
31:16	<p><b>Number of I/O Completion Queues Requested (NCQR):</b> Indicates the number of I/O Completion Queues requested by software. This number does not include the Admin Completion Queue. A minimum of one queue shall be requested, reflecting that the minimum support is for one I/O Completion Queue. This is a 0's based value. The maximum value that may be specified is 65,534 (i.e., 65,535 I/O Completion Queues). If the value specified is 65,535, the controller should abort the command with a status code of Invalid Field in Command.</p>

**Figure 443: Number of Queues – Command Dword 11**

Bits	Description
15:00	<b>Number of I/O Submission Queues Requested (NSQR):</b> Indicates the number of I/O Submission Queues requested by software. This number does not include the Admin Submission Queue. A minimum of one queue shall be requested, reflecting that the minimum support is for one I/O Submission Queue. This is a 0's based value. The maximum value that may be specified is 65,534 (i.e., 65,535 I/O Submission Queues). If the value specified is 65,535, the controller should abort the command with a status code of Invalid Field in Command.

Note: The value allocated may be smaller or larger than the number of queues requested, often in virtualized implementations. The controller may not have as many queues to allocate as are requested. Alternatively, the controller may have an allocation unit of queues (e.g., power of two) and may supply more queues to host software to satisfy its allocation unit.

**Figure 444: Number of Queues – Completion Queue Entry Dword 0**

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

#### 5.1.25.2.2 Interrupt Coalescing (Feature Identifier 08h)

This Feature configures controller configures interrupt coalescing settings. The controller should signal an interrupt when either the Aggregation Time or the Aggregation Threshold conditions are met. If either the Aggregation Time or the Aggregation Threshold fields are cleared to 0h, then an interrupt may be generated (i.e., interrupt coalescing is implicitly disabled). This Feature applies only to the I/O Queues. It is recommended that interrupts for commands that complete in error are not coalesced. The settings are specified in Command Dword 11.

If the controller detects that interrupts are already being processed for this vector, then the controller may delay additional interrupts. Specifically, if the Completion Queue Head Doorbell property that is associated with a particular interrupt vector is being updated, then the controller has a positive indication that completion queue entries are already being processed. In this case, the aggregation time and/or the aggregation threshold may be reset/restarted upon the associated property write. This may result in interrupts being delayed indefinitely in certain workloads where the aggregation time or aggregation threshold is non-zero.

If a Get Features command is submitted for this Feature, the attributes specified in Figure 445 are returned in Dword 0 of the completion queue entry for that command.

This Feature is valid when the controller is configured for Pin Based, MSI, Multiple MSI or MSI-X interrupts. There is no requirement for the controller to persist these settings if interrupt modes are changed. It is recommended that the host re-issue this Feature after changing interrupt modes.

**Figure 445: Interrupt Coalescing – Command Dword 11**

Bits	Description
31:16	Reserved
15:08	<b>Aggregation Time (TIME):</b> Specifies the recommended maximum time in 100 microsecond increments that a controller may delay an interrupt due to interrupt coalescing. A value of 0h corresponds to no delay. The controller may apply this time per interrupt vector or across all interrupt vectors. The reset value of this setting is 0h.



**Figure 445: Interrupt Coalescing – Command Dword 11**

Bits	Description
07:00	<b>Aggregation Threshold (THR):</b> Specifies the recommended minimum number of completion queue entries to aggregate per interrupt vector before signaling an interrupt to the host. This is a 0's based value. The reset value of this setting is 0h.

#### 5.1.25.2.3 Interrupt Vector Configuration (Feature Identifier 09h)

This Feature configures settings specific to a particular controller interrupt vector. The settings are specified in Command Dword 11.

By default, coalescing settings are enabled for each interrupt vector. Interrupt coalescing is not supported for the Admin Completion Queue.

If a Get Features command is submitted for this Feature, the attributes specified in Figure 446 are returned in Dword 0 of the completion queue entry for that command for the Interrupt Vector specified in Command Dword 11.

Prior to issuing a Set Features command that specifies this Feature, the host shall configure the specified Interrupt Vector with an I/O Completion Queue (refer to section 5.2.1). If the specified Interrupt Vector is invalid, or not associated with an existing I/O Completion Queue (refer to Figure 475), then the controller should abort the command with a status code of Invalid Field in Command.

**Figure 446: Interrupt Vector Configuration – Command Dword 11**

Bits	Description
31:17	Reserved
16	<b>Coalescing Disable (CD):</b> If this bit is set to '1', then any interrupt coalescing settings shall not be applied for this interrupt vector. If this bit is cleared to '0', then interrupt coalescing settings apply for this interrupt vector.
15:00	<b>Interrupt Vector (IV):</b> This field specifies the interrupt vector for which the configuration settings are applied.

#### 5.1.25.2.4 Host Memory Buffer (Feature Identifier 0Dh)

This Feature controls use of the Host Memory Buffer by the controller. The attributes are specified in Command Dword 11, Command Dword 12, Command Dword 13, Command Dword 14, and Command Dword 15.

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

- a) The Host Memory Descriptor List (refer to Figure 452); and
- b) the associated host memory region (i.e., the memory regions described by the Host Memory Descriptor List),

until the host memory buffer has been disabled.

If the host memory buffer is enabled, then a Set Features command to enable the host memory buffer (i.e., the EHM bit (refer to Figure 447) set to '1') shall abort with a status code of Command Sequence Error.

If the host memory buffer is not enabled, then a Set Features command to disable the host memory buffer (i.e., the EHM bit (refer to Figure 447) cleared to '0') shall succeed without taking any action.

After a successful completion of a Set Features command that disables the host memory buffer, the controller shall not access any data in the host memory buffer until the host memory buffer has been enabled. The controller should retrieve any necessary data from the host memory buffer in use before posting the completion queue entry for the Set Features command that disables the host memory buffer. Posting of the completion queue entry for the Set Features command that disables the host memory buffer

acknowledges that it is safe for the host software to modify the host memory buffer contents. Refer to section 8.2.3.

A host is able to restrict access to the host memory buffer (HMB) while the controller is in a non-operational power state that was configured by the host (refer to section 5.1.25.1.2). If this HMB non-operational power state access restriction is enabled by the host (refer to Figure 447) and the host configures a non-operational power state, then the controller does not access the HMB until the controller transitions to an operational power state except for HMB access required to process Admin commands and background operations initiated by Admin commands. Enabling or disabling Non-Operational Power State Permissive Mode (refer to section 5.1.25.1.10) shall have no effect on HMB non-operational power state access restriction.

Enabling or disabling HMB non-operational power state access restriction should not affect the Entry Latency (ENLAT) for non-operational power states (refer to section 8.1.17) that are reported in the power state descriptors in Identify Controller data structure (e.g., if HMB non-operational power state access restriction is enabled, the controller may consume additional time beyond the applicable Entry Latency value in order to retrieve necessary data from the HMB before the controller transitions to a non-operational power state).

If HMB non-operational power state access restriction is enabled and the controller autonomously transitions from an operational power state to a non-operational power state, then HMB access by the controller is not restricted and that access should be minimized (e.g., access ceases as soon as possible after that transition and does not resume until after the controller transitions to an operational power state).

If HMB non-operational power state access restriction is enabled and the host configures a non-operational power state while the controller is in a non-operational power state, then HMB access by the controller is restricted.

If a Get Features command is issued for this Feature, then the completion queue entry indicates whether HMB non-operational power state access restriction is enabled and whether HMB non-operational power state access restriction is currently restricting controller access to the HMB (refer to Figure 454):

**Figure 447: Host Memory Buffer – Command Dword 11**

Bits	Description
31:04	Reserved
03	<b>Cleared to Zero (CTZ):</b> This bit shall be cleared to '0'
02	<p><b>Host Memory Non-operational Access Restriction Enable (HMNARE):</b> If the HMBR bit is set to '1' in the Controller Attributes (CTRATT) field in the Identify Controller data structure and:</p> <ul style="list-style-type: none"> <li>this bit is set to '1', then HMB non-operational power state access restriction shall be enabled (i.e., the controller shall not access the HMB after a non-operational power state is configured by the host (refer to section 5.1.25.1.2) until the controller is in an operational power state except for access required to process Admin commands and background operations initiated by Admin commands); and</li> <li>this bit is cleared to '0', then HMB non-operational power state access restriction shall be disabled (i.e., the controller may access the HMB while the controller is in any non-operational power state).</li> </ul> <p>If this bit set to '1' and the HMBR bit is cleared to '0', then the controller shall abort the command with a status code of Invalid Field in Command.</p> <p>If this bit is cleared to '0' and the HMBR bit is cleared to '0', then this bit has no effect.</p>
01	<p><b>Memory Return (MR):</b> If this bit is set to '1', then the host is returning memory previously allocated to the controller for use as the host memory buffer (HMB). That memory may have been in use for the HMB prior to a reset or entering the Runtime D3 state (e.g., prior to the HMB being disabled). A returned host memory buffer shall have the exact same size, descriptor list address, descriptor list contents, and host memory buffer contents as last seen by the controller before the host memory buffer was disabled (i.e., a Set Features command with the EHM bit cleared to '0' was processed). If this bit is cleared to '0', then the host is allocating host memory resources with undefined content.</p>

**Figure 447: Host Memory Buffer – Command Dword 11**

Bits	Description
00	<p><b>Enable Host Memory (EHM):</b> If this bit is set to '1', then the host memory buffer shall be enabled and the controller may use the host memory buffer. If this bit is cleared to '0', then the host memory buffer shall be disabled, and the controller shall not use the host memory buffer.</p> <p>If a Set Features command is processed with this bit cleared to '0', then the controller shall ignore Command Dword 12, Command Dword 13, Command Dword 14, and Command Dword 15.</p>

**Figure 448: Host Memory Buffer – Command Dword 12**

Bits	Description
31:00	<p><b>Host Memory Buffer Size (HSIZE):</b> This field specifies the size of the host memory buffer allocated in memory page size (CC.MPS) units.</p>

**Figure 449: Host Memory Buffer– Command Dword 13**

Bits	Description
31:00	<p><b>Host Memory Descriptor List Lower Address (HMDLLA):</b> This field specifies the least significant 32 bits of the physical location of the Host Memory Descriptor List (refer to Figure 452) for the Host Memory Buffer. This address shall be 16 byte aligned, indicated by bits 3:0 being cleared to 0h.</p> <p>Note: The controller shall operate as if bits 3:0 are cleared to 0h. However, the controller is not required to check that bits 3:0 are cleared to 0h.</p>

**Figure 450: Host Memory Buffer – Command Dword 14**

Bits	Description
31:00	<p><b>Host Memory Descriptor List Upper Address (HMDLUA):</b> This field specifies the most significant 32 bits of the physical location of the Host Memory Descriptor List for the Host Memory Buffer.</p>

The Host Memory Descriptor List Address (HMDLLA/HMDLUA) specifies the address of a physically contiguous data structure in host memory that describes the address and length pairs of the Host Memory Buffer. The number of address and length pairs is specified in the Host Memory Descriptor List Entry Count in Figure 451. The Host Memory Descriptor List is described in Figure 452.

**Figure 451: Host Memory Buffer – Command Dword 15**

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

If the host specifies the Host Memory Descriptor List Entry Count field cleared to 0h, then the controller shall abort the command with a status code of Invalid Field in Command.

**Figure 452: Host Memory Buffer – Host Memory Descriptor List**

Bytes	Description
<b>Host Memory Descriptor List</b>	
15:0	<p><b>Host Memory Buffer Descriptor Entry 0:</b> This field is the first Host Memory Buffer Descriptor (refer to Figure 453) in the list, if any.</p>
31:16	<p><b>Host Memory Buffer Descriptor Entry 1:</b> This field is the second Host Memory Buffer Descriptor in the list, if any.</p>
...	
16*n+15:16*n	<p><b>Host Memory Buffer Descriptor Entry n:</b> This field is the last Host Memory Buffer Descriptor in the list where n = HMDLEC - 1 (refer to Figure 451), if any.</p>

Each Host Memory Buffer Descriptor Entry shall describe a host memory address in memory page size units and the number of contiguous memory page size units associated with the host address.

**Figure 453: Host Memory Buffer – Host Memory Buffer Descriptor Entry**

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

If a Get Features command is issued for this Feature, the attributes specified in Figure 454 are returned in Dword 0 of the completion queue entry and the Host Memory Buffer Attributes data structure, whose structure is defined in Figure 455, is returned in the data buffer for that command.

**Figure 454: Host Memory Buffer – Completion Queue Entry Dword 0**

Bits	Description
31:04	Reserved
03	<b>Host Memory Non-operational Access Restricted (HMNAR):</b> If this bit is set to '1', then HMB non-operational power state access restriction is currently restricting the controller from accessing the HMB. If this bit is cleared to '0', then HMB non-operational power state access restriction is not currently restricting the controller from accessing the HMB (e.g., the HMB is not enabled, the controller is currently in an operational power state, HMB non-operational power state access restriction is not supported, or HMB non-operational power state access restriction is not enabled).
02	<b>Host Memory Non-operational Access Restriction Enable (HMNARE):</b> If this bit is set to '1', then HMB non-operational power state access restriction is enabled (refer to the HMNARE bit description in Figure 447). If this bit is cleared to '0', then HMB non-operational power state access restriction is not enabled.
01	<b>Not Used (NUSED):</b> This bit is not used for a Get Feature command and shall be cleared to '0'.
00	<b>Enable Host Memory (EHM):</b> If this bit is set to '1', then the host memory buffer is enabled and the controller may use the host memory buffer. If this bit is cleared to '0', then the host memory buffer is disabled, and the controller is not using the host memory buffer.

**Figure 455: Host Memory Buffer – Attributes Data Structure**

Bytes	Description
3:0	<b>Host Memory Buffer Size (HSIZE):</b> This field indicates the size of the host memory buffer allocated in memory page size units.
7:4	<b>Host Memory Descriptor List Address Lower (HMDLAL):</b> This field indicates the least significant 32 bits of the physical location of the Host Memory Descriptor List (refer to Figure 452) for the host memory buffer. This address shall be 16 byte aligned. The least significant 4 bits shall be cleared to '0'.
11:8	<b>Host Memory Descriptor List Address Upper (HMDLAU):</b> This field indicates the most significant 32 bits of the physical location of the Host Memory Descriptor List (refer to Figure 452) for the host memory buffer.
15:12	<b>Host Memory Descriptor List Entry Count (HMDLEC):</b> This field indicates the number of valid Host Memory Descriptor Entries (refer to Figure 453) in the Host Memory Descriptor List (refer to Figure 452).
4095:16	Reserved

### 5.1.25.3 Message-Based Transport Feature Specific Information (Fabrics)

There are no features that are specific to the Message-based transport model.

### 5.1.25.3.1 Asynchronous Event Configuration (Feature Identifier 0Bh) for Fabrics

The Asynchronous Event Configuration feature is not Fabrics specific. Configuration of notifications for Fabrics specific events is described in section 5.1.25.1.5.

### 5.1.25.4 Command Completion

Upon completion of the Set Features command, the controller posts a completion queue entry to the Admin Completion Queue. If a status code of Successful Completion is returned, the completion queue entry shall not be posted until the controller has completed setting attributes associated with the Feature. Set Features command specific status values are defined in Figure 456.

**Figure 456: Set Features – Command Specific Status Values**

Value	Description
0Dh	<b>Feature Identifier Not Saveable:</b> The Feature Identifier specified does not support a saveable value.
0Eh	<b>Feature Not Changeable:</b> The Feature Identifier specified does not support a changeable value or the value is not changeable at this time.
0Fh	<b>Feature Not Namespace Specific:</b> The Feature Identifier does not have namespace scope. The scope is defined in Figure 385. The Feature Identifier settings apply across all namespaces.
14h	<b>Overlapping Range:</b> Command Set specific definition. Refer to each I/O Command Set specification for applicability and details.
15h	<b>I/O Command Set Combination Rejected:</b> This error indicates that the controller did not accept the request to select the requested I/O Command Set Combination.
37h	<b>Invalid Controller Data Queue:</b> This error indicates that the specified Controller Data Queue Identifier is invalid for the controller processing the command.

### 5.1.26 Track Receive command

The Track Receive command is used to obtain the information being tracked by the controller that was enabled by a Track Send command (refer to section 5.1.27).

The Track Receive command uses the Data Pointer field, Command Dword 10 field, and Command Dword 12 field. The use of the Command Dword 11 field is specific to the management operation specified by the Select field. All other command specific fields are reserved.

The Select field defined in Figure 457 specifies the management operation to be performed. Refer to section 5.1.26.1 for a description of each management operation.

**Figure 457: Track Receive – Command Dword 10**

Bits	Description												
31:08	Reserved												
07:00	<p><b>Select (SEL):</b> This field specifies the type of management operation to perform.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>M/O<sup>1</sup></th> <th>Management Operation</th> <th>Reference Section</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>O</td> <td>Tracked Memory Changes</td> <td>5.1.26.1.1</td> </tr> <tr> <td>1h to FFh</td> <td></td> <td>Reserved</td> <td></td> </tr> </tbody> </table> <p>Notes: 1. O/M definition: O = Optional, M = Mandatory</p>	Value	M/O <sup>1</sup>	Management Operation	Reference Section	0h	O	Tracked Memory Changes	5.1.26.1.1	1h to FFh		Reserved	
Value	M/O <sup>1</sup>	Management Operation	Reference Section										
0h	O	Tracked Memory Changes	5.1.26.1.1										
1h to FFh		Reserved											

**Figure 458: Track Receive – Command Dword 12**

Bits	Description
31:00	<b>Number of Dwords (NUMDL):</b> This field specifies the number of dwords to return. If the host specifies a size larger than defined by the specified management operation of the command, then the controller returns the data specified by that management operation with undefined results for dwords beyond the end of the data specified by that management operation. This is a 0's based value.

### 5.1.26.1 Track Receive Management Operations

#### 5.1.26.1.1 Tracked Memory Changes (Management Operation 0h)

The Tracked Memory Changes management operation of the Track Receive command is used to report host memory changes by the controller specified by the Controller Identifier (CNTLID) field (refer to Figure 459) processing any and all commands (e.g., Admin commands, I/O command) as a result of tracking host memory modifications being started by a Tracked Memory Changes management operation of the Track Send command (refer to section 5.1.27.1.2).

The data returned is a Tracked Memory Change data structure defined by Figure 460. The identifier for the controller that is tracked is specified in the Controller Identifier field in the Command Dword 11 field as defined in Figure 459.

**Figure 459: Tracked Memory Changes – Command Dword 11**

Bits	Description
31:16	Reserved
15:00	<b>Controller Identifier (CNTLID):</b> This field specifies the identifier of the controller for which the tracked memory changes, if any, are to be returned.

If the Track Receive command for this management operation completes successfully, then the controller shall remove the tracked memory changes that were reported in the returned data. The reported memory ranges identify the host memory that has changed due to the specified controller processing commands since:

- Track Memory Changes was enabled (refer to 5.1.27.1.2) and no tracked memory changes have been reported by the controller; or
- the controller last reported tracking change while the memory tracking remained enabled.

If tracked memory changes occur for the same memory address at a high frequency, the controller reporting of the Tracking Memory Changed Descriptors for that address may be delayed by a vendor specific amount of time so that Tracking Memory Changed Descriptors for other addresses are able to be reported during that vendor specific amount of time.

If the Suspended bit is set to '1' and the More To Report (MTR) bit is cleared to '0' in the returned Track Memory Changed data structure, then all host memory changes have been reported to the host. If the controller specified by the CNTLID field remains suspended, then the controller has no more tracked host memory changes to report to the host until the controller specified by the CNTLID field is resumed.

If the Suspended bit is cleared to '0' and the More To Report (MTR) bit is cleared to '0' in the returned Track Memory Changed data structure, then no additional tracked host memory changes are available to report to the host in a subsequent commands at the time of processing this command.

**Figure 460: Tracked Memory Change Data Structure**

Bytes	Description
<b>Header</b>	
00	<b>Version (VER):</b> This field indicates the version of this data structure. This field shall be cleared to 0h.

**Figure 460: Tracked Memory Change Data Structure**

Bytes	Description								
01	<b>Attributes (ATTRB):</b> This field identifies attributes associated with the tracked memory changes.								
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:2</td> <td>Reserved</td> </tr> <tr> <td>1</td> <td><b>Suspended (SUSP):</b> If this bit is set to '1', then the controller was suspended during the processing of the command. If this bit is cleared to '0', then the controller was not suspended during the processing of the command.</td> </tr> <tr> <td>0</td> <td><b>More To Report (MTR):</b> If this bit is set to '1', then the controller was not able to report all of the tracked memory changes available to be reported in this returned data structure. The host should issue a subsequent Track Receive command to obtain any tracked memory changes not reported.  If this bit is cleared to '0', the controller reported all of the tracked memory changes available to be reported, if any, at the time the Track Receive command was processed.  If the NTMCD field is cleared to 0h, then this bit shall be cleared to '0'.</td> </tr> </tbody> </table>	Bits	Description	7:2	Reserved	1	<b>Suspended (SUSP):</b> If this bit is set to '1', then the controller was suspended during the processing of the command. If this bit is cleared to '0', then the controller was not suspended during the processing of the command.	0	<b>More To Report (MTR):</b> If this bit is set to '1', then the controller was not able to report all of the tracked memory changes available to be reported in this returned data structure. The host should issue a subsequent Track Receive command to obtain any tracked memory changes not reported.  If this bit is cleared to '0', the controller reported all of the tracked memory changes available to be reported, if any, at the time the Track Receive command was processed.  If the NTMCD field is cleared to 0h, then this bit shall be cleared to '0'.
	Bits	Description							
7:2	Reserved								
1	<b>Suspended (SUSP):</b> If this bit is set to '1', then the controller was suspended during the processing of the command. If this bit is cleared to '0', then the controller was not suspended during the processing of the command.								
0	<b>More To Report (MTR):</b> If this bit is set to '1', then the controller was not able to report all of the tracked memory changes available to be reported in this returned data structure. The host should issue a subsequent Track Receive command to obtain any tracked memory changes not reported.  If this bit is cleared to '0', the controller reported all of the tracked memory changes available to be reported, if any, at the time the Track Receive command was processed.  If the NTMCD field is cleared to 0h, then this bit shall be cleared to '0'.								
03:02	<b>Controller Identifier (CNTLID):</b> This field indicates the identifier for the controller whose memory modifications are being tracked and reported in the Tracked Memory Changed Descriptor list. (i.e., the same controller identifier as specified by the host in the Management Operation Specific field (refer to Figure 459)).								
07:04	<b>Number of Tracked Memory Changed Descriptors (NTMCD):</b> This field indicates the number of Tracked Memory Changed Descriptors in this data structure. A value of 0h indicates that at the time of processing the Track Receive command, there was no tracked memory changes to report.								
09:08	<b>Reported Memory Range Granularity (RPMPG):</b> This field specifies the granularity unit size of the Length field in each Tracked Memory Changed Descriptor containing in this data structure. The value specified is a power of two ( $2^n$ ) times 4 KiB. For example, a value of 8h indicates that the granularity of tracking is $(2^8) * 4$ KiB which is equal to 1 MiB.								
15:10	Reserved								
<b>Tracked Memory Changed Descriptor List</b>									
31:16	<b>Tracked Memory Changed Descriptor 1:</b> This field contains the first Tracked Memory Change Descriptor as defined in Figure 463, if any.								
47:32	<b>Tracked Memory Changed Descriptor 2:</b> This field contains the second Tracked Memory Changed Descriptor as defined in Figure 463, if any.								
...									
(NTMCD*16)+15: (NTMCD-1)*16+16	<b>Tracked Memory Changed Descriptor NTMCD:</b> This field contains the last Tracked Memory Changed Descriptor as defined in Figure 463, if any.								
...									

Figure 463 defines a Tracked Memory Changed Descriptor.

**Figure 461: Tracked Memory Changed Descriptor**

Bytes	Description
07:00	<b>Start Address (SADDR):</b> This field indicates the 64-bit starting address of a memory range in which one or more bytes of data have been written. The address in this field shall be aligned to the granularity specified by the RPMPG field. For example, if the value in the RPMPG field is 8h, then the value in this field is required to be aligned to 1 MiB (i.e., bits 19:0 are cleared to 0h).
11:08	<b>Length (LEN):</b> This field specifies the length of this memory range in units of the tracking granularity as indicated in the RPMPG field. For example, if the value in the RPMPG field is 8h and the value of this field is 4h then the length of this length of this memory range is 1 MiB * 4h which is 4 MiB.
15:12	Reserved

### 5.1.26.1.2 Command Completion

Upon completion of the Track Receive command, the controller posts a completion queue entry to the Admin Completion Queue indicating the status for the command. Section 5.1.26.1 describes completion details each for management operation.

Track Receive command specific status values (i.e., SCT field set to 1h) are shown in Figure 462.

**Figure 462: Track Receive – Command Specific Status Values**

Value	Definition
1Fh	<b>Invalid Controller Identifier:</b> The controller for the specified Controller Identifier is already tracking host memory changes.

### 5.1.27 Track Send command

The Track Send command is used to manage the tracking of information by a controller (e.g., what to track during the migration of a controller in an NVM subsystem).

The Track Send command uses the Command Dword 10 field. The use of the Data Pointer field and Command Dword 11 field is specific to the management operation specified by the Select field. All other command specific fields are reserved.

The Select field defined in Figure 463 specifies the management operation to be performed. Refer to section 5.1.27.1 for a description of each management operation.

**Figure 463: Track Send – Command Dword 10**

Bits	Description																
31:16	<b>Management Operation Specific (MOS):</b> The definition of this field is specific to a management operation (refer to the Select field). If a management operation does not define the use of this field, then this field is reserved.																
15:08	Reserved																
07:00	<p><b>Select (SEL):</b> This field specifies the type of management operation to perform.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>M/O<sup>1</sup></th> <th>Management Operation</th> <th>Reference Section</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>O</td> <td>Log User Data Changes</td> <td>5.1.27.1.1</td> </tr> <tr> <td>1h</td> <td>O</td> <td>Track Memory Changes</td> <td>5.1.27.1.2</td> </tr> <tr> <td>2h to FFh</td> <td></td> <td>Reserved</td> <td></td> </tr> </tbody> </table> <p>Notes: 1. O/M definition: O = Optional, M = Mandatory</p>	Value	M/O <sup>1</sup>	Management Operation	Reference Section	0h	O	Log User Data Changes	5.1.27.1.1	1h	O	Track Memory Changes	5.1.27.1.2	2h to FFh		Reserved	
Value	M/O <sup>1</sup>	Management Operation	Reference Section														
0h	O	Log User Data Changes	5.1.27.1.1														
1h	O	Track Memory Changes	5.1.27.1.2														
2h to FFh		Reserved															

#### 5.1.27.1 Track Send Management Operations

##### 5.1.27.1.1 Log User Data Changes (Management Operation 0h)

The Log User Data Changes management operation of the Track Send command is used to start or stop logging user data changes to namespaces attached to the controller associated with the User Data Migration Queue (refer to section 5.1.4.1.1.1) specified in Controller Data Queue Identifier field in Command Dword 11 (refer to Figure 465).

Refer to the applicable I/O command Set specification for additional requirements.



**Figure 464: Log User Data Changes – Management Operation Specific Field**

Bits	Description								
15:04	Reserved								
03:00	<b>Logging Action (LACT):</b> This field specifies the logging action to be performed.								
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td><b>Stop Logging:</b> The controller shall stop logging user data changes to namespaces attached to the controller associated with the User Data Migration Queue specified in the Controller Data Queue Identifier field.</td> </tr> <tr> <td>1h</td> <td><b>Start Logging:</b> The controller shall start logging user data changes to namespaces attached to the controller associated with the User Data Migration Queue (refer to the CDQID field in Figure 465) into that User Data Migration Queue where those user data changes are caused by the controller associated with that User Data Migration Queue processing commands.</td> </tr> <tr> <td>2h to Fh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	<b>Stop Logging:</b> The controller shall stop logging user data changes to namespaces attached to the controller associated with the User Data Migration Queue specified in the Controller Data Queue Identifier field.	1h	<b>Start Logging:</b> The controller shall start logging user data changes to namespaces attached to the controller associated with the User Data Migration Queue (refer to the CDQID field in Figure 465) into that User Data Migration Queue where those user data changes are caused by the controller associated with that User Data Migration Queue processing commands.	2h to Fh	Reserved
	Value	Definition							
	0h	<b>Stop Logging:</b> The controller shall stop logging user data changes to namespaces attached to the controller associated with the User Data Migration Queue specified in the Controller Data Queue Identifier field.							
1h	<b>Start Logging:</b> The controller shall start logging user data changes to namespaces attached to the controller associated with the User Data Migration Queue (refer to the CDQID field in Figure 465) into that User Data Migration Queue where those user data changes are caused by the controller associated with that User Data Migration Queue processing commands.								
2h to Fh	Reserved								
2h to Fh	Reserved								

The Log User Data Changes management operation uses Command Dword 11 as defined in Figure 465.

**Figure 465: Log User Data Changes – Command Dword 11**

Bits	Description
31:16	Reserved
15:00	<b>Controller Data Queue Identifier (CDQID):</b> This field specifies the Controller Data Queue Identifier of the User Data Migration Queue.

Refer to the applicable I/O Command Set specification for any requirements on the posting of entries into the User Data Migration Queue if this command is successful.

If the value in the Controller Data Queue Identifier field specifies a User Data Migration Queue that does not exist in the controller processing the command, then the controller shall abort the command with a status code of Invalid Controller Data Queue.

If the LACT field is set to 1h and any controller in the NVM subsystem is already logging user data for the controller associated with the Controller Data Queue specified by the CDQID field, then the controller shall abort the command with a status code of Invalid Controller Data Queue.

If the controller associated with the User Data Migration Queue specified by the CDQID field is suspended as a result of a Migration Send command (refer to section 5.1.17.1.1), then the controller shall abort the command with a status code of Controller Suspended.

#### 5.1.27.1.2 Track Memory Changes (Management Operation 1h)

The Track Memory Changes management operation of the Track Send command requests the controller processing that Track Send command to:

- start tracking host memory changes described by the specified host memory ranges that are caused by the processing commands in the controller specified by Controller Identifier field (refer to Figure 467); or
- stop tracking the host memory changes that are caused by the processing commands by the controller specified by Controller Identifier field.

If the TACT bit is set to '1' (refer to Figure 466), then:

- The data buffer contains a Track Memory Changes data structure (refer to Figure 468) that specifies one or more host memory ranges associated with the controller specified by the Controller Identifier field.
- If the value in the Requested Memory Range Tracking Granularity (RMRTG) field is greater than the value in the Maximum Memory Range Tracking Granularity (MAXMRTG) field in the Identify Controller data structure (refer to Figure 312), then the controller shall abort the command with a status code of Invalid Field in Command.

- If the value in the Requested Memory Range Tracking Granularity (RMRTG) field is less than the value in the Minimum Memory Range Granularity (MINMRG) field in the Identify Controller data structure (refer to Figure 312), then the controller shall abort the command with a status code of Invalid Field in Command.
- If the sum of:
  - the value in the Requested Number of Memory Range Tracking Descriptors (RNMRTD) field; and
  - the number of Memory Range Tracking Descriptors currently being tracked by the controller processing the command for other controllers in the NVM subsystem,

is greater than the value in the Controller Maximum Memory Range Tracking Descriptors (CMMRTD) field in the Identify Controller data structure (refer to Figure 312), then the controller processing the command shall abort the command with a status code of Invalid Field in Command.

- If the sum of the value in the Requested Number of Memory Range Tracking Descriptors (RNMRTD) field plus the number of Memory Range Tracking Descriptors currently being tracked by all controllers in the NVM subsystem is greater than the value in the NVM subsystem Maximum Memory Range Tracking Descriptors (NMMRTD) field in the Identify Controller data structure (refer to Figure 312), then the controller shall abort the command with a status code of Invalid Field in Command.
- If any controller in the NVM subsystem is already tracking host memory changes for the controller specified by the CNTLID field, then the controller processing the command shall abort the command with a status code of Invalid Controller Identifier.
- If:
  - the Memory Range Tracking Length Limit (MRTLL) bit in the Identify Controller data structure is set to '1'; and
  - the length as specified by the Requested Memory Range Tracking Granularity field and any Length field in a Track Memory Changes data structure is not a value that is a power of 2,

then the controller shall abort the command with a status code of Invalid Field in Command.

If the TACT bit is cleared to '0', then the data buffer shall be ignored by the controller.

**Figure 466: Track Memory Changes – Management Operation Specific Field**

Bits	Description
15:01	Reserved
00	<p><b>Tracking Action (TACT):</b> If this bit is set to '1', then the data buffer specifies the host memory ranges associated with the controller specified by the Controller Identifier field. Any writes to those host memory ranges by the controller specified by the Controller Identifier field shall be tracked by the controller processing the command (i.e., tracking starts).</p> <p>If this bit is cleared to '0', then controller processing the command shall stop tracking host memory changes by the controller specified by the Controller Identifier field.</p>

**Figure 467: Track Memory Changes – Command Dword 11**

Bits	Description
31:16	Reserved
15:00	<p><b>Controller Identifier (CNTLID):</b> This field specifies the identifier of the controller for which changes to host memory is being tracked. If the value of this field is the controller identifier of the controller processing the command, then the controller processing the command shall abort the command with a status code of Invalid Controller Identifier.</p>

If the controller specified by the CNTLID field is suspended (refer to section 5.1.17.1.1), then the controller shall abort the command with a status code of Controller Suspended.

**Figure 468: Track Memory Changes Data Structure**

Bytes	Description
<b>Header</b>	
00	<b>Version (VER):</b> This field specifies the version of this data structure. If this field is not cleared to 0h, then the controller shall abort this command with a status code of Invalid Field in Command.
02:01	Reserved
03	<b>Requested Memory Range Tracking Granularity (RMRTG):</b> This field specifies the granularity for the Length field in each Memory Range Tracking Descriptor contained in this data structure. The value specified is a power of two ( $2^n$ ) times 4 KiB. For example, a value of 8h indicates that the granularity of tracking is $(2^8) * 4$ KiB which is 1 MiB.
07:04	<b>Number of Memory Range Tracking Descriptors (RNMRTD):</b> This field specifies the number of Memory Range Tracking Descriptors in the Memory Range Tracking Descriptor list.  This field is a 1's based number.
<b>Memory Range Tracking Descriptor List</b>	
19:08	<b>Memory Range Tracking Descriptor 0:</b> This field contains the first Memory Range Tracking Descriptor as defined in Figure 469.
31:20	<b>Memory Range Tracking Descriptor 1:</b> This field contains the second Memory Range Tracking Descriptor as defined in Figure 469, if any.
...	
(RNMRTD*12)+7: (RNMRTD-1)*12+8	<b>Memory Range Tracking Descriptor RNMRTD-1:</b> This field contains the last Memory Range Tracking Descriptor as defined in Figure 469, if any.

Figure 469 defines a Memory Range Tracking Descriptor. The address range specified by a Memory Range Tracking Descriptor describes addresses specified in an SGL or PRP for a command submitted by a host to the controller specified in the CNTLID field (refer to Figure 467).

**Figure 469: Memory Range Tracking Descriptor**

Bytes	Description
07:00	<b>Address (SADDR):</b> This field specifies the starting 64-bit address of the memory range where the address is host memory (refer to section 1.5.46) associated with the controller specified by the CNTLID field (refer to Figure 467). The address is aligned to the granularity specified by the RMRTG field. For example, if the value in the RMRTG field is 8h, then the value in this field is required to be aligned to 1 MiB (i.e., bits 19:0 are cleared to 0h).
11:08	<b>Length (LEN):</b> This field specifies the length of this host memory range in units of the tracking granularity as defined by RMRTG field. For example, if the value n the RMRTG field is 8h and the value of this field is 4h, then the length of this memory range is 1 MiB * 4h which is 4 MiB.

If the controller is unable to track the requested number of Memory Range Tracking Descriptors (i.e., specified in the RNMRTD field) or unable to track the memory changes at the specified granularity (i.e., RMRTG field), then:

- the controller shall abort the command with a status code of Not Enough Resources; and
- Dword 0 and Dword 1 of the completion queue entry shall be returned as defined in Figure 470 and Figure 471.

If any Address field is not aligned to the granularity specified by the RMRTG field, then the controller shall abort the command with a status code of Invalid Field in Command.

If the address range specified by the Address field and the Length field specifies an address that is not host memory (refer to section 1.5.46) associated with the controller specified by the CNTLID field, then the controller processing the command shall abort the command with a status code of Invalid Field in Command. If:

- the command is successful; and

- the configuration within the controller associated with the controller specified by the CNTLID field changes resulting in that address range is no longer specifying host memory associated with the controller specified by the CNTLID field (e.g., the address range overlaps with PMR or CMB), then the behavior for tracking host memory changes for the controller specified by the CNTLID field is undefined.

A host should not specify MSI or MSI-X registers in the address range. If the address range specified by the Address field and the Length field contains addresses for MSI or MSI-X registers on the controller specified by the CNTLID field, then the controller may or may not report modification to those MSI or MSI-X register addresses.

**Figure 470: Track Memory Changes – Completion Queue Entry Dword 0**

Bits	Description
31:16	Reserved
15:00	<b>Memory Range Tracking Granularity (MRTG):</b> This field indicates the lowest granularity that the controller is able track the memory changes with the specified Memory Range Tracking Descriptor list (refer to Figure 468).  This units for this field is defined by the RMRTG field.

**Figure 471: Track Memory Changes – Completion Queue Entry Dword 1**

Bits	Description
31:00	<b>Number of Memory Range Tracking Descriptors (NMRTD):</b> This field indicates the number of Memory Range Tracking Descriptors for which the controller is able to track the memory changes with the requested granularity specified in the RMRTG field (refer to Figure 468).

### 5.1.27.2 Command Completion

Upon completion of the Track Send command, the controller posts a completion queue entry to the Admin Completion Queue indicating the status for the command. Section 5.1.27.1 describes completion details each for management operation.

The use of Dword 0 and Dword 1 of the completion queue entry is specific to the management operation specified in the Select field (refer to Figure 463). Refer to each management operation description in section 5.1.27.1 for details.

Track Send command specific status values (i.e., SCT field set to 1h) are shown in Figure 482.

**Figure 472: Track Send – Command Specific Status Values**

Value	Definition
1Fh	<b>Invalid Controller Identifier:</b> The controller for the specified Controller Identifier is already tracking host memory changes.
37h	<b>Invalid Controller Data Queue:</b> This error indicates: <ul style="list-style-type: none"> <li>that the specified Controller Data Queue Identifier is invalid for the controller processing the command; or</li> <li>the controller associated with the Controller Data Queue specified by the Controller Data Queue Identifier is already logging user data changes.</li> </ul>
38h	<b>Not Enough Resources:</b> This error indicates that there are not enough resources in the controller to process the command successfully.
39h	<b>Controller Suspended:</b> The requested operation is not allowed if the controller specified in the Controller Identifier field is suspended by a Migration Send command (refer to section 5.1.17.1.1).

## 5.2 Memory-Based Transport Admin Commands (PCIe)

This section describes Admin commands that are specific to the Memory-based transport model.

### 5.2.1 Create I/O Completion Queue command

The Create I/O Completion Queue command is used to create all I/O Completion Queues with the exception of the Admin Completion Queue. The Admin Completion Queue is created by specifying its base address in the ACQ property. If a PRP List is provided to describe the CQ, then the PRP List shall be maintained by host software at the same location in host physical memory and the values in the PRP List shall not be modified until the corresponding Delete I/O Completion Queue command for this CQ is completed successfully or the controller is reset. If the PRP List values are modified, the behavior is undefined.

The Create I/O Completion Queue command uses the PRP Entry 1, Command Dword 10, and Command Dword 11 fields. All other command specific fields are reserved.

**Figure 473: Create I/O Completion Queue – PRP Entry 1**

Bits	Description
63:00	<b>PRP Entry 1 (PRP1):</b> If CDW11.PC is set to '1', then this field specifies a 64-bit base memory address pointer of the Completion Queue that is physically contiguous. The address pointer is memory page aligned (based on the value in CC.MPS) unless otherwise specified. If CDW11.PC is cleared to '0', then this field specifies a PRP List pointer that describes the list of pages that constitute the Completion Queue. The list of pages is memory page aligned (based on the value in CC.MPS) unless otherwise specified. In both cases the PRP Entry shall have an offset of 0h. In a non-contiguous Completion Queue, each PRP Entry in the PRP List shall have an offset of 0h. If there is a PRP Entry with a non-zero offset, then the controller should return an error of PRP Offset Invalid.

**Figure 474: Create I/O Completion Queue – Command Dword 10**

Bits	Description
31:16	<b>Queue Size (QSIZE):</b> This field indicates the size of the Completion Queue to be created. If the size is 0h or larger than the controller supports, the controller should return an error of Invalid Queue Size. Refer to section 3.3.3.1. This is a 0's based value.
15:00	<b>Queue Identifier (QID):</b> This field indicates the identifier to assign to the Completion Queue to be created. This identifier corresponds to the Completion Queue Head Doorbell used for this command (i.e., the value y in the CQyHDBL section of the NVMe over PCIe Transport Specification). This value shall not exceed the value reported in the Number of Queues feature (refer to section 5.1.25.2.1) for I/O Completion Queues. If the value specified is 0h, exceeds the Number of Queues reported, or corresponds to an identifier already in use, the controller should return an error of Invalid Queue Identifier.

**Figure 475: Create I/O Completion Queue – Command Dword 11**

Bits	Description
31:16	<b>Interrupt Vector (IV):</b> This field's value is transport specific and is described in the applicable NVMe Transport binding specification if required. If not defined by the transport, then this field shall be cleared to 0h.
15:02	Reserved
01	<b>Interrupts Enabled (IEN):</b> If this bit is set to '1', then interrupts are enabled for this Completion Queue. If this bit is cleared to '0', then interrupts are disabled for this Completion Queue.
00	<p><b>Physically Contiguous (PC):</b> If this bit is set to '1', then the Completion Queue is physically contiguous and PRP Entry 1 (PRP1) is the address of a contiguous physical buffer. If this bit is cleared to '0', then the Completion Queue is not physically contiguous and PRP Entry 1 (PRP1) is a PRP List pointer. If this bit is cleared to '0' and CAP.CQR is set to '1', then the controller shall abort the command with a status code of Invalid Field in Command.</p> <p>If the:</p> <ul style="list-style-type: none"> <li>• queue is located in the Controller Memory Buffer;</li> <li>• PC is cleared to '0'; and</li> <li>• CMBLOC.CQPDS is cleared to '0',</li> </ul> <p>then the controller shall abort the command with a status code of Invalid Use of Controller Memory Buffer.</p>

### 5.2.1.1 Command Completion

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

Create I/O Completion Queue command specific status values are defined in Figure 476.

**Figure 476: Create I/O Completion Queue – Command Specific Status Values**

Value	Description
1h	<b>Invalid Queue Identifier:</b> The creation of the I/O Completion Queue failed due to an invalid queue identifier specified as part of the command. An invalid queue identifier is one that identifies the Admin Queue (i.e., 0h), is outside the range supported by the controller, or is a Completion Queue Identifier that is already in use.
2h	<b>Invalid Queue Size:</b> The host attempted to create an I/O Completion Queue: <ul style="list-style-type: none"> <li>with an invalid number of entries (e.g., a value of 0h or a value which exceeds the maximum supported by the controller, specified in CAP.MQES); or</li> <li>before initializing the CC.IOCQES field.</li> </ul>
8h	<b>Invalid Interrupt Vector:</b> The creation of the I/O Completion Queue failed due to an invalid interrupt vector specified as part of the command.

### 5.2.2 Create I/O Submission Queue command

The Create I/O Submission Queue command is used to create I/O Submission Queues. The Admin Submission Queue is created by specifying its base address in the ASQ property. If a PRP List is provided to describe the SQ to be created, then the PRP List shall be maintained by host software at the same location in host physical memory and the values in the PRP List shall not be modified until the corresponding Delete I/O Submission Queue command for that SQ is completed or the controller is reset. If the PRP List values are modified, the behavior is undefined.

The Create I/O Submission Queue command uses the PRP Entry 1, Command Dword 10, Command Dword 11, and Command Dword 12 fields. All other command specific fields are reserved.

**Figure 477: Create I/O Submission Queue – PRP Entry 1**

Bits	Description
63:00	<b>PRP Entry 1 (PRP1):</b> If CDW11.PC is set to '1', then this field specifies a 64-bit base memory address pointer of the Submission Queue that is physically contiguous. The address pointer is memory page aligned (based on the value in CC.MPS) unless otherwise specified. If CDW11.PC is cleared to '0', then this field specifies a PRP List pointer that describes the list of pages that constitute the Submission Queue. The list of pages is memory page aligned (based on the value in CC.MPS) unless otherwise specified. In both cases, the PRP Entry shall have an offset of 0h. In a non-contiguous Submission Queue, each PRP Entry in the PRP List shall have an offset of 0h. If there is a PRP Entry with a non-zero offset, then the controller should return an error of PRP Offset Invalid.

**Figure 478: Create I/O Submission Queue – Command Dword 10**

Bits	Description
31:16	<b>Queue Size (QSIZE):</b> This field specifies the size of the Submission Queue to be created. If the size is 0h or larger than the controller supports, the controller should return an error of Invalid Queue Size. Refer to section 3.3.3.1. This is a 0's based value.
15:00	<b>Queue Identifier (QID):</b> This field specifies the identifier to assign to the Submission Queue to be created. This identifier corresponds to the Submission Queue Tail Doorbell used for this command (i.e., the value <i>y</i> in SQyTDBL section of the NVMe over PCIe Transport Specification). This value shall not exceed the value reported in the Number of Queues feature (refer to section 5.1.25.2.1) for I/O Submission Queues. If the value specified is 0h, exceeds the Number of Queues reported, or corresponds to an identifier already in use, the controller should return an error of Invalid Queue Identifier.

**Figure 479: Create I/O Submission Queue – Command Dword 11**

Bits	Description										
31:16	<p><b>Completion Queue Identifier (CQID):</b> This field specifies the identifier of the I/O Completion Queue to utilize for any command completions entries associated with this Submission Queue.</p> <p>If the value specified:</p> <ol style="list-style-type: none"> <li>is 0h (i.e., the Admin Completion Queue), then the controller shall abort the command with a status code of Invalid Queue Identifier;</li> <li>is outside the range supported by the controller, then the controller shall abort the command with a status code of Invalid Queue Identifier; or</li> <li>is within the range supported by the controller and does not specify an I/O Completion Queue that has been created, then the controller shall abort the command with a status code of Completion Queue Invalid.</li> </ol>										
15:03	Reserved										
02:01	<p><b>Queue Priority (QPRIO):</b> This field specifies the priority class to use for commands within this Submission Queue. This field is only used when the weighted round robin with urgent priority class is the arbitration mechanism selected. This field shall be ignored by the controller if weighted round robin with urgent priority class is not used. Refer to section 3.4.4.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Urgent</td> </tr> <tr> <td>01b</td> <td>High</td> </tr> <tr> <td>10b</td> <td>Medium</td> </tr> <tr> <td>11b</td> <td>Low</td> </tr> </tbody> </table>	Value	Definition	00b	Urgent	01b	High	10b	Medium	11b	Low
Value	Definition										
00b	Urgent										
01b	High										
10b	Medium										
11b	Low										
00	<p><b>Physically Contiguous (PC):</b> If this bit is set to '1', then the Submission Queue is physically contiguous and PRP Entry 1 (PRP1) is the address of a contiguous physical buffer. If this bit is cleared to '0', then the Submission Queue is not physically contiguous and PRP Entry 1 (PRP1) is a PRP List pointer. If this bit is cleared to '0' and CAP.CQR is set to '1', then the controller should return an error of Invalid Field in Command.</p> <p>If the:</p> <ul style="list-style-type: none"> <li>queue is located in the Controller Memory Buffer;</li> <li>PC is cleared to '0'; and</li> <li>CMBLOC.CQPDS is cleared to '0',</li> </ul> <p>then the controller shall abort the command with Invalid Use of Controller Memory Buffer status.</p>										

**Figure 480: Create I/O Submission Queue – Command Dword 12**

Bits	Description
31:16	Reserved
15:00	<p><b>NVM Set Identifier (NVMSETID):</b> This field indicates the identifier of the NVM Set to be associated with this Submission Queue.</p> <p>If this field is cleared to 0h or the SQ Associations capability is not supported (refer to section 8.1.25), then this Submission Queue is not associated with any specific NVM Set.</p> <p>If this field is set to a non-zero value that is not specified in the NVM Set List (refer to Figure 317) and the SQ Associations capability is supported (refer to section 8.1.25), then the controller shall abort the command with a status code of Invalid Field in Command.</p> <p>The host should not submit commands for namespaces associated with other NVM Sets in this Submission Queue (refer to section 8.1.25).</p>

### 5.2.2.1 Command Completion

Upon completion of the Create I/O Submission Queue command, the controller posts a completion queue entry to the Admin Completion Queue.

Create I/O Submission Queue command specific status values are defined in Figure 481.

**Figure 481: Create I/O Submission Queue – Command Specific Status Values**

Value	Definition
0h	<b>Completion Queue Invalid:</b> The Completion Queue identifier specified in the command has not been created.
1h	<b>Invalid Queue Identifier:</b> The creation of the I/O Submission Queue failed due an invalid queue identifier specified as part of the command. An invalid queue identifier is one that identifies the Admin Queue (i.e., 0h), is outside the range supported by the controller, or is a Submission Queue Identifier that is already in use.
2h	<b>Invalid Queue Size:</b> The host attempted to create an I/O Submission Queue: <ul style="list-style-type: none"> <li>with an invalid number of entries (e.g., a value of 0h or a value which exceeds the maximum supported by the controller, specified in CAP.MQES); or</li> <li>before initializing the CC.IOSQES field.</li> </ul>

### 5.2.3 Delete I/O Completion Queue command

The Delete I/O Completion Queue command is used to delete an I/O Completion Queue. The Delete I/O Completion Queue command uses the Command Dword 10 field. All other command specific fields are reserved. After this command has completed, the PRP List that describes the Completion Queue may be deallocated by host software.

Host software shall ensure that any associated I/O Submission Queue is deleted prior to deleting a Completion Queue. If there are any associated I/O Submission Queues present, then the Delete I/O Completion Queue command shall abort with a status code of Invalid Queue Deletion.

Note: It is not possible to delete the Admin Completion Queue.

**Figure 482: Delete I/O Completion Queue – Command Dword 10**

Bits	Description
31:16	Reserved
15:00	<b>Queue Identifier (QID):</b> This field indicates the identifier of the Completion Queue to be deleted. The value of 0h (Admin Completion Queue) shall not be specified.

#### 5.2.3.1 Command Completion

Upon completion of the Delete I/O command, the controller posts a completion queue entry to the Admin Completion Queue. Delete I/O Completion Queue command specific status values are defined in Figure 483.

**Figure 483: Delete I/O Completion Queue – Command Specific Status Values**

Value	Definition
01h	<b>Invalid Queue Identifier:</b> The Queue Identifier specified in the command is invalid. This error is also indicated if the Admin Completion Queue identifier is specified.
0Ch	<b>Invalid Queue Deletion:</b> This error indicates that it is invalid to delete the I/O Completion Queue specified. The typical reason for this error condition is that there is an associated I/O Submission Queue that has not been deleted.

### 5.2.4 Delete I/O Submission Queue command

The Delete I/O Submission Queue command is used to delete an I/O Submission Queue. The Delete I/O Submission Queue command uses the Command Dword 10 field. All other command specific fields are reserved. After this command has completed, the PRP List that describes the Submission Queue may be deallocated by host software.

Upon successful completion of the Delete I/O Submission Queue command, all I/O commands previously submitted to the indicated Submission Queue shall be either explicitly completed or implicitly completed. Prior to returning a completion queue entry for the Delete I/O Submission Queue command, other commands previously submitted to the I/O Submission Queue to be deleted may be completed with appropriate status (e.g., Successful Completion, Command Aborted due to SQ Deletion). After successful



completion of the Delete I/O Submission Queue command, the controller shall not post completion status for any I/O commands that were submitted to the deleted I/O Submission Queue. The successful completion of the Delete I/O Submission Queue command indicates an implicit completion status of Command Aborted due to SQ Deletion for any previously submitted I/O commands that did not have a completion queue entry posted by the controller.

Note: It is not possible to delete the Admin Submission Queue.

**Figure 484: Delete I/O Submission Queue – Command Dword 10**

Bits	Description
31:16	Reserved
15:00	<b>Queue Identifier (QID):</b> This field indicates the identifier of the Submission Queue to be deleted. The value of 0h (Admin Submission Queue) shall not be specified.

### 5.2.4.1 Command Completion

After all commands submitted to the indicated I/O Submission Queue are either completed or aborted, a completion queue entry is posted to the Admin Completion Queue when the queue has been deleted. Delete I/O Submission Queue command specific status values are defined in Figure 485.

**Figure 485: Delete I/O Submission Queue – Command Specific Status Values**

Value	Definition
1h	<b>Invalid Queue Identifier:</b> The Queue Identifier specified in the command is invalid. This error is also indicated if the Admin Submission Queue identifier is specified.

### 5.2.5 Doorbell Buffer Config command

The Doorbell Buffer Config command is used to provide two separate memory buffers that mirror the controller’s doorbell properties defined in section 3.1.4. This command is intended for emulated controllers and is not typically supported by a physical NVMe controller. The two buffers are known as “Shadow Doorbell” and “EventIdx”, respectively. Refer to “Updating Controller Doorbell Properties using a Shadow Doorbell Buffer” in section B.5.2 for an example of how these buffers may be used.

The Doorbell Buffer Config command uses the PRP Entry 1 and PRP Entry 2 fields. All other command specific fields are reserved. The command is not namespace specific, does not support metadata, and does not support SGLs. The settings are not retained across a Controller Level Reset.

Each buffer supplied with the Doorbell Buffer Config command shall be a single physical memory page as defined by the CC.MPS field. The controller shall ensure that the following condition is satisfied:

$$(4 \ll \text{CAP.DSTRD}) * (\max(\text{NSQA}, \text{NCQA})+1) \leq (2^{(12+\text{CC.MPS})})$$

**Figure 486: Doorbell Buffer Config – Shadow Doorbell and EventIdx**

Start (Offset in Buffer) <sup>1, 2</sup>	End (Offset in Buffer) <sup>1, 2</sup>	Description <sup>2</sup>
00h	03h	Submission Queue 0 Tail Doorbell or EventIdx (Admin)
00h + (1 * (4 << CAP.DSTRD))	03h + (1 * (4 << CAP.DSTRD))	Completion Queue 0 Head Doorbell or EventIdx (Admin)
00h + (2 * (4 << CAP.DSTRD))	03h + (2 * (4 << CAP.DSTRD))	Submission Queue 1 Tail Doorbell or EventIdx
00h + (3 * (4 << CAP.DSTRD))	03h + (3 * (4 << CAP.DSTRD))	Completion Queue 1 Head Doorbell or EventIdx
...	...	...
00h + (2y * (4 << CAP.DSTRD))	03h + (2y * (4 << CAP.DSTRD))	Submission Queue y Tail Doorbell or EventIdx
00h + ((2y + 1) * (4 << CAP.DSTRD))	03h + ((2y + 1) * (4 << CAP.DSTRD))	Completion Queue y Head Doorbell or EventIdx
Notes:		

**Figure 486: Doorbell Buffer Config – Shadow Doorbell and EventIdx**

Start (Offset in Buffer) <sup>1, 2</sup>	End (Offset in Buffer) <sup>1, 2</sup>	Description <sup>2</sup>
1. The offsets in Start and End are referenced to the value provided in PRP1 for the doorbell buffer and to the value provided in PRP2 for the EventIdx buffer.		
2. The value of y is equal to max(NSQA, NCQA).		

**Figure 487: Doorbell Buffer Config – PRP Entry 1**

Bits	Description
63:00	<b>PRP Entry 1 (PRP1):</b> This field specifies a 64-bit base memory address pointer to the Shadow Doorbell buffer with the definition specified in Figure 486. The Shadow Doorbell buffer is updated by the host. This buffer shall be memory page aligned.

**Figure 488: Doorbell Buffer Config – PRP Entry 2**

Bits	Description
63:00	<b>PRP Entry 2 (PRP2):</b> This field specifies a 64-bit base memory address pointer to the EventIdx buffer with the definition specified in Figure 486. The EventIdx buffer is updated by the para-virtualized controller. This buffer shall be memory page aligned.

### 5.2.5.1 Command Completion

When the command is completed, the controller posts a completion queue entry to the Admin Completion Queue indicating the status for the command. If the Shadow Doorbell buffer or EventIdx buffer memory addresses are invalid, then a status code of Invalid Field in Command shall be returned.

### 5.2.6 Virtualization Management command

The Virtualization Management command is supported by primary controllers that support the Virtualization Enhancements capability. This command is used for several functions:

- Modifying Flexible Resource allocation for the primary controller;
- Assigning Flexible Resources for secondary controllers; and
- Setting the Online and Offline state for secondary controllers.

Refer to section 8.2.6 for more on the Virtualization Enhancements capability and the Virtualization Management command.

The Virtualization Management command uses the Command Dword 10 and Command Dword 11 fields. All other command specific fields are reserved.

If the action requested specifies a range of controller resources that:

- does not exist;
- is a Private Resource (e.g., VQ resources are requested when VQ resources are not supported, VI resources are requested when VI resources are not supported); or
- is currently in use (e.g., the number of Controller Resources (NR) is greater than the number of remaining available flexible resources),

then the command is aborted with a status code of Invalid Resource Identifier.

**Figure 489: Virtualization Management – Command Dword 10**

Bits	Description
31:16	<b>Controller Identifier (CNTLID):</b> This field indicates the controller for which controller resources are to be modified.
15:11	Reserved

**Figure 489: Virtualization Management – Command Dword 10**

Bits	Description																
10:08	<p><b>Resource Type (RT):</b> This field indicates the type of controller resource to be modified.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td><b>VQ Resources</b></td> </tr> <tr> <td>001b</td> <td><b>VI Resources</b></td> </tr> <tr> <td>010b to 111b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	000b	<b>VQ Resources</b>	001b	<b>VI Resources</b>	010b to 111b	Reserved								
Value	Definition																
000b	<b>VQ Resources</b>																
001b	<b>VI Resources</b>																
010b to 111b	Reserved																
07:04	Reserved																
03:00	<p><b>Action (ACT):</b> This field indicates the operation for the command to perform as described below.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Reserved</td> </tr> <tr> <td>1h</td> <td><b>Primary Controller Flexible Allocation:</b> Set the number of Flexible Resources allocated to this primary controller following the next Controller Level Reset other than a Controller Reset. If the Controller Identifier field does not correspond to this primary controller, then a status code of Invalid Controller Identifier is returned. This value is persistent across power cycles and resets.</td> </tr> <tr> <td>2h to 6h</td> <td>Reserved</td> </tr> <tr> <td>7h</td> <td><b>Secondary Controller Offline:</b> 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><b>Secondary Controller Assign:</b> 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><b>Secondary Controller Online:</b> 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 8.2.6) 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>Ah to Fh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	Reserved	1h	<b>Primary Controller Flexible Allocation:</b> Set the number of Flexible Resources allocated to this primary controller following the next Controller Level Reset other than a Controller Reset. If the Controller Identifier field does not correspond to this primary controller, then a status code of Invalid Controller Identifier is returned. This value is persistent across power cycles and resets.	2h to 6h	Reserved	7h	<b>Secondary Controller Offline:</b> 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	<b>Secondary Controller Assign:</b> 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	<b>Secondary Controller Online:</b> 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 8.2.6) 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.	Ah to Fh	Reserved
Value	Definition																
0h	Reserved																
1h	<b>Primary Controller Flexible Allocation:</b> Set the number of Flexible Resources allocated to this primary controller following the next Controller Level Reset other than a Controller Reset. If the Controller Identifier field does not correspond to this primary controller, then a status code of Invalid Controller Identifier is returned. This value is persistent across power cycles and resets.																
2h to 6h	Reserved																
7h	<b>Secondary Controller Offline:</b> 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	<b>Secondary Controller Assign:</b> 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	<b>Secondary Controller Online:</b> 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 8.2.6) 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.																
Ah to Fh	Reserved																

**Figure 490: Virtualization Management – Command Dword 11**

Bits	Description
31:16	Reserved
15:00	<b>Number of Controller Resources (NR):</b> This field indicates a number of controller resources to allocate or assign.

### 5.2.6.1 Command Completion

Command specific status values associated with the Virtualization management command are defined in Figure 491.

**Figure 491: Virtualization Management – Command Specific Status Values**

Value	Definition
1Fh	<b>Invalid Controller Identifier:</b> An invalid Controller Identifier was specified.

**Figure 491: Virtualization Management – Command Specific Status Values**

Value	Definition
20h	<b>Invalid Secondary Controller State:</b> The action requested for the secondary controller is invalid based on the current state of the secondary controller and its primary controller.
21h	<b>Invalid Number of Controller Resources:</b> The specified number of Flexible Resources is invalid (e.g., the Number of Controller Resources (NR) is greater than VQ Resources Flexible Total (VQFRT) (refer to Figure 330), the Number of Controller Resources (NR) is greater than VQ Resources Flexible Secondary Maximum (VQFRSM) (refer to Figure 330)).
22h	<b>Invalid Resource Identifier:</b> At least one of the specified resource identifiers was invalid (e.g., the Number of Controller Resources (NR) is greater than the number of remaining available flexible resources).

Dword 0 of the completion queue entry contains information about the controller resources that were modified as part of the Primary Controller Flexible Allocation and Secondary Controller Assign actions. Dword 0 of the completion queue entry is defined in Figure 492.

**Figure 492: Virtualization Management – Completion Queue Entry Dword 0**

Bits	Description
31:16	Reserved
15:00	<b>Number of Controller Resources Modified (NRM):</b> This field indicates the number of controller resources that were allocated or assigned. The value may be smaller or larger than the number requested.

### 5.3 Message-Based Transport Admin Commands (Fabrics)

This section describes Admin commands that are specific to the Message-Based transport model.

#### 5.3.1 Clear Exported NVM Resource Configuration command

The Clear Exported NVM Resource Configuration command is used to delete all Exported NVM resource configuration information including removing all Exported NVM Resources (refer to section 1.5.35 for a list of Exported NVM Resources).

All command specific fields are reserved.

The Clear Exported NVM Resource Configuration command shall not be supported by Exported NVM Subsystems.

The Clear Exported NVM Resource Configuration command shall not affect Underlying Namespaces or Underlying NVM Subsystems.

If a Clear Exported NVM Resource Configuration command is issued without first disconnecting all hosts from all Exported NVM Subsystems, then the controller shall abort the command with a status code of Command Sequence Error.

##### 5.3.1.1 Command Completion

Upon completion of the Clear Exported NVM Resource Configuration command, the controller shall post a completion queue entry to the Admin Completion Queue indicating the status for the command. Refer to section 8.3.3 for usages.

#### 5.3.2 Create Exported NVM Subsystem command

The Create Exported NVM Subsystem command is used to create a new Exported NVM Subsystem. The Create Exported NVM Subsystem command uses the Data Pointer and Command Dword 10. All other command specific fields are reserved.

The Create Exported NVM Subsystem command shall not be supported by Exported NVM Subsystems.

**Figure 493: Create Exported NVM Subsystem – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field.

**Figure 494: Create Exported NVM Subsystem – Command Dword 10**

Bits	Description
31:09	Reserved
08	<b>Restricted Access (RA):</b> A value of '0' configures this Exported NVM Subsystem for unrestricted access and may be accessed by any host. A value of '1' configures this Exported NVM Subsystem to restrict access to Host NQNs present in this Exported NVM Subsystem's Allowed Host List.
07:00	Reserved

The Restricted Access bit defined in Figure 494 specifies the initial access setting for the Exported NVM Subsystem.

### 5.3.2.1 Command Completion

Upon successful completion of the Create Exported NVM Subsystem command:

- a new Exported NVM Subsystem is created with the Access Control mode specified in the Restricted Access bit of the command:
  - a) the newly created Exported NVM Subsystem shall have an empty list of allowed Host NQNs; and
  - b) there are no Exported Namespaces linked to the Exported NVM Subsystem;
- the controller posts a completion queue entry to the Admin Completion Queue indicating the status for the command (refer to section 8.3.3 for usages); and
- the SUBNQN for the newly created Exported NVM Subsystem shall be returned in the command data buffer.

### 5.3.3 Discovery Information Management command

The Discovery Information Management command registers, de-registers, or updates discovery information entries.

The type of task performed by the Discovery Information Management command (i.e., registration, de-registration, or update) is determined by the value set in Task (TAS) field of Command Dword 10 (refer to Figure 496).

For a register task (i.e., TAS field cleared to 0h), the Discovery Information Management command registers:

- one or more host discovery information entries; or
- one or more NVM subsystem discovery information entries.

For a de-register task (i.e., TAS field set to 1h), the Discovery Information Management command de-registers:

- one or more host discovery information entries; or
- one or more NVM subsystem discovery information entries.

For an update task (i.e., TAS field set to 2h), the Discovery Information Management command updates:

- one host discovery information entry; or
- one NVM subsystem discovery information entry.

Discovery information entries may be one of the following:

- a basic discovery information entry (refer to Figure 294); or
- an extended discovery information entry (refer to Figure 498).

Host discovery information entries shall be extended discovery information entries.

NVM subsystem discovery information entries may be basic discovery information entries or may be extended discovery information entries. Each NVM subsystem discovery information entry may specify an NVM subsystem that exposes namespaces that hosts may access or may specify a referral to another Discovery subsystem.

The Discovery Information Management command uses the Data Pointer field as shown in Figure 495 and Command Dword 10 as shown in Figure 496. All other command specific fields are reserved.

The data portion of the Discovery Information Management command contains a header that identifies the entity performing the register, de-register, or update task.

If a register task is being performed and the data portion of the Discovery Information Management command does not contain one or more discovery information entries, then the controller shall abort the command with status code of Invalid Field in Command. If the number of discovery information entries contained in the data portion of the Discovery Information Management command exceeds the available capacity for new discovery information entries on the CDC or DDC, then the controller shall abort the command with a status code of Insufficient Discovery Resources. If multiple register tasks are performed by the same entity (i.e., the value set in the Entity Identifier (EID) field of the header is associated with an existing registration record contained in the CDC or DDC), and:

- a) the Entry Key associated with a discovery information entry in the Discovery Information Management command matches the Entry Key associated with an existing registration record, then the existing registration record contained in the CDC or DDC shall be updated with the discovery information entry from the Discovery Information Management command; or
- b) the Entry Key associated with a discovery information entry in the Discovery Information Management command does not match the Entry Key associated with an existing registration record, then the discovery information entry in the Discovery Information Management command shall be registered with the CDC or DDC.

If a de-register task is being performed and the data portion of the Discovery Information Management command does not contain one or more discovery information entries, then the controller shall abort the command with a status code of Invalid Field in Command.

If an update task is being performed and the data portion of the Discovery Information Management command does not contain two discovery information entries, then the controller shall abort the command with a status code of Invalid Field in Command. The first discovery information entry identifies the registration record contained in the CDC or DDC that is to be updated. The fields in the first discovery information entry that are not used as part of the Entry Key are ignored. The second discovery information entry replaces the existing registration record identified by the Entry Key from the first discovery information entry. The update task shall be atomic.

The format for the data portion of the Discovery Information Management command is defined in Figure 497.

Based upon the value set in the Entity Type (ETYPE) field of the header, the data portion of a Discovery Information Management command shall:

- only contain host extended discovery information entries if the ETYPE field is set to 1h (i.e., a host is performing the register, de-register, or update task);
- only contain host extended discovery information entries if the ETYPE field is set to 3h (i.e., a CDC is performing the register, de-register, or update task); and

- only contain NVM subsystem basic discovery information entries or NVM subsystem extended discovery information entries if the ETYPE field is set to 2h (i.e., a DDC is performing the register, de-register, or update task). The DDC may set the Port Local (PORTLCL) field to 1h if the NVM subsystem discovery information entries being registered, de-registered, or updated are only for NVM subsystem ports that are presented through the same NVM subsystem port on the DDC that is performing the register, de-register, or update task.

Host extended discovery information entries and NVM subsystem extended discovery information entries each contain the same set of fields, but not all of the fields are used for both extended discovery information entry types. Refer to Figure 500 for the usage of the extended discovery information entry fields for each extended discovery information entry type. If the entity performing a register, de-register, or update task uses any field in an extended discovery information entry that conflicts with the value set in the ETYPE field of the Discovery Information Management command data portion's header (e.g., a host uses a field intended for only NVM subsystem extended discovery information entries), then the controller shall abort the command with a status code of Invalid Discovery Information.

Host extended discovery information entries shall contain at least one extended attribute containing a Host Identifier. NVM subsystem extended discovery information entries may contain zero or more extended attributes. The format for an extended attribute is defined in Figure 499.

**Figure 495: Discovery Information Management – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field.

**Figure 496: Discovery Information Management – Command Dword 10**

Bits	Description										
31:04	Reserved										
03:00	<p><b>Task (TAS):</b> This field selects the type of management task to perform.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Register</td> </tr> <tr> <td>1h</td> <td>De-Register</td> </tr> <tr> <td>2h</td> <td>Update</td> </tr> <tr> <td>3h to Fh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	Register	1h	De-Register	2h	Update	3h to Fh	Reserved
Value	Definition										
0h	Register										
1h	De-Register										
2h	Update										
3h to Fh	Reserved										

**Figure 497: Discovery Information Management – Data**

Bytes	O/M <sup>1</sup>	Description
<b>Header</b>		
03:00	M	<b>Total Data Length (TDL):</b> This field specifies the length in bytes of the entire data portion of the command.
07:04		Reserved
15:08	M	<b>Number of Entries (NUMENT):</b> This field specifies the number of discovery information entries being registered, de-registered, or updated. For a register or de-register task, this field shall be non-zero. For an update task, this field shall be set to 2h.

Figure 497: Discovery Information Management – Data

Bytes	O/M <sup>1</sup>	Description																														
17:16	M	<p><b>Entry Format (ENTFMT):</b> This field specifies the format of the discovery information entries being registered, de-registered, or updated.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Reserved</td> </tr> <tr> <td>1h</td> <td>Basic discovery information entry</td> </tr> <tr> <td>2h</td> <td>Extended discovery information entry</td> </tr> <tr> <td>3h to FFFFh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	Reserved	1h	Basic discovery information entry	2h	Extended discovery information entry	3h to FFFFh	Reserved																				
Value	Definition																															
0h	Reserved																															
1h	Basic discovery information entry																															
2h	Extended discovery information entry																															
3h to FFFFh	Reserved																															
19:18	M	<p><b>Entity Type (ETYPE):</b> This field specifies the type of entity performing the register, de-register, or update task.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>Reserved</td> </tr> <tr> <td>1h</td> <td>Host</td> </tr> <tr> <td>2h</td> <td>Direct Discovery controller</td> </tr> <tr> <td>3h</td> <td>Centralized Discovery controller</td> </tr> <tr> <td>4h to FFFFh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	0h	Reserved	1h	Host	2h	Direct Discovery controller	3h	Centralized Discovery controller	4h to FFFFh	Reserved																		
Value	Definition																															
0h	Reserved																															
1h	Host																															
2h	Direct Discovery controller																															
3h	Centralized Discovery controller																															
4h to FFFFh	Reserved																															
20	O <sup>2</sup>	<p><b>Port Local (PORTLCL):</b> This field specifies if the NVM subsystem discovery information entries being registered, de-registered, or updated are only for NVM subsystem ports that are presented through the same NVM subsystem port on the DDC that is performing the register, de-register, or update task. If the Entity Type (ETYPE) field is set to any value other than 2h (i.e., an entity other than a DDC is performing the register, de-register, or update task) and this field is not cleared to 0h, then the controller shall abort the command with a status code of Invalid Field in Command.</p> <p>If this field is set to 1h, then the NVM subsystem discovery information entries being registered, de-registered, or updated are only for NVM subsystem ports that are presented through the same NVM subsystem port on the DDC that is performing the register, de-register, or update task.</p> <p>If this field is cleared to 0h, then the NVM subsystem discovery information entries being registered, de-registered, or updated may be for NVM subsystem ports that are presented through any NVM subsystem port on that DDC.</p>																														
21		Reserved																														
23:22	M	<p><b>Entry Key Type (EKTYPE):</b> This field specifies the fields in each discovery information entry that shall be used to determine if the entry matches an existing registration record contained in the CDC or DDC. Host discovery information entries shall use an EKTYPE of 5Fh (i.e., TRADDR Based). NVM subsystem discovery information entries may use an EKTYPE of either 3Fh (i.e., Port ID Based) or 5Fh (i.e., TRADDR Based). All other values are reserved.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> <th>Port ID Based (3Fh)</th> <th>TRADDR Based (5Fh)</th> </tr> </thead> <tbody> <tr> <td>15:7</td> <td>Reserved</td> <td></td> <td></td> </tr> <tr> <td>6</td> <td>Transport Address (TRADDR)</td> <td>Not used</td> <td>Used</td> </tr> <tr> <td>5</td> <td>Port ID (PORTID)</td> <td>Used</td> <td>Not used</td> </tr> <tr> <td>4</td> <td>Transport Type (TRTYPE)</td> <td rowspan="4">Used</td> <td rowspan="4">Used</td> </tr> <tr> <td>3</td> <td>Address Family (ADRFAM)</td> </tr> <tr> <td>2</td> <td>Transport Service Identifier (TRSVCID)</td> </tr> <tr> <td>1</td> <td>Transport Specific Address Subtype (TSAS)</td> </tr> <tr> <td>0</td> <td>NVMe Qualified Name (NQN)</td> <td></td> <td></td> </tr> </tbody> </table>	Bits	Description	Port ID Based (3Fh)	TRADDR Based (5Fh)	15:7	Reserved			6	Transport Address (TRADDR)	Not used	Used	5	Port ID (PORTID)	Used	Not used	4	Transport Type (TRTYPE)	Used	Used	3	Address Family (ADRFAM)	2	Transport Service Identifier (TRSVCID)	1	Transport Specific Address Subtype (TSAS)	0	NVMe Qualified Name (NQN)		
Bits	Description	Port ID Based (3Fh)	TRADDR Based (5Fh)																													
15:7	Reserved																															
6	Transport Address (TRADDR)	Not used	Used																													
5	Port ID (PORTID)	Used	Not used																													
4	Transport Type (TRTYPE)	Used	Used																													
3	Address Family (ADRFAM)																															
2	Transport Service Identifier (TRSVCID)																															
1	Transport Specific Address Subtype (TSAS)																															
0	NVMe Qualified Name (NQN)																															



**Figure 497: Discovery Information Management – Data**

Bytes	O/M <sup>1</sup>	Description
279:24	M	<b>Entity Identifier (EID):</b> This field specifies an NQN that is used to uniquely identify the entity performing the registration, de-registration, or update task within the fabric in UUID-based format. Refer to section 4.7 for the formatting requirements of UUID-based format NQNs.
535:280	O	<b>Entity Name (ENAME):</b> This field specifies the name associated with the entity as an ASCII string (refer to section 1.4.2 for ASCII string requirements). The value specified in this field is determined by the value set in the Entity Type (ETYPE) field.  If the ETYPE field is set to 1h (i.e., a host is performing the register, de-register, or update task), then this field specifies the name associated with the host.  If the ETYPE field is set to 2h (i.e., a DDC is performing the register, de-register, or update task), then this field specifies the name associated with the DDC.  If the ETYPE field is set to 3h (i.e., a CDC is performing the register, de-register, or update task), then this field specifies the name associated with the CDC.
599:536	O	<b>Entity Version (EVER):</b> This field specifies the entity version as an ASCII string (refer to section 1.4.2 for ASCII string requirements). The value specified in this field is determined by the value set in the Entity Type (ETYPE) field.  If the ETYPE field is set to 1h (i.e., a host is performing the register, de-register, or update task), then this field specifies the host entity version (e.g., host operating system name and version).  If the ETYPE field is set to 2h (i.e., a DDC is performing the register, de-register, or update task), then this field specifies the Discovery controller entity version (e.g., currently active firmware revision of the DDC).  If the ETYPE field is set to 3h (i.e., a CDC is performing the register, de-register, or update task), then this field may specify either:  a) the host entity version (e.g., host operating system name and version); or b) the Discovery controller entity version (e.g., currently active firmware revision of the CDC).
1023:600		Reserved
<b>Discovery Information Entry List</b>		
(TEL - 1) + 1024: 1024	M	<b>Discovery Information Entry 0:</b> This field contains the first discovery information entry as defined in Figure 497 for extended discovery information entries and as defined in Figure 294 for basic discovery information entries, where TEL is the size specified in the Total Entry Length (TEL) field of the extended discovery information entry. For basic discovery information entries, TEL shall be 1,024 bytes.
...	O	...
TDL - 1: TDL - TEL	M <sup>3</sup>	<b>Discovery Information Entry NUMENT-1:</b> This field contains the last discovery information entry as defined in Figure 498 for extended discovery information entries (if present) and as defined in Figure 294 for basic discovery information entries (if present), where TEL is the size specified in the Total Entry Length (TEL) field of the extended discovery information entry and TDL is the size specified in the Total Data Length (TDL) field. For basic discovery information entries, TEL shall be 1,024 bytes.
<p>Notes:</p> <ol style="list-style-type: none"> <li>O/M definition: O = Optional, M = Mandatory.</li> <li>This field is optional for a register, de-register, or update task performed by a DDC. For a register, de-register, or update task performed by a host or CDC, this field shall be cleared to 0h.</li> <li>This field is mandatory for an update task. For a register or de-register task, this field is optional.</li> </ol>		

Figure 498: Extended Discovery Information Entry

Bytes	O/M <sup>1</sup>	Description
00	M	<b>Transport Type (TRTYPE):</b> This field specifies the transport type as defined in the Transport Type (TRTYPE) field in Figure 294.
01	M	<b>Address Family (ADRFAM):</b> This field specifies the address family as defined in the Address Family (ADRFAM) field in Figure 294.
02	M <sup>2</sup>	<b>Subsystem Type (SUBTYPE):</b> This field specifies the type of the NVM subsystem that is indicated in this entry as defined in the Subsystem Type (SUBTYPE) field in Figure 294.
03	M <sup>2</sup>	<b>Transport Requirements (TREQ):</b> This field specifies requirements for the NVMe Transport as defined in the Transport Requirements (TREQ) field in Figure 294.
05:04	M <sup>2</sup>	<b>Port ID (PORTID):</b> This field specifies a particular NVM subsystem port as defined in the Port ID (PORTID) field in Figure 294.
07:06	M <sup>2</sup>	<b>Controller ID (CNTLID):</b> This field specifies the controller ID as defined in the Controller ID (CNTLID) field in Figure 294. This field shall specify a controller ID that a host is able to use in a Connect command to the NVM subsystem being registered.
09:08	M <sup>2</sup>	<b>Admin Max SQ Size (ASQSZ):</b> This field specifies the maximum size of an Admin Submission Queue as defined in the Admin Max SQ Size (ASQSZ) field in Figure 294.
31:10		Reserved
63:32	M <sup>2</sup>	<b>Transport Service Identifier (TRSVCID):</b> This field specifies the NVMe Transport service identifier as an ASCII string as defined in the Transport Service Identifier (TRSVCID) field in Figure 294.
255:64		Reserved
511:256	M	<p><b>NVMe Qualified Name (NQN):</b> If the Entity Type (ETYPE) field in the Discovery Information Management command data portion's header is set to 2h (i.e., a DDC is performing the register, de-register, or update task), then this field specifies the NVMe Qualified Name (NQN) that uniquely identifies the NVM subsystem as defined in the NVM Subsystem Qualified Name (SUBNQN) field in Figure 294.</p> <p>If the ETYPE field in the Discovery Information Management command data portion's header is set to 1h or 3h (i.e., a host or CDC is performing the register, de-register, or update task), then this field specifies the NQN that uniquely identifies the host as defined in the Host NVMe Qualified Name (HOSTNQN) field in Figure 299.</p>
767:512	M	<p><b>Transport Address (TRADDR):</b> If the Entity Type (ETYPE) field in the Discovery Information Management command data portion's header is set to 2h (i.e., a DDC is performing the registration, de-registration, or update task), then this field specifies the address of a fabric interface on the NVM subsystem that may be used for a Connect command as an ASCII string as defined in the Transport Address (TRADDR) field in Figure 294.</p> <p>If the ETYPE field in the Discovery Information Management command data portion's header is set to 1h or 3h (i.e., a host or CDC is performing the registration, de-registration, or update task), then this field specifies the address of a fabric interface on the host that may be used for a Connect command as an ASCII string as defined in the Transport Address (TRADDR) field in Figure 299.</p> <p>If the first byte (i.e., byte 512) of this field is NULL (i.e., cleared to 00h), then the TRADDR value used by the controller shall be the remote IP address associated with the connection used to transport the Discovery Information Management command. If the host attempts to register or de-register multiple discovery information entries with the first byte of this field containing a NULL, then the controller shall abort the command with a status code of Invalid Field in Command.</p>

**Figure 498: Extended Discovery Information Entry**

Bytes	O/M <sup>1</sup>	Description
1023:768	M	<b>Transport Specific Address Subtype (TSAS):</b> This field specifies NVMe Transport specific information about the address as defined in the Transport Specific Address Subtype (TSAS) field in Figure 294.
1027:1024	M	<b>Total Entry Length (TEL):</b> This field specifies the length in bytes of the entire extended discovery information entry.
1029:1028	M <sup>3</sup>	<b>Number of Extended Attributes (NUMEXAT):</b> This field specifies the number of extended attributes contained in the extended discovery information entry.  This field shall be set to a non-zero value (i.e., the extended discovery information entry shall contain at least one extended attribute) if the Entity Type (ETYPE) field in the Discovery Information Registration command data portion's header is set to 1h or 3h (i.e., a host or CDC is performing the register, de-register, or update task). If the ETYPE field is set to 1h or 3h and this field is cleared to 0h, then the controller shall abort the command with a status code of Invalid Discovery Information.
1031:1030		Reserved
<b>Extended Attribute List</b>		
((EXATLEN - 1) + 4) + 1032: 1032	M <sup>3</sup>	<b>Extended Attribute 0:</b> This field contains the first extended attribute as defined in Figure 499 (if present), where EXATLEN is the size specified in the Extended Attribute Length (EXATLEN) field of the extended attribute.
...	O	...
TEL - 1: TEL - (EXATLEN + 4)	O	<b>Extended Attribute N:</b> This field contains the Nth extended attribute as defined in Figure 499 (if present), where EXATLEN is the size specified in the Extended Attribute Length (EXATLEN) field of the extended attribute and TEL is the size specified in the Total Entry Length (TEL) field.
<p>Notes:</p> <ol style="list-style-type: none"> <li>O/M definition: O = Optional, M = Mandatory.</li> <li>This field is mandatory for NVM subsystem discovery information entries. For host discovery information entries, this field shall be cleared to 0h.</li> <li>This field is mandatory for host discovery information entries and optional for NVM subsystem discovery information entries.</li> </ol>		

**Figure 499: Extended Attribute**

Bytes	Description																					
01:00	<p><b>Extended Attribute Type (EXATTYPE):</b> This field specifies the type of extended attribute for the extended attribute contained in the extended discovery information entry.</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Value</th> <th>O/M<sup>1</sup></th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0h</td> <td></td> <td>Reserved</td> </tr> <tr> <td>1h</td> <td>HM</td> <td>Host Identifier</td> </tr> <tr> <td>2h</td> <td>O</td> <td>Admin label ASCII</td> </tr> <tr> <td>3h</td> <td>O</td> <td>Admin label UTF-8</td> </tr> <tr> <td>4h to FEFFh</td> <td></td> <td>Reserved</td> </tr> <tr> <td>FF00h to FFFFh</td> <td>O</td> <td>Vendor Specific</td> </tr> </tbody> </table>	Value	O/M <sup>1</sup>	Definition	0h		Reserved	1h	HM	Host Identifier	2h	O	Admin label ASCII	3h	O	Admin label UTF-8	4h to FEFFh		Reserved	FF00h to FFFFh	O	Vendor Specific
Value	O/M <sup>1</sup>	Definition																				
0h		Reserved																				
1h	HM	Host Identifier																				
2h	O	Admin label ASCII																				
3h	O	Admin label UTF-8																				
4h to FEFFh		Reserved																				
FF00h to FFFFh	O	Vendor Specific																				

**Figure 499: Extended Attribute**

Bytes	Description								
03:02	<p><b>Extended Attribute Length (EXATLEN):</b> This field specifies the length of the Extended Attribute Value (EXATVAL) field for the extended attribute contained in the extended discovery information entry. The length specified in this field shall be a non-zero value that is a multiple of four, and is either a fixed length or within a variable range based upon the value set in the Extended Attribute Type (EXATYPE) field. If the length specified in this field is not a multiple of four, then the controller shall abort the command with a status code of Invalid Field in Command.</p> <table border="1"> <thead> <tr> <th>Extended Attribute Type</th> <th>Length</th> </tr> </thead> <tbody> <tr> <td>Host Identifier</td> <td>16 bytes</td> </tr> <tr> <td>Admin label ASCII</td> <td rowspan="2">4 to 256 bytes</td> </tr> <tr> <td>Admin label UTF-8</td> </tr> </tbody> </table>	Extended Attribute Type	Length	Host Identifier	16 bytes	Admin label ASCII	4 to 256 bytes	Admin label UTF-8	
Extended Attribute Type	Length								
Host Identifier	16 bytes								
Admin label ASCII	4 to 256 bytes								
Admin label UTF-8									
(EXATLEN - 1) + 4:04	<p><b>Extended Attribute Value (EXATVAL):</b> This field specifies the value for the extended attribute contained in the extended discovery information entry. The value specified in this field is based upon the value set in the Extended Attribute Type (EXATYPE) field. Unused bytes, if any, shall be cleared to 0h.</p> <table border="1"> <thead> <tr> <th>Extended Attribute Type</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>Host Identifier</td> <td>This value specifies the Host Identifier of the host, as defined in section 5.1.25.1.28.</td> </tr> <tr> <td>Admin label ASCII</td> <td>This value specifies the admin label of the host or NVM subsystem encoded in ASCII. The admin label is used to identify a host or an NVM subsystem.</td> </tr> <tr> <td>Admin label UTF-8</td> <td>This value specifies the admin label of the host or NVM subsystem encoded in UTF-8. The admin label is used to identify a host or an NVM subsystem.</td> </tr> </tbody> </table>	Extended Attribute Type	Definition	Host Identifier	This value specifies the Host Identifier of the host, as defined in section 5.1.25.1.28.	Admin label ASCII	This value specifies the admin label of the host or NVM subsystem encoded in ASCII. The admin label is used to identify a host or an NVM subsystem.	Admin label UTF-8	This value specifies the admin label of the host or NVM subsystem encoded in UTF-8. The admin label is used to identify a host or an NVM subsystem.
Extended Attribute Type	Definition								
Host Identifier	This value specifies the Host Identifier of the host, as defined in section 5.1.25.1.28.								
Admin label ASCII	This value specifies the admin label of the host or NVM subsystem encoded in ASCII. The admin label is used to identify a host or an NVM subsystem.								
Admin label UTF-8	This value specifies the admin label of the host or NVM subsystem encoded in UTF-8. The admin label is used to identify a host or an NVM subsystem.								
<p>Notes:</p> <p>1. O/M definition: O = Optional, M = Mandatory, HM = Mandatory for hosts and prohibited for NVM subsystems.</p>									

**Figure 500: Extended Discovery Information Entry – Applicable Fields**

Field Name	Extended Discovery Information Entry Type:	
	Host	NVM Subsystem
Transport Type (TRTYPE)	Used	Used
Address Family (ADRFAM)	Used	Used
Subsystem Type (SUBTYPE)	Ignored <sup>1</sup>	Used
Transport Requirements (TREQ)	Ignored <sup>1</sup>	Used
Port ID (PORTID)	Ignored <sup>1</sup>	Used
Controller ID (CNTLID)	Ignored <sup>1</sup>	Used
Admin Max SQ Size (ASQSZ)	Ignored <sup>1</sup>	Used
Transport Service Identifier (TRVCID)	Ignored <sup>1</sup>	Used
NVMe Qualified Name (NQN)	Used	Used
Transport Address (TRADDR)	Used	Used
Transport Specific Address Subtype (TSAS)	Used	Used
Total Entry Length (TEL)	Used	Used
Number of Extended Attributes (NUMEXAT)	Used	Used
Extended Attribute 0 (EXAT0)...Extended Attribute N (EXATN)	Used	Used
<p>Notes:</p> <p>1. This field shall be cleared to 0h.</p>		

### 5.3.3.1 Command Completion

Upon completion of the Discovery Information Management command, the controller posts a completion queue entry to the Admin Completion Queue indicating the status of the command. Discovery Information Management command specific status values are defined in Figure 501.

**Figure 501: Discovery Information Management – Command Specific Status Values**

Value	Definition
2Fh	<b>Invalid Discovery Information:</b> The discovery information provided in one or more extended discovery information entries is not applicable for the type of entity selected in the Entity Type (ETYPE) field of the Discovery Information Management command data portion's header. Refer to Figure 500 for restrictions, if any, that apply to each field in the extended discovery information entry based upon the type of extended discovery information entry being registered, de-registered, or updated.
32h	<b>Insufficient Discovery Resources:</b> The number of discovery information entries provided in the data portion of the Discovery Information Management command for a registration task (i.e., TAS field cleared to 0h) exceeds the available capacity for new discovery information entries on the CDC or DDC. This may be a transient condition.

### 5.3.4 Fabric Zoning Lookup command

The Fabric Zoning Lookup (FZL) command is used to lookup a key associated with a Zoning data structure in the CDC. The FZL command uses the Data Pointer field, as shown in Figure 502.

**Figure 502: Fabric Zoning Lookup (FZL) – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field.

#### 5.3.4.1 Command Completion

Upon completion of the Fabric Zoning Lookup (FZL) command, the controller posts a completion queue entry to the Admin Completion Queue indicating the status of the command. Command specific status values for the FZL command are defined in Figure 503.

**Figure 503: Fabric Zoning Lookup (FZL) – Command Specific Status Values**

Value	Definition
30h	<b>Zoning Data Structure Locked:</b> The requested Zoning data structure is locked on the CDC.
31h	<b>Zoning Data Structure Not Found:</b> The requested Zoning data structure does not exist on the CDC.
33h	<b>Requested Function Disabled:</b> Fabric Zoning is not enabled on the CDC.
34h	<b>ZoneGroup Originator Invalid:</b> The DDC is not allowed to access the specified ZoneGroup.

The key associated with the Zoning data structure that matches the specified FZL data (refer to section 8.3.2.3.8) is returned in the Completion Queue Entry Dword 0, as shown in Figure 504.

**Figure 504: Fabric Zoning Lookup (FZL) – Completion Queue Entry Dword 0**

Bits	Description
31:00	<b>Zoning Data Key (ZDK):</b> The key associated with the Zoning data structure that matches the specified FZL data.

### 5.3.5 Fabric Zoning Receive command

The Fabric Zoning Receive (FZR) command is used to receive a Zoning data structure. The FZR command uses the Data Pointer, Command Dword 10, Command Dword 11, and Command Dword 12 fields, as shown in Figure 505, Figure 506, Figure 507, and Figure 508 respectively.

**Figure 505: Fabric Zoning Receive (FZR) – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field.

**Figure 506: Fabric Zoning Receive (FZR) – Command Dword 10**

Bits	Description
31:00	<b>Zoning Data Key (ZDK):</b> If the FZR command is issued by a DDC, then this field specifies the key identifying a Zoning data structure in the Zoning database of the CDC. If the FZR command is issued by the CDC, then this field specifies the Transaction ID of the current Zoning operation

**Figure 507: Fabric Zoning Receive (FZR) – Command Dword 11**

Bits	Description
31:00	<p><b>Zoning Data Offset (ZDO):</b> This field specifies the byte offset within a Zoning data structure to store the transferred data.</p> <p>The offset shall be dword aligned, indicated by bits 1:0 being cleared to 00b. The controller is not required to check that bits 1:0 are cleared to 00b. The controller may return a status code of Invalid Field in Command if bits 1:0 are not cleared to 00b. If the controller does not return a status code of Invalid Field in Command, then the controller shall operate as if bits 1:0 are cleared to 00b.</p> <p>If an offset greater than the size of the requested Zoning data structure is specified, then the controller shall abort the command with a status code of Invalid Field in Command.</p>

**Figure 508: Fabric Zoning Receive (FZR) – Command Dword 12**

Bits	Description
31:29	Reserved
28	<b>ZDK Context (ZDKC):</b> This bit specifies the content of the ZDK field. If this bit is set to '1', then the ZDK field contains a Transaction ID. If this bit is cleared to '0', then the ZDK field contains a Zoning data key. A CDC receiving a FZR command with this bit set to '1' shall abort the command with a status code of Invalid Field in Command. A DDC receiving a FZR command with this bit cleared to '0' shall abort the command with a status code of Invalid Field in Command.
27:00	<b>Number of Dwords (NUMD):</b> This field specifies the number of dwords to transfer. If this field is cleared to 0h, then the controller shall abort the command with a status code of Invalid Field in Command.

### 5.3.5.1 Command Completion

Upon completion of the Fabric Zoning Receive (FZR) command, the controller posts a completion queue entry to the Admin Completion Queue indicating the status of the command. Command specific status values for the FZR command are defined in Figure 509.

**Figure 509: Fabric Zoning Receive (FZR) – Command Specific Status Values**

Value	Definition
30h	<b>Zoning Data Structure Locked:</b> The requested Zoning data structure is locked on the CDC.
31h	<b>Zoning Data Structure Not Found:</b> The requested Zoning data structure does not exist on the CDC.
33h	<b>Requested Function Disabled:</b> Fabric Zoning is not enabled on the CDC.
34h	<b>ZoneGroup Originator Invalid:</b> The DDC is not allowed to access the specified ZoneGroup.

The last fragment indication is returned in Dword 0 of the completion queue entry, as defined in Figure 510.

**Figure 510: Fabric Zoning Receive (FZR) – Completion Queue Entry Dword 0**

Bits	Description
31	<b>Last Fragment (LF):</b> This bit specifies if the transferred data buffer contains the last fragment of this Zoning data structure. If this bit is set to '1', then the transferred data buffer contains the last fragment. If this bit is cleared to '0', then the transferred data buffer does not contain the last fragment.
30:00	Reserved

### 5.3.6 Fabric Zoning Send command

The Fabric Zoning Send (FZS) command is used to send a Zoning data structure. The FZS command uses the Data Pointer, Command Dword 10, Command Dword 11, and Command Dword 12 fields, as shown in Figure 511, Figure 512, Figure 513, and Figure 514 respectively.

**Figure 511: Fabric Zoning Send (FZS) – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field.

**Figure 512: Fabric Zoning Send (FZS) – Command Dword 10**

Bits	Description
31:00	<b>Zoning Data Key (ZDK):</b> If the FZS command is issued by a DDC, then this field specifies the key identifying a Zoning data structure in the Zoning database of the CDC. If the FZS command is issued by the CDC, then this field specifies the Transaction ID of the current Zoning operation.

**Figure 513: Fabric Zoning Send (FZS) – Command Dword 11**

Bits	Description
31:00	<b>Zoning Data Offset (ZDO):</b> This field specifies the byte offset within a Zoning data structure to store the transferred data.  The offset shall be dword aligned, indicated by bits 1:0 being cleared to 00b. The controller is not required to check that bits 1:0 are cleared to 00b. The controller may return a status code of Invalid Field in Command if bits 1:0 are not cleared to 00b. If the controller does not return a status code of Invalid Field in Command, then the controller shall operate as if bits 1:0 are cleared to 00b.

**Figure 514: Fabric Zoning Send (FZS) – Command Dword 12**

Bits	Description
31	<b>Last Fragment (LF):</b> This bit specifies if the transferred data buffer contains the last fragment of this Zoning data structure. If this bit is set to '1', then the transferred data buffer contains the last fragment. If this bit is cleared to '0', then the transferred data buffer does not contain the last fragment.
30:29	Reserved
28	<b>ZDK Context (ZDKC):</b> This bit specifies the content of the ZDK field. If this bit is set to '1', then the ZDK field contains a Transaction ID. If this bit is cleared to '0', then the ZDK field contains a Zoning data key. A CDC receiving a FZS command with this bit set to '1' shall abort the command with a status code of Invalid Field in Command. A DDC receiving a FZS command with this bit cleared to '0' shall abort the command with a status code of Invalid Field in Command.
27:00	<b>Number of Dwords (NUMD):</b> This field specifies the number of dwords to transfer. If this field is cleared to 0h, then the controller shall abort the command with a status code of Invalid Field in Command.

### 5.3.6.1 Command Completion

Upon completion of the Fabric Zoning Send (FZS) command, the controller posts a completion queue entry to the Admin Completion Queue indicating the status of the command. Command specific status values for the FZS command are defined in Figure 515.

**Figure 515: Fabric Zoning Send (FZS) – Command Specific Status Values**

Value	Definition
30h	<b>Zoning Data Structure Locked:</b> The requested Zoning data structure is locked on the CDC.
31h	<b>Zoning Data Structure Not Found:</b> The requested Zoning data structure does not exist on the CDC.
33h	<b>Requested Function Disabled:</b> Fabric Zoning is not enabled on the CDC.
34h	<b>ZoneGroup Originator Invalid:</b> The DDC is not allowed to access the specified ZoneGroup.

### 5.3.7 Manage Exported Namespace command

The Manage Exported Namespace command is used to manage associations of Exported Namespaces with Exported NVM Subsystems. The Manage Exported Namespace command uses the Data Pointer and Command Dword 10. All other command specific fields are reserved.

The Select field defined in Figure 517 determines which management operation is to be performed in this command. The specified management operation determines the data structure used as part of the command. The data structure is 4,096 bytes in size (refer to subsections below for description of operation specific data structures).

The Manage Exported Namespace command shall not be supported by Exported NVM Subsystems.

**Figure 516: Manage Exported Namespace – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field.

**Figure 517: Manage Exported Namespace – Command Dword 10**

Bits	Description															
31:08	Reserved															
07:00	<b>Select (SEL):</b> This field selects the type of management operation to perform.															
	<table border="1"> <thead> <tr> <th>Value</th> <th>Management Operation</th> <th>Reference Section</th> </tr> </thead> <tbody> <tr> <td>00h</td> <td>Reserved</td> <td></td> </tr> <tr> <td>01h</td> <td>Associate Namespace</td> <td>5.3.7.1.1</td> </tr> <tr> <td>02h</td> <td>Disassociate Namespace</td> <td>5.3.7.1.2</td> </tr> <tr> <td>03h to FFh</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Value	Management Operation	Reference Section	00h	Reserved		01h	Associate Namespace	5.3.7.1.1	02h	Disassociate Namespace	5.3.7.1.2	03h to FFh	Reserved	
	Value	Management Operation	Reference Section													
	00h	Reserved														
	01h	Associate Namespace	5.3.7.1.1													
02h	Disassociate Namespace	5.3.7.1.2														
03h to FFh	Reserved															

#### 5.3.7.1 Manage Exported Namespace Management Operations

##### 5.3.7.1.1 Associate Namespace (Management Operation 01h)

The Associate Namespace operation of the Manage Exported Namespace command is used to create an association between an Exported Namespace ID (ENSID) and an Underlying Namespace; this command also associates the ENSID with an Exported NVM Subsystem identified by the Exported NVM Subsystem NQN.

The Underlying Namespace to be associated with the Exported Namespace ID is returned in the Underlying Namespace List as an Underlying Namespace List Entry (refer to Figure 334) and identified with an Underlying Controller ID, Underlying NVM Subsystem, and Underlying Namespace ID combination. That information is specified in an Associate Namespace data structure (refer to Figure 518).

The Manage Exported Namespace command with an Associate Namespace operation does not attach the Exported Namespace to a controller. The Exported Namespace becomes an attached namespace when



attached to a controller due to the processing of a Namespace Attachment command (refer to section 5.1.20).

The data pointer shall point to an Associate Namespace data structure (refer to Figure 518).

**Figure 518: Associate Namespace Data Structure**

Bytes	Description
31:00	<b>Exported Namespace ID (ENSID):</b> A valid and unassociated Exported NSID to be associated to an Underlying Namespace and associated with the specified Exported NVM Subsystem.
253:32	<b>Exported NVM Subsystem NQN (ENSNQN):</b> Identifies an existing Exported NVM Subsystem to be associated with the Exported Namespace ID.
285:254	<b>Underlying Namespace ID (UNSID):</b> An active Namespace ID to which Exported Namespace ID (ENSID) is to be associated. Refer to Figure 65 for the definition of NSID types and relationship to namespaces.
287:286	<b>Underlying Controller ID (UCTRLID):</b> Identifies the Underlying Controller used to access the specified Underlying Namespace.
543:288	<b>Underlying NVM Subsystem NQN (UNSNQN):</b> Identifies the Underlying NVM Subsystem which provides the Underlying Namespace to be associated with the Exported Namespace ID.
575:544	Reserved

The controller shall abort the command with a status code of Invalid Field in Command if:

- a valid Exported Namespace ID is not specified in the Exported Namespace ID field in Figure 518;
- the Exported NVM Subsystem NQN field in Figure 518 does not refer to an existing Exported NVM Subsystem;
- the Underlying Namespace ID field in Figure 518 does not refer to an active Underlying Namespace ID;
- the Underlying Controller ID field in Figure 518 does not identify a controller contained in the Underlying NVM Subsystem specified by the Underlying NVM Subsystem NQN field;
- the Underlying Namespace specified by the Underlying Namespace ID field shown in Figure 518 is not attached to the controller specified by the Underlying Controller ID in the Underlying Controller ID field in Figure 518;
- the specified Underlying NVM Subsystem NQN field in Figure 518 does not refer to an existing Underlying NVM Subsystem; or
- the specified Underlying Namespace ID is not an allocated namespace in the Underlying NVM Subsystem NQN field in Figure 518.

Upon successful completion of a Manage Exported Namespace command with an Associate Namespace operation:

- the Exported NSID which is associated with an attached Underlying Namespace is an allocated namespace to the Exported NVM Subsystem. The Exported Namespace and the Underlying Namespace contain the same user data (e.g., format, read, and write operations on the Exported Namespace affect user data stored in the Underlying Namespace; similarly, format, read and write operations on the Underlying Namespace affect user data stored in the Exported Namespace); and
- a Namespace Attribute Changed asynchronous event is reported as described in Figure 151.

### 5.3.7.1.2 Disassociate Namespace (Management Operation 02h)

The Disassociate Namespace operation of the Manage Exported Namespace command is used to remove an association between an Exported Namespace ID (ENSID) and an Exported NVM Subsystem and delete the Exported Namespace ID.

The data pointer shall point to a Disassociate Namespace data structure (refer to Figure 519).

**Figure 519: Disassociate Namespace Data Structure**

Bytes	Description
31:00	<b>Exported Namespace ID (ENSID):</b> A specified Exported NSID associated with the specified Exported NVM Subsystem
287:32	<b>Exported NVM Subsystem NQN (ENSNQN):</b> Specifies the Exported NVM Subsystem NQN that is associated with the specified Exported Namespace ID specified in the ENSID field.
319:288	Reserved

If the Exported Namespace is attached to any controller, then the controller processing the command shall abort the command with a status code of Command Sequence Error.

If the Disassociate Namespace data structure does not contain an Exported NVM Subsystem NQN (SUBNQN) for an existing Exported NVM Subsystem associated with the specified Exported Namespace ID and an Exported NSID associated with the specified Exported NVM Subsystem, then the controller shall abort the command with a status code of Invalid Field in Command.

Upon successful completion of a Manage Exported Namespace command with a Disassociate Namespace operation:

- the specified Exported Namespace ID has been deleted; and
- a Namespace Attribute Changed asynchronous event is reported as described in Figure 151.

### 5.3.7.2 Command Completion

Upon completion of the Manage Exported Namespace command, the controller posts a completion queue entry to the Admin Completion queue indicating the status of the command. Refer to section 8.3.3 for usages.

### 5.3.8 Manage Exported NVM Subsystem command

The Manage Exported NVM Subsystem command is used to configure and manage an Exported NVM Subsystem.

The Manage Exported NVM Subsystem command uses the Data Pointer and Command Dword 10. All other command specific fields are reserved.

The Select field defined in Figure 521 determines which management operation is to be performed by this command. The specified management operation determines the data structure used as part of the command. The data structure is 4,096 bytes in size. Refer to section 5.3.8.1 for a description of each management operation.

The Manage Exported NVM Subsystem command shall not be supported by Exported NVM Subsystems.

**Figure 520: Manage Exported NVM Subsystem – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer Figure 92 for the definition of this field.

**Figure 521: Manage Exported NVM Subsystem– Command Dword 10**

Bits	Description																					
31:16	Reserved																					
15:08	<b>Management Operation Specific (MOS):</b> If not defined for the management operation specified by the Select field, this field is reserved.																					
07:00	<b>Select (SEL):</b> This field selects the type of management operation to perform.																					
	<table border="1"> <thead> <tr> <th>Value</th> <th>Management Operation</th> <th>Reference Section</th> </tr> </thead> <tbody> <tr> <td>00h</td> <td>Reserved</td> <td></td> </tr> <tr> <td>01h</td> <td>Delete</td> <td>5.3.8.1.1</td> </tr> <tr> <td>02h</td> <td>Change Access Mode</td> <td>5.3.8.1.2</td> </tr> <tr> <td>03h</td> <td>Grant Host Access</td> <td>5.3.8.1.3</td> </tr> <tr> <td>04h</td> <td>Revoke Host Access</td> <td>5.3.8.1.4</td> </tr> <tr> <td>05h to FFh</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Value	Management Operation	Reference Section	00h	Reserved		01h	Delete	5.3.8.1.1	02h	Change Access Mode	5.3.8.1.2	03h	Grant Host Access	5.3.8.1.3	04h	Revoke Host Access	5.3.8.1.4	05h to FFh	Reserved	
	Value	Management Operation	Reference Section																			
	00h	Reserved																				
	01h	Delete	5.3.8.1.1																			
	02h	Change Access Mode	5.3.8.1.2																			
	03h	Grant Host Access	5.3.8.1.3																			
04h	Revoke Host Access	5.3.8.1.4																				
05h to FFh	Reserved																					

### 5.3.8.1 Manage Exported NVM Subsystem Management Operations

#### 5.3.8.1.1 Delete (Management Operation 01h)

The Delete operation of the Manage Exported NVM Subsystem command is used to delete a specified Exported NVM Subsystem. The data buffer for the Delete operation of the Manage Exported NVM Subsystem command contains an NVM Subsystem NQN specifying the Exported NVM Subsystem to be deleted.

The Management Operation Specific field in Command Dword 10 is reserved and not used by this operation.

If a Manage Exported NVM Subsystem command is processed that specifies the Delete operation without specifying an existing Exported NVM Subsystem identified by the NVM Subsystem NQN (SUBNQN) in the data buffer, then the controller shall abort the command with a status code of Invalid Field in Command.

If there are:

- any active controllers in the specified Exported NVM Subsystem;
- any associations of Exported Namespaces in the specified Exported NVM Subsystem; or
- any Exported Ports exist in the specified Exported NVM Subsystem,

then the controller shall abort the command with a status code of Command Sequence Error.

Upon successful completion of a Manage Exported NVM Subsystem command with a Delete operation the Exported NVM Subsystem identified shall be deleted.

#### 5.3.8.1.2 Change Access Mode (Management Operation 02h)

The Change Access Mode operation of the Manage Exported NVM Subsystem command is used to change the access mode of an Exported NVM Subsystem to:

- allow all hosts access to the specified Exported NVM Subsystem, or
- restrict access to specified hosts listed in the Allowed Host List associated to the specified Exported NVM Subsystem.

The data buffer for the Change Access Mode operation of the Manage Exported NVM Subsystem command specifies an NVM Subsystem NQN indicating the Exported NVM Subsystem for which the Access Mode is to be changed. The Management Operation Specific field in Command Dword 10 for the Change Access Mode operation is shown in Figure 522.

**Figure 522: Management Operation Specific: Change Access Mode operation**

Bits	Description
15:09	Reserved

**Figure 522: Management Operation Specific: Change Access Mode operation**

Bits	Description
08	<b>Restricted Access (RA):</b> A value of '0' configures this Exported NVM Subsystem for unrestricted access and may be accessed by any host. A value of '1' configures this Exported NVM Subsystem to restrict access to Host NQNs present in the Allowed Host List.

The Restricted Access bit defined in Figure 522 specifies the access setting for the Exported NVM Subsystem:

- A value of '0' in the Restricted Access bit configures this Exported NVM Subsystem with Unrestricted Access. This value shall enable all hosts to access the specified Exported NVM Subsystem.
- A value of '1' in the Restricted Access bit shall restrict access of Exported NVM Subsystem to entries in the Allowed Host List associated to the specified Exported NVM Subsystem. Any connected host not in the Allowed Host List associated to the specified Exported NVM Subsystem shall be disconnected from all Exported Namespaces in the Exported NVM Subsystem.

If a Manage Exported NVM Subsystem command is processed that specifies the Change Access Mode operation without specifying an existing Exported NVM Subsystem in the NVM Subsystem NQN (SUBNQN) field, then the controller shall abort the command with a status code of Invalid Field in Command.

Upon successful completion of a Manage Exported NVM Subsystem command with a Change Access Mode operation, the access mode for the specified Exported NVM Subsystem shall be set to the value specified in the Restricted Access bit in the Management Operation Specific field.

#### 5.3.8.1.3 Grant Host Access (Management Operation 03h)

The Grant Host Access operation of the Manage Exported NVM Subsystem command is used to grant hosts access to Exported NVM Subsystems by adding the specified hosts to the Allowed Host List associated to the specified Exported NVM Subsystem. The data pointer shall point to a Subsystem Management data structure (refer to Figure 523). The Management Operation Specific field in Command Dword 10 is reserved and not used by this operation.

The Grant Host Access operation may be used to grant access from a list of hosts (refer to the Host Entry List and "M" in Figure 523) to a list of Exported NVM Subsystems (refer to the Exported NVM Subsystem Entry List and "N" in Figure 523) through specified port(s) (refer to Port ID of the Underlying Port in Figure 523) on each of the Exported NVM Subsystems.

**Figure 523: Subsystem Management Data Structure**

Bytes	Description
63:00	Reserved
65:64	<b>Number of Host Entries (NUMHENT):</b> Specifies the number of Host Entries in this data structure. This field shall be greater than 0h. The value of this field is represented as M.
67:66	<b>Number of Exported NVM Subsystem Entries (NUMENSE):</b> Specifies the number of Exported NVM Subsystem Entries in this data structure. This field shall be greater than 0h. The value of this field is represented as N.
255:68	Reserved
Host Entry List	
575:256	<b>Host Entry 1:</b> Contains a Host Entry data structure as defined by Figure 524.
895:576	<b>Host Entry 2:</b> Contains a Host Entry data structure as defined by Figure 524, if any.
...	...
$320*M+255:320*(M-1)+256$	<b>Host Entry M:</b> Contains a Host Entry data structure as defined by Figure 524, if any.
Exported NVM Subsystem Entry List	
$320+(320*M+255):$ $320*(N-1)+(320*M+256)$	<b>Exported NVM Subsystem Entry 1:</b> Contains an Exported NVM Subsystem Entry data structure as defined by Figure 525.
...	...
$320*N+(320*M+255):$ $320*(N-1)+(320*M+256)$	<b>Exported NVM Subsystem Entry N:</b> Contains an Exported NVM Subsystem Entry data structure as defined by Figure 525, if any.

**Figure 524: Host Entry Data Structure**

Bytes	Description
07:00	Reserved
23:08	<b>Host Identifier (HOSTID):</b> This field has the same definition as the Host Identifier defined in Figure 438.
279:24	<b>Host NVMe Qualified Name (HOSTNQN):</b> NVMe Qualified Name (NQN) that uniquely identifies the host. Refer to section 4.7.
319:280	Reserved

**Figure 525: Exported NVM Subsystem Entry Data Structure**

Bytes	Description
23:00	Reserved
279:24	<b>NVM Subsystem NVMe Qualified Name (SUBNQN):</b> Uniquely identifies an Exported NVM Subsystem. The Allowed Host List associated to this Exported NVM Subsystem is being modified. Refer to section 4.7.
281:280	<b>Port ID of the Underlying Port (PIDUP):</b> Refer to the PORTID field in Figure 294.
319:282	Reserved

The controller shall abort the command with a status code of Invalid Field in Command if:

- the Number of Host Entries field in the Subsystem Management Data Structure (refer to Figure 523) is cleared to 0h; or
- the Number of Exported NVM Subsystem Entries field in the Subsystem Management Data Structure (refer to Figure 523) is cleared to 0h.

Command specific status values associated with the Grant Host Access operation of the Manage Exported NVM Subsystem command are defined in Figure 526. For failures, the byte offset of a failing entry shall be reported in the Command Specific Information field of the Error Information Log Entry.

**Figure 526: Modify Host Access – Command Operation Specific Status Values**

Value	Definition
35h	Invalid Host
36h	Invalid NVM Subsystem

#### 5.3.8.1.4 Revoke Host Access (Management Operation 04h)

The Revoke Host Access operation of the Manage Exported NVM Subsystem command is used to revoke access to the specified Exported NVM Subsystems from a list of specified hosts. The data pointer shall point to a Subsystem Management data structure (refer to Figure 523). The Management Operation Specific field in Command Dword 10 is reserved and not used by this operation.

The Revoke Host Access operation may be used to revoke 1 to N Hosts access from:

- a specified Exported NVM Subsystem through 1 to M specified ports; or
- N different Exported NVM Subsystems through a specified port on each Exported NVM Subsystem.

Connected Hosts that have access revoked from a specified Exported NVM Subsystem on a specified Exported Port shall be disconnected from the specified Exported NVM Subsystem(s) on the specified Exported port.

Command specific status values associated with the Revoke Host Access operation of the Manage Exported NVM Subsystem command are defined in Figure 526. For failures, the byte offset of a failing entry shall be reported in the Command Specific Information field of the Error Information Log Entry.

#### 5.3.8.2 Command Completion

Upon completion of the Manage Exported NVM Subsystem command, the controller posts a completion queue entry to the Admin Completion Queue indicating the status for the command. Reference sections 5.3.8.1.1 through 5.3.8.1.4 for operation specific command completion details. Refer to section 8.3.3 for usages.

#### 5.3.9 Manage Exported Port command

The Manage Exported Port command is used to manage associations of Exported Ports with Exported NVM Subsystems. The Manage Exported Port command uses the Data Pointer and Command Dword 10. All other command specific fields are reserved.

The Select field defined in Figure 528 determines which management operation is to be performed in this command. The specified management operation determines the data structure used as part of the command. The data structure is 4,096 bytes in size.

The Manage Exported Port command shall not be supported by Exported NVM Subsystems.

**Figure 527: Manage Exported Port – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field.

**Figure 528: Manage Exported Port Data Structure– Command Dword 10**

Bits	Description															
31:16	Reserved															
15:08	<b>Management Operation Specific (MOS):</b> If not defined for the management operation specified by the Select field, this field is reserved.															
07:00	<b>Select (SEL):</b> This field selects the type of management operation to perform.															
	<table border="1"> <thead> <tr> <th>Value</th> <th>Management Operation</th> <th>Reference Section</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Reserved</td> <td></td> </tr> <tr> <td>01h</td> <td>Create</td> <td>5.3.9.1.1</td> </tr> <tr> <td>02h</td> <td>Delete</td> <td>5.3.9.1.2</td> </tr> <tr> <td>03h to FFh</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Value	Management Operation	Reference Section	00b	Reserved		01h	Create	5.3.9.1.1	02h	Delete	5.3.9.1.2	03h to FFh	Reserved	
	Value	Management Operation	Reference Section													
	00b	Reserved														
	01h	Create	5.3.9.1.1													
02h	Delete	5.3.9.1.2														
03h to FFh	Reserved															

### 5.3.9.1 Manage Exported Port Management Operations

#### 5.3.9.1.1 Create (Management Operation 01h)

The Create operation of the Manage Exported Port command is used to create an Exported Port in an Exported NVM Subsystem and associate the Exported Port with an Underlying Port in the Ports List. The resulting association enables the specified Exported NVM Subsystem to be accessed by hosts through the specified Exported Port. The Exported Port ID for the Exported Port that is created may be specified by the host or generated by the controller.

The data pointer shall point to an Create data structure (refer to Figure 530).

**Figure 529: Management Operation Specific: Create operation**

Bits	Description
15:09	Reserved
08	<b>Generate Exported Port ID (GEPID):</b> A value of '0' indicates an Exported Port ID is specified in the data buffer and shall be used to create the new Exported Port. A value of '1' specifies a Port ID shall be generated by the controller and used to identify the Exported Port.

**Figure 530: Create Data Structure**

Bytes	Description
255:00	<b>Exported NVM Subsystem NVMe Qualified Name (SUBNQN):</b> NVMe Qualified Name (NQN) that uniquely identifies the Exported NVM subsystem which shall be associated with the created Exported Port. Refer to section 4.7.
257:256	<b>Exported Port ID (EPID):</b> If the Generate Exported Port ID bit in the Create operation is cleared to '0', then this field specifies a particular NVM subsystem port to be used for this Exported Transport and associated with the Exported NVM Subsystem identified in the NVM Subsystem NVMe Qualified Name field in this data structure. Different NVMe Transports or address families may utilize the same Port ID value (e.g., a Port ID may support both iWARP and RoCE). If the Generate Exported Port ID bit in the Create operation is set to '1', then this field shall be ignored by the controller.
259:258	<b>Port ID of the Underlying Port (PIDUD):</b> Refer to the PORTID field in Figure 294.
291:260	<b>Transport Service ID (TRSVCID):</b> Specifies the NVMe Transport service identifier as an ASCII string. The NVMe Transport service identifier is specified by the associated NVMe Transport specification.
319:292	Reserved

Upon successful completion of a Manage Exported Port command with a Create operation, Dword 0 of the completion queue entry of the Manage Exported Port command contains the Exported Port ID associated with the Exported NVM Subsystem (refer to Figure 531).

**Figure 531: Create Completion Queue Entry Dword 0 Data Structure**

Bytes	Description
01:00	<b>Exported Port ID (EPID):</b> If the Generate Exported Port ID bit is set to '1' in the Create operation, then this field specifies Exported Port ID to be used for this Exported Transport and associated with the Exported NVM Subsystem identified by the Exported NVM Subsystem NVMe Qualified Name field in the Create data structure. If the Generate Exported Port ID bit is cleared to '0' in the Create operation, this field shall be set to the Exported Port ID provided in the Create data structure.
03:02	Reserved

If the Exported NVM Subsystem NVMe Qualified Name field specified in the Create Data Structure does not identify an existing Exported NVM Subsystem, then the controller shall abort the command with a status code of Invalid Field in Command.

If the Generate Exported Port ID bit is cleared to '0' in the Create data structure indicating an Exported Port ID is specified in the Manage Exported Port data structure and used to identify the new Exported Port; and:

- an Exported Port ID is not specified in the Manage Exported Port data structure in the data buffer; or
- an Exported Port ID specified in the data buffer does not uniquely identify an Exported NVM Subsystem Port (i.e., if the specified Exported Port ID is used to identify an already existing Exported NVM Subsystem Port),

then the controller shall abort the command with a status code of Invalid Field in Command.

#### 5.3.9.1.2 Delete (Management Operation 02h)

The Delete operation of the Manage Exported Port command is used to remove an Exported Port from an Exported NVM Subsystem and delete the Exported Port.

The Exported Port to be deleted is specified in the Exported Port ID (EPID) field in the Delete operation data structure (refer to Figure 532) in the data buffer.

The Exported NVM Subsystem NQN to which this Exported Port ID is assigned is specified in the NVM Subsystem NVMe Qualified Name (SUBNQN) field of the Delete operation data structure in the data buffer.

The Exported Port that is deleted by a Delete operation should not be in use (i.e., there should be no association between any host and a controller through that Exported Port). If the Exported Port is in use, then all associations between any host and a controller through that port are terminated by the Delete operation, and for any association that is terminated, the behavior of outstanding commands submitted via that association and resources underlying that association (e.g., fabric connections) is undefined.

**Figure 532: Delete Data Structure**

Bytes	Description
255:00	<b>NVM Subsystem NVMe Qualified Name (SUBNQN):</b> NVMe Qualified Name (NQN) that uniquely identifies the Exported NVM subsystem which is associated with the Exported Port to be deleted in command. Refer to section 4.7.
257:256	<b>Exported Port ID (EPID):</b> Specifies a particular NVM subsystem port to be deleted in this command.
319:258	Reserved

If:

- the Exported port ID field in the Delete operation data structure does not contain an Exported Port ID associated with the Exported NVM Subsystem NQN (SUBNQN) specified in the Delete data structure (refer to Figure 532); or
- the NVM Subsystem NVMe Qualified Name field in the Delete data structure (refer to Figure 532) does not contain an NVM Subsystem NQN for an existing Exported NVM Subsystem that is associated with the Exported Port ID specified in the Delete data structure,

then the controller shall abort the command with a status code of Invalid Field in Command.



Upon successful completion of a Manage Exported Port command with a Delete operation the specified Exported Port ID has been removed from the specified Exported NVM Subsystem.

### 5.3.9.2 Command Completion

Upon completion of the Manage Exported Port command, the controller posts a completion queue entry to the Admin Completion queue indicating the status of the command. Refer to section 8.3.3 for usages.

### 5.3.10 Send Discovery Log Page command

The Send Discovery Log Page (SDLP) command is used by a CDC to send a discovery log page to a pull model DDC. This command uses the Data Pointer, Command Dword 10, Command Dword 11, Command Dword 12, and Command Dword 13, as shown in Figure 533, Figure 534, Figure 535, Figure 536, and Figure 537 respectively.

**Figure 533: Send Discovery Log Page (SDLP) – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the start of the data buffer. Refer to Figure 92 for the definition of this field.

**Figure 534: Send Discovery Log Page (SDLP) – Command Dword 10**

Bits	Description										
31:30	<b>Requested Log Page Status (RLPS):</b> This field specified the status of the requested log page. Defined values are:										
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td><b>Valid Log Page:</b> the requested log page is carried in the command.</td> </tr> <tr> <td>01b</td> <td><b>Invalid Log Page:</b> The requested log page is invalid or not supported.</td> </tr> <tr> <td>10b</td> <td><b>Not Allowed Log Page:</b> The requested log page is not allowed to be transferred by an SDLP command (refer to Figure 538).</td> </tr> <tr> <td>11b</td> <td><b>Not Successful:</b> Retrieving the requested log page failed. Further details are provided in the SCT field and the SC field.</td> </tr> </tbody> </table>	Value	Definition	00b	<b>Valid Log Page:</b> the requested log page is carried in the command.	01b	<b>Invalid Log Page:</b> The requested log page is invalid or not supported.	10b	<b>Not Allowed Log Page:</b> The requested log page is not allowed to be transferred by an SDLP command (refer to Figure 538).	11b	<b>Not Successful:</b> Retrieving the requested log page failed. Further details are provided in the SCT field and the SC field.
	Value	Definition									
	00b	<b>Valid Log Page:</b> the requested log page is carried in the command.									
01b	<b>Invalid Log Page:</b> The requested log page is invalid or not supported.										
10b	<b>Not Allowed Log Page:</b> The requested log page is not allowed to be transferred by an SDLP command (refer to Figure 538).										
11b	<b>Not Successful:</b> Retrieving the requested log page failed. Further details are provided in the SCT field and the SC field.										
29:28	Reserved										
27:25	<b>Status Code Type (SCT):</b> Refer to Figure 100.										
24:17	<b>Status Code (SC):</b> Refer to Figure 100.										
16:15	Reserved										
14:08	<b>Transferred Log Specific Parameter (TLSP):</b> If not defined for the log page specified by the Log Page Identifier field, this field is reserved.										
07:00	<b>Transferred Log Page Identifier (TLID):</b> This field specifies the identifier of the sent log page.										

**Figure 535: Send Discovery Log Page (SDLP) – Command Dword 11**

Bits	Description
31:00	<b>Number of Dwords (NDWS):</b> This field specifies the number of transferred dwords.

**Figure 536: Send Discovery Log Page (SDLP) – Command Dword 12**

Bits	Description
31:00	<b>Transferred Log Page Offset Lower (TLPOL):</b> This field specifies the least-significant 32 bits of the offset of the transferred log page.

**Figure 537: Send Discovery Log Page (SDLP) – Command Dword 13**

Bits	Description
31:00	<b>Transferred Log Page Offset Upper (TLPOU):</b> This field specifies the most-significant 32 bits of the offset of the transferred log page.

The transferred log page has the same format of the correspondent discovery log page retrieved by a Get Log Page command (refer to section 5.1.12). The discovery log pages allowed to be transferred by a SDLP command are shown in Figure 538.

**Figure 538: Send Discovery Log Page (SDLP) – Allowed Log Page Identifiers**

Log Page Identifier	Log Page Name
70h	Discovery
71h	Host Discovery
72h	AVE Discovery

If a not allowed log page is requested, the CDC shall set the RLPS field (refer to Figure 534) to 02h (i.e., Not Allowed Log Page) and transfer no log page.

The pull model DDC receiving a SDLP command may request the CDC to resend the requested log page when that log page is updated on the CDC. This is done through the LPUR bit of the SDLP Completion Queue Entry Dword 0, as shown in Figure 539.

**Figure 539: Send Discovery Log Page (SDLP) – Completion Queue Entry Dword 0**

Bits	Description
31	<b>Log Page Update Registration (LPUR):</b> This bit specifies if the CDC shall resend the requested log page when that log page is updated on the CDC. If this bit is set to '1', then the CDC shall issue to the requesting pull model DDC a Send Discovery Log Page command including the requested log page when that log page is updated on the CDC. If this bit is cleared to '0', then the CDC is not required to issue a Send Discovery Log Page command when that log page is updated on the CDC. This registration is retained until the CDC issues the subsequent Send Discovery Log Page command to that requesting pull model DDC.
30:00	Reserved

## 6 Fabrics Command Set

Fabrics commands are used to create queues and initialize a controller. Fabrics commands have an Opcode field of 7Fh and are distinguished by the Fabrics Command Type as shown in Figure 540. Fabrics commands are processed regardless of the state of controller enable (CC.EN). The Fabrics command capsule is defined in section 3.3.2.1.1 and the Fabrics response capsule and status is defined in section 3.3.2.1.2. The common Fabrics Submission Queue entry is shown in Figure 94 and the common Fabrics Completion Queue entry is shown in Figure 99.

Restrictions on processing commands listed in Figure 540 are defined in the Admin Command Set in section 5 (e.g., while the NVM subsystem is performing a sanitize operation or processing of a Format NVM command).

**Figure 540: Fabrics Command Type**

Some Fabric Command Type by Field		Combined Fabrics Command Type <sup>2</sup>	O/M <sup>1</sup>	I/O Queue <sup>3</sup>	Command
(07:02)	(01:00)				
Function	Data Transfer <sup>4</sup>				
0000 00b	00b	00h	M	No	Property Set
0000 00b	01b	01h	M	Yes	Connect <sup>5</sup>
0000 01b	00b	04h	M	No	Property Get
0000 01b	01b	05h	O	Yes	Authentication Send
0000 01b	10b	06h	O	Yes	Authentication Receive
0000 10b	00b	08h	O	Yes	Disconnect
<b>Vendor Specific</b>					
11xx xxb	Note 4	C0h to FFh	O		Vendor specific

Notes:

- O/M definition: O = Optional, M = Mandatory.
- Fabrics Command Types not listed are reserved.
- All Fabrics commands, other than the Disconnect command, may be submitted on the Admin Queue. The I/O Queue supports Fabrics commands as specified in this column. If a Fabrics command that is not supported on an I/O Queue is sent on an I/O Queue, that command shall be aborted with a status code of Invalid Field in Command.
- 00b = no data transfer; 01b = host to controller; 10b = controller to host; 11b = reserved. Refer to the Transfer Direction field in Figure 94.
- The Connect command is submitted and completed on the same queue that the Connect command creates. Refer to section 3.3.2.2.

### 6.1 Authentication Receive Command and Response

The Authentication Receive command transfers the status and data result of one or more Authentication Send commands that were previously submitted to the controller.

The association between an Authentication Receive command and previous Authentication Send commands is dependent on the Security Protocol. The format of the data to be transferred is dependent on the Security Protocol. Refer to SPC-5 for Security Protocol details.

Authentication Receive commands return the appropriate data corresponding to an Authentication Send command as defined by the rules of the Security Protocol. The Authentication Receive command data shall not be retained if there is a loss of communication between the controller and host, or if a Controller Level Reset occurs.

**Figure 541: Authentication Receive Command – Submission Queue Entry**

Bytes	Description
03:00	<b>Command Dword 0 (CDW0):</b> Refer to Figure 95.
04	<b>Fabrics Command Type (FCTYPE):</b> Set to 06h to specify an Authentication Receive command.
23:05	Reserved

**Figure 541: Authentication Receive Command – Submission Queue Entry**

Bytes	Description
39:24	<b>SGL Descriptor 1 (SGL1):</b> Refer to Figure 94.
40	Reserved
41	<b>SP Specific 0 (SPSP0):</b> The value of this field contains bits 07:00 of the Security Protocol Specific field as defined in SPC-5.
42	<b>SP Specific 1 (SPSP1):</b> The value of this field contains bits 15:08 of the Security Protocol Specific field as defined in SPC-5.
43	<b>Security Protocol (SECP):</b> This field specifies the security protocol as defined in SPC-5. The controller shall abort the command with Invalid Parameter indicated if a reserved value of the Security Protocol is specified.
47:44	<b>Allocation Length (AL):</b> The value of this field is specific to the Security Protocol as defined in SPC-5 where INC_512 is cleared to '0'.
63:48	Reserved

**Figure 542: Authentication Receive Response**

Bytes	Description						
07:00	Reserved						
09:08	<b>SQ Head Pointer (SQHD):</b> Indicates the current Submission Queue Head pointer for the associated Submission Queue.						
11:10	Reserved						
13:12	<b>Command Identifier (CID):</b> Indicates the identifier of the command that is being completed.						
15:14	<b>Status Info (STS):</b> Indicates status for the command.						
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:01</td> <td><b>Status (STATUS):</b> The Status field for the command. Refer to section 4.2.3.</td> </tr> <tr> <td>00</td> <td>Reserved</td> </tr> </tbody> </table>	Bits	Description	15:01	<b>Status (STATUS):</b> The Status field for the command. Refer to section 4.2.3.	00	Reserved
	Bits	Description					
15:01	<b>Status (STATUS):</b> The Status field for the command. Refer to section 4.2.3.						
00	Reserved						

## 6.2 Authentication Send Command and Response

The Authentication Send command is used to transfer security protocol data to the controller. The data structure transferred as part of this command contains security protocol specific commands to be performed by the controller. The data structure may contain data or parameters associated with the security protocol specific commands. Status and data that is to be returned to the host for the security protocol specific commands submitted by an Authentication Send command are retrieved with the Authentication Receive command defined in section 6.1.

The association between an Authentication Send command and subsequent Authentication Receive commands is Security Protocol field dependent as defined in SPC-5.

**Figure 543: Authentication Send Command – Submission Queue Entry**

Bytes	Description
03:00	<b>Command Dword 0 (CDW0):</b> Refer to Figure 95.
04	<b>Fabrics Command Type (FCTYPE):</b> Set to 05h to specify an Authentication Send command.
23:05	Reserved
39:24	<b>SGL Descriptor 1 (SGL1):</b> Refer to Figure 94.
40	Reserved
41	<b>SP Specific 0 (SPSP0):</b> The value of this field contains bits 07:00 of the Security Protocol Specific field as defined in SPC-5.
42	<b>SP Specific 1 (SPSP1):</b> The value of this field contains bits 15:08 of the Security Protocol Specific field as defined in SPC-5.
43	<b>Security Protocol (SECP):</b> This field specifies the security protocol as defined in SPC-5. The controller shall abort the command with a status code of Invalid Parameter indicated if a reserved value of the Security Protocol is specified.

**Figure 543: Authentication Send Command – Submission Queue Entry**

Bytes	Description
47:44	<b>Transfer Length (TL):</b> The value of this field is specific to the Security Protocol as defined in SPC-5 where INC_512 is cleared to '0'.
63:48	Reserved

**Figure 544: Authentication Send Response**

Bytes	Description						
07:00	Reserved						
09:08	<b>SQ Head Pointer (SQHD):</b> Indicates the current Submission Queue Head pointer for the associated Submission Queue.						
11:10	Reserved						
13:12	<b>Command Identifier (CID):</b> Indicates the identifier of the command that is being completed.						
15:14	<b>Status Info (STS):</b> Indicates status for the command.						
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:01</td> <td><b>Status (STATUS):</b> The Status field for the command. Refer to section 4.2.3.</td> </tr> <tr> <td>00</td> <td>Reserved</td> </tr> </tbody> </table>	Bits	Description	15:01	<b>Status (STATUS):</b> The Status field for the command. Refer to section 4.2.3.	00	Reserved
	Bits	Description					
15:01	<b>Status (STATUS):</b> The Status field for the command. Refer to section 4.2.3.						
00	Reserved						
00	Reserved						

### 6.3 Connect Command and Response

The Connect command is used to create a Submission and Completion Queue pair. If the Admin Queue is specified, then the Connect command establishes an association between a host and a controller. The fields for the submission queue entry are defined in Figure 545 and the fields for the data portion are defined in Figure 546.

A host that uses a single Host NQN may employ multiple Host Identifiers to designate elements of the host that access an NVM subsystem independently of each other (e.g., physical or logical partitions of the host). Alternatively, a host may employ multiple Host NQN values to cause each element to be treated as a separate host by an NVM subsystem.

If an NVM subsystem supports DH-HMAC-CHAP authentication (refer to section 8.3.4), then the Host NQN and the NVM Subsystem NQN parameters in a Connect command are required to be different. If the Host NQN and the NVM Subsystem NQN parameters in a Connect command are identical and the NVM subsystem supports DH-HMAC-CHAP authentication, then the controller shall abort the command with a status code of Connect Invalid Host.

The NVM subsystem shall not allocate a Controller ID in the range FFF0h to FFFFh as a valid Controller ID on completion of a Connect command. Controller IDs FFF0h to FFFFh are defined in Figure 27. If the host is not allowed to establish an association to any controller in the NVM subsystem, then the controller shall abort the command with a status code of Connect Invalid Host.

If the NVM subsystem supports the dynamic controller model, then:

- the Controller ID of FFFFh is specified as the Controller ID in a Connect command for the Admin Queue. If the controller ID is not set to FFFFh, then the controller shall abort the command with a status code of Connect Invalid Parameters;
- the NVM subsystem shall allocate any available controller to the host; and
- return that allocated Controller ID in the Connect response.

If the NVM subsystem supports the static controller model, then:

- The host is able to request a specific controller in a Connect command for the Admin Queue. If the host is not allowed to establish an association to the specified controller, then the controller shall abort the command with a status code of Connect Invalid Host;
- The Controller ID of FFFEh on the Admin Queue specifies that any Controller ID may be allocated and returned in the Connect response; and

- If the host specifies a Controller ID value of FFFFh for the Admin Queue, then the controller shall abort the command with a status code of Connect Invalid Parameters.

The NVM subsystem may allocate specific controllers to particular hosts. If a host requests a controller that is not allocated to that host, then the controller shall abort the command with a status code of Connect Invalid Host. The mechanism for allocating specific controllers to particular hosts is outside the scope of this specification.

The host shall establish an association with a controller and enable the controller before establishing a connection with an I/O Queue of the controller. If the host sends a Connect command specifying a Queue ID for an Admin Queue or I/O Queue that has already been created, then the controller shall abort the command with a status code of Command Sequence Error.

The Controller shall abort a Connect command with a status code of Connect Invalid Parameters if:

- the host sends a Connect command to create an I/O Queue while the controller is disabled;
- the Host NQN, NVM Subsystem NQN, and the Controller ID values specified for an I/O Queue are not the same as the values specified for the associated Admin Queue in which the association between the host and controller was established;
- the Host Identifier for an I/O Queue is not set to a value of 0h and is not set to the same value as the value specified for the associated Admin Queue in which the association between the host and controller was established;
- the Host NQN or NVM Subsystem NQN values do not match the values that the NVM subsystem is configured to support;
- there is a syntax error in the Host NQN or NVM Subsystem NQN value (refer to section 4.7); or
- the host specifies a Controller ID value in the range FFF0h to FFFDh.

If the NVMe Subsystem Port, NVMe Transport Type or NVMe Transport Address used by the NVMe Transport (refer to section 6.3) are not the same as the values used for the associated Admin Queue in which the association between the host and controller was established, then it is possible that the Connect command is not received by an NVM subsystem. If the Connect command is received by an NVM subsystem, then:

- the NVM subsystem that receives the command may not be the same NVM subsystem to which the association between the host and controller was established (i.e., the NVMe Transport Type and NVMe Transport Address are unique to an NVM Subsystem Port); and
- the values of the NVM Subsystem NQN or Controller ID may not be valid at that NVM Subsystem Port (e.g., the NVM Subsystem NQN may specify a different NVM subsystem than the one that received that Connect command, or the Controller ID may specify a controller that is already bound to a different NVM Subsystem Port).

If this situation occurs and the Connect command is aborted, then the status code shall be set to Connect Invalid Parameters. There is no requirement that such a Connect command be received by an NVM subsystem (e.g., if the NVMe Transport Address is not a valid transport address, or is the address of a fabric endpoint that does not support NVMe over Fabrics, then the resulting error, if any, is specific to the fabric).

Submission Queue (SQ) flow control based on the SQ Head Pointer (SQHD) field in Fabrics response capsules (refer to section 3.3.2.1.2) shall be supported by all hosts and controllers. Use of SQ flow control is negotiated by the Connect command and response. A host requests that SQ flow control be disabled by setting the Disable SQ Flow Control (DISSQFC) bit of the Connect Attributes field to '1' in a Connect command. A controller that agrees to disable SQ flow control shall set the SQHD field to FFFFh in the response to that Connect command. A controller that does not agree to disable SQ flow control shall set the SQHD field to a value other than FFFFh in the response to that Connect command.

If the Connect command did not request that SQ flow control be disabled, then the controller shall not set the SQHD field to FFFFh in the response to that Connect command.

SQ flow control is disabled and shall not be used for a created queue pair only if:

- a) the DISSQFC bit is set to '1' in the Connect command that creates the queue pair; and

- b) the SQHD field is set to FFFFh in the response to that Connect command.

If SQ flow control is disabled, then the SQHD field is reserved in Fabrics response capsules for all command completions on that queue pair after the response that completes the Connect command.

SQ flow control is enabled and shall be used for a created queue pair if:

- a) the DISSQFC bit is cleared to '0' in the Connect command that creates the queue pair; or
- b) the SQHD field is not set to FFFFh in the response to that Connect command.

If SQ flow control is enabled, then the controller shall use the SQHD field in Fabrics response capsules for all command completions on that queue pair, except for command completions that omit the SQHD value due to use of the SQHD pointer update optimization described in section 3.3.2.7.

**Figure 545: Connect Command – Submission Queue Entry**

Bytes	Description
03:00	<b>Command Dword 0 (CDW0):</b> Refer to Figure 95.
04	<b>Fabrics Command Type (FCTYPE):</b> Set to 01h to specify a Connect command.
23:05	Reserved
39:24	<b>SGL Descriptor 1 (SGL1):</b> Refer to Figure 94.
41:40	<b>Record Format (RECFMT):</b> Specifies the format of the Connect command capsule. The format of the record specified in this definition shall be 0h. If the NVM subsystem does not support the value specified, then a status code of Incompatible Format shall be returned.
43:42	<b>Queue ID (QID):</b> Specifies the Queue Identifier for the Admin Queue or I/O Queue to be created. The identifier is used for both the Submission and Completion Queue. The identifier for the Admin Submission Queue and Admin Completion Queue is 0h. The identifier for an I/O Submission and Completion Queue is in the range 1 to 65,534.  If the value in this field specifies the Queue ID of a queue that already exists, then the controller shall abort the command with a status code of Invalid Queue Identifier.
45:44	<b>Submission Queue Size (SQSIZE):</b> This field indicates the size of the Submission Queue to be created. If the size is 0h or larger than the controller supports, then a status code of Connect Invalid Parameters shall be returned. The maximum size of the Admin Submission Queue is specified in the Discovery Log Page Entry for the NVM subsystem. Refer to Figure 294. This is a 0's based value.

Figure 545: Connect Command – Submission Queue Entry

Bytes	Description																						
46	<p><b>Connect Attributes (CATTR):</b> This field indicates attributes for the connection.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:5</td> <td>Reserved</td> </tr> <tr> <td>4</td> <td><b>Connecting Entity (CONNENT):</b> Indicates the type of entity performing the Connect command. If this bit is set to '1', then the entity performing the Connect command is a Discovery controller. If this bit is cleared to '0', then the entity performing the Connect command is a host.</td> </tr> <tr> <td>3</td> <td><b>Individual I/O Queue Deletion Support (INDIVIOQDELS):</b> Indicates support for deleting individual I/O Queues. If this bit is set to '1', then the host supports the deletion of individual I/O Queues. If this bit is cleared to '0', then the host does not support the deletion of individual I/O Queues.</td> </tr> <tr> <td>2</td> <td><b>Disable SQ Flow Control (DISSQFC):</b> If this bit is set to '1', then the host is requesting that SQ flow control be disabled. If this bit is cleared to '0', then SQ flow control shall not be disabled.</td> </tr> <tr> <td>1:0</td> <td> <p><b>Priority Class (PRIOCLASS):</b> Indicates the priority class to use for commands within this Submission Queue. This field is only used when the weighted round robin with urgent priority class is the arbitration mechanism selected (refer to CC.AMS in Figure 41, the field is ignored if weighted round robin with urgent priority class is not used. Refer to section 3.4.4. This field is only valid for I/O Queues and shall be cleared to 00b for Admin Queue connections.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Urgent</td> </tr> <tr> <td>01b</td> <td>High</td> </tr> <tr> <td>10b</td> <td>Medium</td> </tr> <tr> <td>11b</td> <td>Low</td> </tr> </tbody> </table> </td> </tr> </tbody> </table>	Bits	Description	7:5	Reserved	4	<b>Connecting Entity (CONNENT):</b> Indicates the type of entity performing the Connect command. If this bit is set to '1', then the entity performing the Connect command is a Discovery controller. If this bit is cleared to '0', then the entity performing the Connect command is a host.	3	<b>Individual I/O Queue Deletion Support (INDIVIOQDELS):</b> Indicates support for deleting individual I/O Queues. If this bit is set to '1', then the host supports the deletion of individual I/O Queues. If this bit is cleared to '0', then the host does not support the deletion of individual I/O Queues.	2	<b>Disable SQ Flow Control (DISSQFC):</b> If this bit is set to '1', then the host is requesting that SQ flow control be disabled. If this bit is cleared to '0', then SQ flow control shall not be disabled.	1:0	<p><b>Priority Class (PRIOCLASS):</b> Indicates the priority class to use for commands within this Submission Queue. This field is only used when the weighted round robin with urgent priority class is the arbitration mechanism selected (refer to CC.AMS in Figure 41, the field is ignored if weighted round robin with urgent priority class is not used. Refer to section 3.4.4. This field is only valid for I/O Queues and shall be cleared to 00b for Admin Queue connections.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Urgent</td> </tr> <tr> <td>01b</td> <td>High</td> </tr> <tr> <td>10b</td> <td>Medium</td> </tr> <tr> <td>11b</td> <td>Low</td> </tr> </tbody> </table>	Value	Definition	00b	Urgent	01b	High	10b	Medium	11b	Low
	Bits	Description																					
	7:5	Reserved																					
	4	<b>Connecting Entity (CONNENT):</b> Indicates the type of entity performing the Connect command. If this bit is set to '1', then the entity performing the Connect command is a Discovery controller. If this bit is cleared to '0', then the entity performing the Connect command is a host.																					
	3	<b>Individual I/O Queue Deletion Support (INDIVIOQDELS):</b> Indicates support for deleting individual I/O Queues. If this bit is set to '1', then the host supports the deletion of individual I/O Queues. If this bit is cleared to '0', then the host does not support the deletion of individual I/O Queues.																					
2	<b>Disable SQ Flow Control (DISSQFC):</b> If this bit is set to '1', then the host is requesting that SQ flow control be disabled. If this bit is cleared to '0', then SQ flow control shall not be disabled.																						
1:0	<p><b>Priority Class (PRIOCLASS):</b> Indicates the priority class to use for commands within this Submission Queue. This field is only used when the weighted round robin with urgent priority class is the arbitration mechanism selected (refer to CC.AMS in Figure 41, the field is ignored if weighted round robin with urgent priority class is not used. Refer to section 3.4.4. This field is only valid for I/O Queues and shall be cleared to 00b for Admin Queue connections.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>Urgent</td> </tr> <tr> <td>01b</td> <td>High</td> </tr> <tr> <td>10b</td> <td>Medium</td> </tr> <tr> <td>11b</td> <td>Low</td> </tr> </tbody> </table>	Value	Definition	00b	Urgent	01b	High	10b	Medium	11b	Low												
Value	Definition																						
00b	Urgent																						
01b	High																						
10b	Medium																						
11b	Low																						
47	Reserved																						
51:48	<p><b>Keep Alive Timeout (KATO):</b> In the Connect command for the Admin Queue, this field has the same definition as the Keep Alive Timeout (KATO) field of the Keep Alive Timer feature (refer to section 5.1.25.1.8). Upon successful completion of the Connect command, the controller shall set the KATO field in the Keep Alive Timer feature. Refer to section 3.9.2 for a description of activating the Keep Alive Timer.</p> <p>In the Connect command for an I/O Queue, this field is reserved.</p>																						
53:52	<p><b>NVM Set Identifier (NVMSETID):</b> This field indicates the identifier of the NVM Set to be associated with this Submission Queue. This field is only valid for I/O Queues.</p> <p>In the Connect command for an Admin Queue, the:</p> <ul style="list-style-type: none"> <li>• host should clear this field to 0h;</li> <li>• controller shall ignore this field; and</li> <li>• Submission Queue is not associated with any NVM Set.</li> </ul> <p>In the Connect command for an I/O queue, if the SQ Associations capability is not supported (refer to section 8.1.25) or this field is cleared to 0h, then this Submission Queue is not associated with any specific NVM Set.</p> <p>If the SQ Associations capability is supported (refer to section 8.1.25) and this field contains a non-zero value that is not indicated in the NVM Set List (refer to Figure 317), then the controller shall abort the command with a status code of Invalid Field in Command.</p> <p>The host should not submit commands for namespaces associated with other NVM Sets in this Submission Queue (refer to section 8.1.25).</p>																						
63:54	Reserved																						



**Figure 546: Connect Command – Data**

Bytes	Description
15:00	<p><b>Host Identifier (HOSTID):</b> This field has the same definition as the Host Identifier defined in section 5.1.25.1.28.</p> <p>For a Connect command to create an Admin Queue (i.e., the QID field is cleared to 0h) that completes successfully the controller shall set the current Host Identifier (refer to section 5.1.25.1.28.2) to this value.</p> <p>For a Connect command to create an I/O queue (i.e., the QID field is set to a non-zero value):</p> <ul style="list-style-type: none"> <li>if this field is cleared to 0h, then the controller shall ignore this field; and</li> <li>if this field is set to a non-zero value and the value in this field does not match the value in the Host Identifier feature, then the controller shall abort the command with a status code of Connect Invalid Parameters.</li> </ul>
17:16	<p><b>Controller ID (CNTLID):</b> Specifies the controller ID requested. This field corresponds to the Controller ID (CNTLID) value returned in the Identify Controller data structure for a particular controller. If the NVM subsystem uses the dynamic controller model, then the value shall be FFFFh for the Admin Queue and any available controller may be returned. If the NVM subsystem uses the static controller model and the value is FFFEh for the Admin Queue, then any available controller may be returned.</p>
255:18	Reserved
511:256	<p><b>NVM Subsystem NVMe Qualified Name (SUBNQN):</b> NVMe Qualified Name (NQN) that uniquely identifies the NVM subsystem. Refer to section 4.7.</p>
767:512	<p><b>Host NVMe Qualified Name (HOSTNQN):</b> NVMe Qualified Name (NQN) that uniquely identifies the host. Refer to section 4.7.</p>
1023:768	Reserved

The Connect response provides status for the Connect command. If a connection is established, then the Controller ID allocated to the host is returned. The Connect response is defined in Figure 547.

For a Connect command that fails, the controller shall not:

- return a status code of Invalid Field in Command; and
- add an entry to the Error Information log page (refer to section 5.1.12.1.2).

**Figure 547: Connect Response**

Bytes	Description						
03:00	<p><b>Status Code Specific (SCS):</b> The value is dependent on the status returned. Refer to Figure 548.</p>						
07:04	Reserved						
09:08	<p><b>SQ Head Pointer (SQHD):</b> If the Connect command requested that SQ flow control be disabled, then a value of FFFFh in this field indicates that SQ flow control is disabled for the created queue pair. Otherwise, this field indicates the current Submission Queue Head pointer for the associated Submission Queue and also indicates that SQ flow control is enabled for the created queue pair.</p>						
11:10	Reserved						
13:12	<p><b>Command Identifier (CID):</b> Indicates the identifier of the command that is being completed.</p>						
15:14	<p><b>Status Info (STS):</b> Indicates status for the command.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:01</td> <td> <p><b>Status (STATUS):</b> The Status field for the command. Refer to section 4.2.3. Refer to Figure 105 for values specific to the Connect command.</p> </td> </tr> <tr> <td>00</td> <td>Reserved</td> </tr> </tbody> </table>	Bits	Description	15:01	<p><b>Status (STATUS):</b> The Status field for the command. Refer to section 4.2.3. Refer to Figure 105 for values specific to the Connect command.</p>	00	Reserved
	Bits	Description					
	15:01	<p><b>Status (STATUS):</b> The Status field for the command. Refer to section 4.2.3. Refer to Figure 105 for values specific to the Connect command.</p>					
00	Reserved						

Figure 548: Connect Response – Dword 0 Value Based on Status Code

Status Code	Definition of Dword 0		
Successful Completion	<b>Bytes</b>	<b>Description</b>	
	01:00	<b>Controller ID (CNTLID):</b> Specifies the controller ID allocated to the host. If a particular controller was specified in the CNTLID field of the Connect command, then this field shall contain the same value.	
	03:02	<b>Authentication and Security Requirements (AUTHREQ):</b> Specifies the NVMe in-band authentication and security requirements. The field is bit significant. If all bits are cleared to '0', then no requirements are specified.	
		<b>Bits</b>	<b>Description</b>
		15:03	Reserved
02		<b>Authentication and Secure Channel Required (ASCR):</b> If this bit is set to '1', then authentication using NVMe over Fabrics Authentication protocols followed by secure channel establishment is required and the ATR bit should be cleared to '0'. If this bit is cleared to '0', then authentication using NVMe over Fabrics Authentication protocols followed by secure channel establishment is not required.	
01	<b>Authentication Transaction Required (ATR):</b> If this bit is set to '1', then authentication using NVMe over Fabrics Authentication protocols is required. If this bit is cleared to '0', then authentication using NVMe over Fabrics Authentication protocols is not required.		
00	Obsolete.		
Connect Invalid Parameters	<b>Bytes</b>	<b>Description</b>	
	01:00	<b>Invalid Parameter Offset (IPO):</b> If an invalid parameter is reported, then this field specifies the offset in bytes to the invalid parameter from the start of the SQE or the data.	
	02	<b>Invalid Attributes (IATTR):</b> Specifies attributes of the invalid field parameter.	
		<b>Bits</b>	<b>Description</b>
7:1	Reserved		
0	<b>Invalid Parameter Start (IPS):</b> If this bit is cleared to '0', then the invalid parameter is specified from the start of the SQE. If this bit is set to '1', then the invalid parameter is specified from the start of the data.		
03	Reserved		
All Other Status Values	<b>Bytes</b>	<b>Description</b>	
	03:00	Reserved	

#### 6.4 Disconnect Command and Response

The Disconnect command is used to delete the I/O Queue on which the command is submitted. If a Disconnect command is submitted on an Admin Queue, then the controller shall abort the command with a status code of Invalid Queue Type. If the controller is not able to delete the I/O Queue, then the controller shall abort the command with a status code of Controller Busy. The fields for the submission queue entry are defined in Figure 549.

The NVMe Transport connection is not deleted upon issuance of a Disconnect command; the host and controller may delete the NVMe Transport connection and associated resources after all NVMe Queues (I/O Queues and/or Admin Queue) associated with that NVMe Transport connection have been deleted (refer to section 3.3.2.4).

The completion queue entry for the Disconnect command shall be the last entry submitted to the I/O Queue Completion queue by the controller (i.e., no completion queue entries shall be submitted to the I/O Queue Completion Queue after the completion queue entry for the Disconnect command). The controller shall ensure that no further command processing is performed for any command on an I/O queue after sending the completion queue entry for the Disconnect command.

The host should not submit commands to an I/O Submission Queue after the submission of a Disconnect command to that I/O Submission Queue; submitting commands to an I/O Queue after a Disconnect command is submitted to that I/O Queue results in undefined behavior.

**Figure 549: Disconnect Command – Submission Queue Entry**

Bytes	Description
03:00	<b>Command Dword 0 (CDW0):</b> Refer to Figure 95. Byte 01 is cleared to 0h.
04	<b>Fabrics Command Type (FCTYPE):</b> Set to 08h to specify a Disconnect command.
39:05	Reserved
41:40	<b>Record Format (RECFMT):</b> Specifies the format of the Disconnect command capsule. The format of the record specified in this definition shall be 0h. If the NVM subsystem does not support the value specified, then a status code of Incompatible Format shall be returned.
63:42	Reserved

The Disconnect response provides status for the Disconnect command. The Disconnect response is defined in Figure 550.

**Figure 550: Disconnect Response**

Bytes	Description						
07:00	Reserved						
09:08	<b>SQ Head Pointer (SQHD):</b> Indicates the current Submission Queue Head pointer for the associated Submission Queue.						
11:10	Reserved						
13:12	<b>Command Identifier (CID):</b> Indicates the identifier of the command that is being completed.						
15:14	<b>Status Info (STS):</b> Indicates status for the command.						
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:01</td> <td><b>Status (STATUS):</b> The Status field for the command. Refer to section 4.2.2. Refer to Figure 105 for values specific to the Disconnect command.</td> </tr> <tr> <td>00</td> <td>Reserved</td> </tr> </tbody> </table>	Bits	Description	15:01	<b>Status (STATUS):</b> The Status field for the command. Refer to section 4.2.2. Refer to Figure 105 for values specific to the Disconnect command.	00	Reserved
	Bits	Description					
15:01	<b>Status (STATUS):</b> The Status field for the command. Refer to section 4.2.2. Refer to Figure 105 for values specific to the Disconnect command.						
00	Reserved						

## 6.5 Property Get Command and Response

The Property Get command is used to specify the property value to return to the host (refer to section 3.1.4). The fields for the Property Get command are defined in Figure 551. If an invalid property or invalid offset is specified, then a status code of Invalid Field in Command shall be returned.

**Figure 551: Property Get Command – Submission Queue Entry**

Bytes	Description																	
03:00	<b>Command Dword 0 (CDW0):</b> Refer to Figure 95. Byte 01 is cleared to 0h.																	
04	<b>Fabrics Command Type (FCTYPE):</b> Set to 04h to specify a Property Get command.																	
39:05	Reserved																	
40	<b>Attributes (ATTRIB):</b> Specifies attributes for the Property Get command.																	
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:3</td> <td>Reserved</td> </tr> <tr> <td rowspan="4">2:0</td> <td><b>Property Return Size (PRS):</b> This field specifies the size of the property to return. Valid values are shown in the table below.</td> </tr> <tr> <td> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>4 bytes</td> </tr> <tr> <td>001b</td> <td>8 bytes</td> </tr> <tr> <td>010b to 111b</td> <td>Reserved</td> </tr> </tbody> </table> </td> </tr> <tr> <td></td> <td></td> </tr> </tbody> </table>	Bits	Description	7:3	Reserved	2:0	<b>Property Return Size (PRS):</b> This field specifies the size of the property to return. Valid values are shown in the table below.	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>4 bytes</td> </tr> <tr> <td>001b</td> <td>8 bytes</td> </tr> <tr> <td>010b to 111b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	000b	4 bytes	001b	8 bytes	010b to 111b	Reserved		
	Bits	Description																
	7:3	Reserved																
2:0	<b>Property Return Size (PRS):</b> This field specifies the size of the property to return. Valid values are shown in the table below.																	
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>4 bytes</td> </tr> <tr> <td>001b</td> <td>8 bytes</td> </tr> <tr> <td>010b to 111b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	000b	4 bytes	001b	8 bytes	010b to 111b	Reserved									
	Value	Definition																
	000b	4 bytes																
001b	8 bytes																	
010b to 111b	Reserved																	
43:41	Reserved																	
47:44	<b>Offset (OFST):</b> Specifies the offset to the property to get. Refer to section 3.1.4.																	
63:48	Reserved																	

The Property Get response is used to return the value of the property requested to the host. The Property Get response is defined in Figure 552.

**Figure 552: Property Get Response**

Bytes	Description						
07:00	<b>Value (VALUE):</b> Indicates the value returned for the property if the Property Get command is successful. If the size of the property is four bytes, then the value is specified in bytes 03:00 and bytes 07:04 are reserved.						
09:08	<b>SQ Head Pointer (SQHD):</b> Indicates the current Submission Queue Head pointer for the associated Submission Queue.						
11:10	Reserved						
13:12	<b>Command Identifier (CID):</b> Indicates the identifier of the command that is being completed.						
15:14	<b>Status Info (STS):</b> Indicates status for the command.						
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:01</td> <td><b>Status (STATUS):</b> The Status field for the command. Refer to section 4.2.2.</td> </tr> <tr> <td>00</td> <td>Reserved</td> </tr> </tbody> </table>	Bits	Description	15:01	<b>Status (STATUS):</b> The Status field for the command. Refer to section 4.2.2.	00	Reserved
	Bits	Description					
15:01	<b>Status (STATUS):</b> The Status field for the command. Refer to section 4.2.2.						
00	Reserved						

## 6.6 Property Set Command and Response

The Property Set command is used to set the value of a property (refer to section 3.1.4). The fields for the Property Set command are defined in Figure 553. If an invalid property or invalid offset is specified, then a status code of Invalid Field in Command shall be returned.

**Figure 553: Property Set Command – Submission Queue Entry**

Bytes	Description																			
03:00	<b>Command Dword 0 (CDW0):</b> Refer to Figure 95. Byte 01 is cleared to 0h.																			
04	<b>Fabrics Command Type (FCTYPE):</b> Cleared to 00h to specify a Property Set command.																			
39:05	Reserved																			
40	<b>Attributes (ATTRIB):</b> Specifies attributes for the Property Set command.																			
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:3</td> <td>Reserved</td> </tr> <tr> <td rowspan="4">2:0</td> <td><b>Property Update Size (PUS):</b> This field specifies the size of the property to update. Valid values are shown in the table below.</td> </tr> <tr> <td> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>4 bytes</td> </tr> <tr> <td>001b</td> <td>8 bytes</td> </tr> <tr> <td>010b to 111b</td> <td>Reserved</td> </tr> </tbody> </table> </td> </tr> <tr> <td></td> <td></td> </tr> <tr> <td></td> <td></td> </tr> </tbody> </table>	Bits	Description	7:3	Reserved	2:0	<b>Property Update Size (PUS):</b> This field specifies the size of the property to update. Valid values are shown in the table below.	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>4 bytes</td> </tr> <tr> <td>001b</td> <td>8 bytes</td> </tr> <tr> <td>010b to 111b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	000b	4 bytes	001b	8 bytes	010b to 111b	Reserved				
	Bits	Description																		
	7:3	Reserved																		
2:0	<b>Property Update Size (PUS):</b> This field specifies the size of the property to update. Valid values are shown in the table below.																			
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>4 bytes</td> </tr> <tr> <td>001b</td> <td>8 bytes</td> </tr> <tr> <td>010b to 111b</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	000b	4 bytes	001b	8 bytes	010b to 111b	Reserved											
	Value	Definition																		
	000b	4 bytes																		
001b	8 bytes																			
010b to 111b	Reserved																			
43:41	Reserved																			
47:44	<b>Offset (OFST):</b> Specifies the offset to the property to set. Refer to section 3.1.4.																			
55:48	<b>Value (VALUE):</b> Specifies the value used to update the property. If the size of the property is four bytes, then the value is specified in bytes 51:48 and bytes 55:52 are reserved.																			
63:56	Reserved																			

The Property Set response provides status for the Property Set command. The Property Set response is defined in Figure 554.

**Figure 554: Property Set Response**

Bytes	Description
07:00	Reserved
09:08	<b>SQ Head Pointer (SQHD):</b> Indicates the current Submission Queue Head pointer for the associated Submission Queue.
11:10	Reserved
13:12	<b>Command Identifier (CID):</b> Indicates the identifier of the command that is being completed.

**Figure 554: Property Set Response**

Bytes	Description						
15:14	<b>Status Info (STS):</b> Indicates status for the command.						
	<table border="1"> <thead> <tr> <th data-bbox="412 344 586 380">Bits</th> <th data-bbox="586 344 1360 380">Description</th> </tr> </thead> <tbody> <tr> <td data-bbox="412 380 586 407">15:01</td> <td data-bbox="586 380 1360 407"><b>Status (STATUS):</b> Status field for the command. Refer to section 4.2.2.</td> </tr> <tr> <td data-bbox="412 407 586 432">00</td> <td data-bbox="586 407 1360 432">Reserved</td> </tr> </tbody> </table>	Bits	Description	15:01	<b>Status (STATUS):</b> Status field for the command. Refer to section 4.2.2.	00	Reserved
	Bits	Description					
15:01	<b>Status (STATUS):</b> Status field for the command. Refer to section 4.2.2.						
00	Reserved						

## 7 I/O Commands

An I/O command is a command submitted to an I/O Submission Queue. Figure 555 lists the I/O commands that are defined for use in all I/O Command Sets. The following subsections provide definitions for each command. Refer to section 3.1.3.4 for mandatory, optional, and prohibited I/O commands for the various controller types. The following subsections describe the definition for each of these commands.

The user data format and any end-to-end protection information is I/O Command Set specific. Refer to each I/O Command Set specification for applicability and additional details, if any. Refer to the referenced I/O Command Set specification for all I/O Command Set specific commands described in Figure 555.

Commands shall only be submitted by the host when the controller is ready as indicated in the Controller Status property (CSTS.RDY) and after appropriate I/O Submission Queue(s) and I/O Completion Queue(s) have been created.

The submission queue entry (SQE) structure and the fields that are common to all I/O commands are defined in section 4.1. The completion queue entry (CQE) structure and the fields that are common to all I/O commands are defined in section 4.2. 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 commands supported across all I/O Command Sets are defined in this section.

**Figure 555: Opcodes for I/O Commands**

Opcode by Field		Combined Opcode <sup>1</sup>	Command <sup>2</sup>	Reference
(07:02)	(01:00)			
Function	Data Transfer <sup>3</sup>			
0000 00b	00b	00h	Flush <sup>4</sup>	7.2
0000 11b	01b	0Dh	Reservation Register	7.6
0000 11b	10b	0Eh	Reservation Report	7.8
0001 00b	01b	11h	Reservation Acquire	7.5
0001 00b	10b	12h	I/O Management Receive	7.3
0001 01b	01b	15h	Reservation Release	7.7
0001 10b	00b	18h	Cancel <sup>4</sup>	7.1
0001 11b	01b	1Dh	I/O Management Send	7.4
0111 11b	11b <sup>5</sup>	7Fh	Fabric Commands <sup>5</sup>	6
<i>Vendor Specific</i>				
1xxx xxb	NOTE 3	80h to FFh	Vendor specific	
Notes:				
1. Opcodes not listed are I/O Command Set specific or reserved. Refer to Figure 91 for Opcode details.				
2. All I/O commands use the Namespace Identifier (NSID) field. The value FFFFFFFFh is not supported in this field unless footnote 4 in this figure indicates that a specific command does support that value.				
3. 00b = no data transfer; 01b = host to controller; 10b = controller to host; 11b = bidirectional. Refer to the Transfer Direction field in Figure 91.				
4. This command may support the use of the Namespace Identifier (NSID) field set to FFFFFFFFh.				
5. All Fabrics commands use the opcode 7Fh with the direction of data transfer specified as shown in Figure 540. Refer to section 6 for details.				

### 7.1 Cancel command

The Cancel command is used to request an abort for specified commands submitted on the same I/O Submission Queue to which the Cancel command is submitted. The Cancel command may apply to a single namespace or to multiple namespaces. The Cancel command may be deeply queued. Some of the commands to abort may have already completed, currently be processing, or be deeply queued. As a result, the performance of the Cancel command may be impacted.

To abort an Admin command, the host uses the Abort command (refer to section 5.1.1). To abort a Cancel command, the host may:

- use the Abort command;
- use a second Cancel command with a Cancel Action set to Single Command Cancel (i.e., 00b) that specifies the CID of the Cancel command to abort; or
- follow the procedure described in section 3.3.1.3 to delete the I/O Submission Queue and recreate the I/O Submission Queue.

The controller applies the specified Cancel Action (refer to Figure 557) to all outstanding commands that have been fetched prior to the processing of the Cancel command. The controller may or may not apply the specified Cancel Action to commands that are fetched:

- after the start of processing of the Cancel command; and
- before the completion of the Cancel command.

There is no requirement to apply the specified Cancel Action to commands that have not been fetched by the controller.

If an abort action is performed on a command to abort, that action may be:

- an immediate abort (i.e., the abort action occurs prior to posting the completion queue entry for the Cancel command); or
- a deferred abort (i.e., the abort action may or may not occur prior to posting the completion queue entry for the Cancel command).

For each command on which an immediate abort is performed, the controller shall meet the immediate abort requirements (refer to section 5.1.1.1) for that command before posting the completion queue entry for the Cancel command.

For each command on which a deferred abort is performed, there are no requirements on the ordering of posting of the completion queue entry for the Cancel command (refer to section 7.1.1).

For each command on which an abort action is performed (i.e., immediate abort or deferred abort) the status code shall be set to Command Abort Requested in the completion queue entry for that command.

If the Cancel command is supported, then the Commands Supported and Effects log page shall be supported.

The Cancel command uses the Command Dword 10 and Command Dword 11 fields. All other command specific fields are reserved.

**Figure 556: Cancel – Command Dword 10**

Bits	Description
31:16	<p><b>Command Identifier (CID):</b> This field specifies the command identifier of the command to be aborted (i.e., the CDW0.CID field within the command to be aborted).</p> <p>If the Action Code field is set to Single Command Cancel and this field is set to the CID for this Cancel command, then the controller shall abort the command with a status code of Invalid Command ID.</p> <p>If the Action Code field is set to Multiple Command Cancel and this field is not set to FFFFh, then the controller shall abort the command with a status code of Invalid Field in Command.</p>
15:00	<p><b>Submission Queue Identifier (SQID):</b> This field specifies the identifier of the Submission Queue that the Cancel command is associated with.</p> <p>If the SQID does not match the Submission Queue Identifier of the submission queue to which the Cancel command is submitted, then the controller shall abort the command with a status code of Invalid Field in Command.</p>

**Figure 557: Cancel – Command Dword 11**

Bits	Description								
31:02	Reserved								
01:00	<b>Action Code (ACODE):</b>								
	<table border="1"> <thead> <tr> <th>Value</th> <th>Cancel Action</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td> <p><b>Single Command Cancel:</b> The hosts requests that the controller abort the specific command submitted to the specified namespace with the specified CID.</p> <p>If the NSID field is set to FFFFFFFFh then, the host requests that the controller abort the specific command submitted to any NSID with the specified CID.</p> <p>If the NSID is not set to FFFFFFFFh and the specified CID is not associated with the specified NSID then, the controller shall abort the Cancel command with a status code of Invalid Field in Command.</p> <p>If the specified CID is not found, then the controller shall complete the Cancel command with the Commands Eligible for Deferred Abort field cleared to 0h and the Commands Aborted field cleared to 0h.</p> </td> </tr> <tr> <td>01b</td> <td> <p><b>Multiple Command Cancel:</b> The hosts requests that the controller abort all commands, other than this Cancel command, on the I/O Submission Queue to which the Cancel command was submitted that were submitted to the specified NSID.</p> <p>If the NSID is set to FFFFFFFFh then the host requests that the controller abort all commands, other than this Cancel command, submitted to the I/O Submission Queue to which the Cancel command was submitted for all namespaces.</p> </td> </tr> <tr> <td>All others</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Cancel Action	00b	<p><b>Single Command Cancel:</b> The hosts requests that the controller abort the specific command submitted to the specified namespace with the specified CID.</p> <p>If the NSID field is set to FFFFFFFFh then, the host requests that the controller abort the specific command submitted to any NSID with the specified CID.</p> <p>If the NSID is not set to FFFFFFFFh and the specified CID is not associated with the specified NSID then, the controller shall abort the Cancel command with a status code of Invalid Field in Command.</p> <p>If the specified CID is not found, then the controller shall complete the Cancel command with the Commands Eligible for Deferred Abort field cleared to 0h and the Commands Aborted field cleared to 0h.</p>	01b	<p><b>Multiple Command Cancel:</b> The hosts requests that the controller abort all commands, other than this Cancel command, on the I/O Submission Queue to which the Cancel command was submitted that were submitted to the specified NSID.</p> <p>If the NSID is set to FFFFFFFFh then the host requests that the controller abort all commands, other than this Cancel command, submitted to the I/O Submission Queue to which the Cancel command was submitted for all namespaces.</p>	All others	Reserved
	Value	Cancel Action							
	00b	<p><b>Single Command Cancel:</b> The hosts requests that the controller abort the specific command submitted to the specified namespace with the specified CID.</p> <p>If the NSID field is set to FFFFFFFFh then, the host requests that the controller abort the specific command submitted to any NSID with the specified CID.</p> <p>If the NSID is not set to FFFFFFFFh and the specified CID is not associated with the specified NSID then, the controller shall abort the Cancel command with a status code of Invalid Field in Command.</p> <p>If the specified CID is not found, then the controller shall complete the Cancel command with the Commands Eligible for Deferred Abort field cleared to 0h and the Commands Aborted field cleared to 0h.</p>							
01b	<p><b>Multiple Command Cancel:</b> The hosts requests that the controller abort all commands, other than this Cancel command, on the I/O Submission Queue to which the Cancel command was submitted that were submitted to the specified NSID.</p> <p>If the NSID is set to FFFFFFFFh then the host requests that the controller abort all commands, other than this Cancel command, submitted to the I/O Submission Queue to which the Cancel command was submitted for all namespaces.</p>								
All others	Reserved								
All others	Reserved								

### 7.1.1 Command Completion

Upon completion of the Cancel command, the controller posts a completion queue entry to the I/O Completion Queue indicating the status for the Cancel command.

If the Cancel Action (refer to Figure 557) specified a Single Command Cancel Action and the Commands Aborted field is cleared to 0h, then the host should examine the status in the completion queue entry of the command to abort to determine whether the command was aborted or not (i.e., whether a deferred abort was performed or not).

If the Cancel Action specified a Multiple Command Cancel Action, then the host should examine the status in the completion queue entry of each command to abort to determine whether the command was aborted or not.

Cancel command specific status code values are defined in Figure 558.

**Figure 558: Cancel – Command Specific Status Values**

Value	Description
84h	<b>Invalid Command ID:</b> The specified CID matched the CID of this Cancel command.

Dword 0 of the completion queue entry contains information about the number of commands that were aborted by this command. Dword 0 of the completion queue entry is defined in Figure 559.



**Figure 559: Cancel – Completion Queue Entry Dword 0**

Bits	Description
31:16	<p><b>Commands Eligible for Deferred Abort (CEDA):</b> This field indicates the number of commands that match the specified criteria (refer to Figure 556 and Figure 557) on which a deferred abort may be performed. A value of 0h indicates that no commands are eligible for a deferred abort or that the controller does not support deferred aborts. A value of FFFFh indicates that FFFFh or more commands are eligible for a deferred abort.</p> <p>If the Cancel Action (refer to Figure 557) specified a Single Command Cancel and an immediate abort was performed on the specified command, then this field shall be cleared to 0h.</p>
15:00	<p><b>Commands Aborted (CMDA):</b> This field indicates the number of commands on which the controller performed an immediate abort as a result of processing this Cancel command. A value of 0h indicates that the controller did not perform an immediate abort on any commands as a result of processing this Cancel command.</p> <p>If the Cancel Action (refer to Figure 557) specified a Single Command Cancel and an immediate abort was performed on the specified command, then this field shall be set to 1h.</p>

## 7.2 Flush command

The Flush command is used to request that the contents of volatile write cache be made non-volatile.

If a volatile write cache is enabled (refer to section 5.1.25.1.4), then the Flush command shall commit data and metadata associated with the specified namespace(s) to non-volatile storage media. The flush applies to all commands for the specified namespace(s) completed by the controller prior to the submission of the Flush command. The controller may also flush additional data and/or metadata from any namespace.

If the Flush Behavior (FB) field is set to 11b in the VWC field in the Identify Controller data structure (refer to Figure 312) and the specified NSID is FFFFFFFFh, then the Flush command applies to all namespaces attached to the controller processing the Flush command. If the FB field is set to 10b and the specified NSID is FFFFFFFFh, then the controller aborts the command with a status code of Invalid Namespace or Format. If the FB field is cleared to 00b, then the controller behavior if the specified NSID is FFFFFFFFh is not indicated. Controllers compliant with NVM Express Base Specification, Revision 1.4 and later shall not set the FB field to the value of 00b.

If a namespace exists in an Endurance Group that has:

- Flexible Data Placement (refer to section 8.1.10) enabled; and
- the Volatile Write Cache Not Present (VWCNP) bit set to '1' in the I/O Command Set Independent Identify Namespace data structure (refer to Figure 319),

then, even though the Volatile Write Cache field in the Identify Controller data structure (refer to Figure 312) indicates the presence of a volatile write cache in the controller, this namespace does not have a volatile write cache present.

A host may use the Volatile Write Cache Not Present (VWCNP) bit in the I/O Command Set Independent Identify Namespace data structure to determine if a Volatile Write Cache is not present for a namespace.

If a volatile write cache is not present or not enabled, then Flush commands shall have no effect and:

- a) shall complete successfully if a sanitize operation is not in progress; and
- b) may complete successfully if a sanitize operation is in progress.

All command specific fields are reserved.

### 7.2.1 Command Completion

Upon completion of the Flush command, the controller posts a completion queue entry to the associated I/O Completion Queue.

### 7.3 I/O Management Receive command

The I/O Management Receive command is used to receive information from the controller used by the host to manage I/O. The behavior of the command is dependent on the specified operation as defined in the Management Operation field in Figure 561.

The command uses the Data Pointer, Command Dword 10, and Command Dword 11 fields. All other command specific fields are reserved. If the command uses PRPs for the data transfer, then the PRP Entry 1 and PRP Entry 2 fields are used. If the command uses SGLs for the data transfer, then the SGL Entry 1 field is used. All other command specific fields are reserved.

If the Number of Dwords (NUMD) field corresponds to a length that is less than the size of the data structure to be returned, then only that specified portion of the data structure is transferred. If the NUMD field corresponds to a length that is greater than the size of the associated data structure, then the entire contents of the data structure are transferred and no additional data is transferred.

**Figure 560: I/O Management Receive – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the location of a data buffer where data is transferred from. Refer to Figure 92 for the definition of this field.

**Figure 561: I/O Management Receive – Command Dword 10**

Bits	Description										
31:16	<b>Management Operation Specific (MOS):</b> This definition of this field is specific the value of the MO field. If this field is not defined for the management operation specified by the MO field, then this field is reserved.										
15:08	Reserved										
07:00	<b>Management Operation (MO):</b> This field specifies the management operation to perform.										
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00h</td> <td>No action</td> </tr> <tr> <td>01h</td> <td><b>Reclaim Unit Handle Status:</b> For each Placement Handle of the namespace, the controller shall return a Reclaim Unit Handle Status Descriptor for each Reclaim Group.</td> </tr> <tr> <td>FFh</td> <td>Vendor specific</td> </tr> <tr> <td>All others</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00h	No action	01h	<b>Reclaim Unit Handle Status:</b> For each Placement Handle of the namespace, the controller shall return a Reclaim Unit Handle Status Descriptor for each Reclaim Group.	FFh	Vendor specific	All others	Reserved
	Value	Definition									
	00h	No action									
	01h	<b>Reclaim Unit Handle Status:</b> For each Placement Handle of the namespace, the controller shall return a Reclaim Unit Handle Status Descriptor for each Reclaim Group.									
FFh	Vendor specific										
All others	Reserved										

**Figure 562: I/O Management Receive – Command Dword 11**

Bits	Description
31:00	<b>Number of Dwords (NUMD):</b> This field specifies the number of dwords to transfer. This is a 0's based value.

#### 7.3.1 I/O Management Receive Operations

##### 7.3.1.1 Reclaim Unit Handle Status (Management Operation 01h)

The Reclaim Unit Handle Status management operation is used to provide information about Reclaim Unit Handles (refer to section 1.5.84) that are accessible by the specified namespace. The returned data contains one or more Reclaim Unit Handle Status Descriptor data structures (refer to the applicable I/O Command Set specification). The information contained in each Reclaim Unit Handle Status Descriptor:

- is the information at the time the controller processes that Reclaim Unit Handle Status Descriptor; and
- may or may not contain the information reflecting any outstanding command that affects the reported Reclaim Unit Handle Status Descriptor.

If Flexible Data Placement is disabled in the Endurance Group containing the specified namespace, then the command shall be aborted with a status code of FDP Disabled.

If the NSID field is set to 0h or FFFFFFFFh, then the controller shall abort the command with a status code of Invalid Namespace or Format.

If the host reads beyond the size of the Reclaim Unit Handle Status data structure (refer to Figure 563), zeroes are returned.

**Figure 563: Reclaim Unit Handle Status**

Bytes	Description
<b>Header</b>	
13:00	Reserved
15:14	<b>Number of Reclaim Unit Handle Status Descriptors (NRUHSD):</b> This field indicates the number of Reclaim Unit Handle Status Descriptors in the Reclaim Unit Handle Status Descriptor list.
<b>Reclaim Unit Handle Status Descriptor List</b>	
47:16	<b>Reclaim Unit Handle Status Descriptor 1:</b> The first Reclaim Unit Handle Status Descriptor (refer to the applicable I/O Command Set specification).
79:48	<b>Reclaim Unit Handle Status Descriptor 2:</b> The second Reclaim Unit Handle Status Descriptor (refer to the applicable I/O Command Set specification), if any.
...	...
(NRUHSD *32)+15: (NRUHSD *32)-16	<b>Reclaim Unit Handle Status Descriptor NRUHSD:</b> The last Reclaim Unit Handle Status Descriptor (refer to the applicable I/O Command Set specification), if any.

The Reclaim Unit Handle Status Descriptors are defined in the I/O Command Set specifications, if supported.

### 7.3.2 Command Completion

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

### 7.4 I/O Management Send command

The I/O Management Send command is used to manage I/O and the behavior of the command is dependent on the specified operation as defined in the Management Operation field in Figure 565.

The command uses the Data Pointer and Command Dword 10 field. All other command specific fields are reserved. If the command uses PRPs for the data transfer, then the PRP Entry 1 and PRP Entry 2 fields are used. If the command uses SGLs for the data transfer, then the SGL Entry 1 field is used. All other command specific fields are reserved.

**Figure 564: I/O Management Send – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the location of a data buffer where data is transferred from. Refer to Figure 92 for the definition of this field.

**Figure 565: I/O Management Send – Command Dword 10**

Bits	Description
31:16	<b>Management Operation Specific (MOS):</b> The definition of this field is specific to the value of the MO field. If this field is not defined for the management operation specified by the MO field, then this field is reserved.
15:08	Reserved

**Figure 565: I/O Management Send – Command Dword 10**

Bits	Description										
07:00	<b>Management Operation (MO):</b> This field specifies the management operation to perform.										
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00h</td> <td>No action</td> </tr> <tr> <td>01h</td> <td><b>Reclaim Unit Handle Update:</b> Update the reference to the Reclaim Unit specified by each entry in the Placement Identifier list to reference an empty Reclaim Unit (refer to section 3.2.4).</td> </tr> <tr> <td>FFh</td> <td>Vendor specific</td> </tr> <tr> <td>All others</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00h	No action	01h	<b>Reclaim Unit Handle Update:</b> Update the reference to the Reclaim Unit specified by each entry in the Placement Identifier list to reference an empty Reclaim Unit (refer to section 3.2.4).	FFh	Vendor specific	All others	Reserved
	Value	Definition									
	00h	No action									
	01h	<b>Reclaim Unit Handle Update:</b> Update the reference to the Reclaim Unit specified by each entry in the Placement Identifier list to reference an empty Reclaim Unit (refer to section 3.2.4).									
FFh	Vendor specific										
All others	Reserved										

### 7.4.1 I/O Management Send Operations

#### 7.4.1.1 Reclaim Unit Handle Update (Management Operation 01h)

The Reclaim Unit Handle Update management operation for the I/O Management Send command provides a list of Placement Identifiers (refer to Figure 567). The number of Placement Identifiers is defined in the Management Operation Specific field defined in Figure 566. For each Placement Identifier in the list:

- If the currently referenced Reclaim Unit has been written with user data, then the Placement Identifier shall be modified to reference a different Reclaim Unit that is empty (refer to section 3.2.4); and
- If the currently referenced Reclaim Unit has not been written with any user data (i.e., is already empty), then the Placement Identifier may be modified to reference a different Reclaim Unit that is empty.

**Figure 566: Management Operation Specific – Reclaim Unit Handle Update Operation**

Bits	Description
15:00	<p><b>Number of Placement Identifiers (NPID):</b> Indicates the number of Placement Identifiers that are specified in the command. This is a 0's based value.</p> <p>This field shall not exceed the minimum of:</p> <ul style="list-style-type: none"> <li>• the value in the Max Placement Identifiers (MAXPIDS) field of the enabled FDP configuration (refer to Figure 280); and</li> <li>• the product of the Number of Reclaim Groups (NRG) field and the Number of Reclaim Unit Handles (NRUH) field of the enabled FDP configuration (refer to Figure 280);</li> </ul>

If a specified Placement Identifier is invalid due to:

- the value of the Reclaim Group Identifier field being greater than or equal to the Number of Reclaim Groups field of the FDP Configuration Descriptor (refer to Figure 280) for the Endurance Group associated with the specified namespace; or
- the specified Placement Handle field being greater than or equal to the Number of Placement Handles field specified when the namespace was created,

then the controller shall abort the command with a status code of Invalid Field in Command.

If the value represented by the Number of Placement Identifiers (NPID) field is greater than the Max Placement Identifiers (MAXPIDS) field (refer to Figure 280) in the current FDP configuration, then the controller shall abort the command with a status code of Invalid Field in Command.

If the command is aborted, then Placement Identifiers may or may not have been updated.

While processing an I/O Management Send command that specifies the Reclaim Unit Handle Update operation, if the controller processes a write command that utilizes a Placement Identifier specified in the Placement Identifier List of that I/O Management Send command, then the controller may write the user data for that write command to the referenced Reclaim Unit:

- prior to processing that I/O Management Send command; or
- upon the completion of that I/O Management Send command.

**Figure 567: Reclaim Unit Handle Update – Data Buffer**

Bytes	Description
<b>Placement Identifier List</b>	
01:00	<b>Placement Identifier 1:</b> This field specifies the first Placement Identifier that indicates a Placement Handle and a Reclaim Group Identifier. Refer to Figure 282 and Figure 283.
03:02	<b>Placement Identifier 2:</b> This field specifies the second Placement Identifier that indicates a Placement Handle and a Reclaim Group Identifier, if any. Refer to Figure 282 and Figure 283.
...	
(NPID*2)+1: (NPID*2)	<b>Placement Identifier NPID:</b> This field specifies the last Placement Identifier that indicates a Placement Handle and a Reclaim Group Identifier, if any. Refer to Figure 282 and Figure 283.

If Flexible Data Placement is disabled in the Endurance Group containing the specified namespace, then the controller shall abort the command with a status code of FDP Disabled.

#### 7.4.2 Command Completion

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

#### 7.5 Reservation Acquire command

The Reservation Acquire command is used to:

- acquire a reservation on a namespace;
- preempt a reservation held on a namespace; or
- preempt a reservation held on a namespace and abort outstanding commands for that namespace.

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

**Figure 568: Reservation Acquire – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the location of a data buffer where data is transferred from. Refer to Figure 92 for the definition of this field.

**Figure 569: Reservation Acquire – Command Dword 10**

Bits	Description
31:16	Reserved
15:08	<b>Reservation Type (RTYPE):</b> This field specifies the type of reservation to be created. The field is defined in Figure 571.
07:05	Reserved

Figure 569: Reservation Acquire – Command Dword 10

Bits	Description															
04	<p><b>Dispersed Namespace Reservation Support (DISNSRS):</b> This bit specifies host support for reservations on dispersed namespaces for a host that supports dispersed namespaces (i.e., for a host that has set the Host Dispersed Namespace Support (HDISNS) field to 1h in the Host Behavior Support feature).</p> <p>If this bit is set to '1', then the host supports reservations on dispersed namespaces (i.e., the host supports receiving a value of FFFDh in the Controller ID (CNTLID) field of a Registered Controller data structure or Registered Controller Extended data structure (refer to section 8.1.9.6)).</p> <p>If this bit is cleared to '0', then the host does not support reservations on dispersed namespaces (i.e., the host does not support receiving a value of FFFDh in the Controller ID (CNTLID) field of a Registered Controller data structure or Registered Controller Extended data structure (refer to section 8.1.9.6)). If the HDISNS field is set to 1h in the Host Behavior Support feature and the host submits the command to a dispersed namespace with this bit cleared to '0', then the controller aborts the command with a status code of Namespace Is Dispersed as described in section 8.1.9.6.</p>															
03	<p><b>Ignore Existing Key (IEKEY):</b> If this bit is set to a '1', then the controller shall return an error of Invalid Field in Command. If this bit is cleared to '0', then the Current Reservation Key is checked.</p>															
02:00	<p><b>Reservation Acquire Action (RACQA):</b> This field specifies the action that is performed by the command.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> <th>Reference</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>Acquire</td> <td>8.1.22.5</td> </tr> <tr> <td>001b</td> <td>Preempt</td> <td>8.1.22.7</td> </tr> <tr> <td>010b</td> <td>Preempt and Abort</td> <td>8.1.22.7</td> </tr> <tr> <td>011b to 111b</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Value	Definition	Reference	000b	Acquire	8.1.22.5	001b	Preempt	8.1.22.7	010b	Preempt and Abort	8.1.22.7	011b to 111b	Reserved	
Value	Definition	Reference														
000b	Acquire	8.1.22.5														
001b	Preempt	8.1.22.7														
010b	Preempt and Abort	8.1.22.7														
011b to 111b	Reserved															

Figure 570: Reservation Acquire Data Structure

Bytes	Description
07:00	<p><b>Current Reservation Key (CRKEY):</b> The field specifies the current reservation key associated with the host.</p>
15:08	<p><b>Preempt Reservation Key (PRKEY):</b> If the Reservation Acquire Action is set to 001b (i.e., Preempt) or 010b (i.e., Preempt and Abort), then this field specifies the reservation key to be unregistered from the namespace. For all other Reservation Acquire Action values, this field is reserved.</p>

Figure 571: Reservation Type Encoding

Value	Definition
0h	Reserved
1h	Write Exclusive Reservation
2h	Exclusive Access Reservation
3h	Write Exclusive - Registrants Only Reservation
4h	Exclusive Access - Registrants Only Reservation
5h	Write Exclusive - All Registrants Reservation
6h	Exclusive Access - All Registrants Reservation
7h to FFh	Reserved

### 7.5.1 Command Completion

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

### 7.6 Reservation Register command

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

The command uses Command Dword 10 and a Reservation Register data structure in memory (refer to Figure 574). If the command uses PRPs for the data transfer, then PRP Entry 1 and PRP Entry 2 fields are used. If the command uses SGLs for the data transfer, then the SGL Entry 1 field is used. All other command specific fields are reserved.

**Figure 572: Reservation Register – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the location of a data buffer where data is transferred from. Refer to Figure 92 for the definition of this field.

**Figure 573: Reservation Register – Command Dword 10**

Bits	Description															
31:30	<p><b>Change Persist Through Power Loss State (CPTPL):</b> This field allows the Persist Through Power Loss (PTPL) state associated with the namespace to be modified as a side effect of processing this command. If the Reservation Persistence feature (refer to section 5.1.25.1.30) is saveable, then any change to the PTPL state as a result of processing this command shall be applied to both the current value and the saved value of that feature.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00b</td> <td>No change to PTPL state</td> </tr> <tr> <td>01b</td> <td>Reserved</td> </tr> <tr> <td>10b</td> <td>Clear PTPL state to '0'. Reservations are released and registrants are cleared on a power on.</td> </tr> <tr> <td>11b</td> <td>Set PTPL state to '1'. Reservations and registrants persist across a power loss.</td> </tr> </tbody> </table>	Value	Definition	00b	No change to PTPL state	01b	Reserved	10b	Clear PTPL state to '0'. Reservations are released and registrants are cleared on a power on.	11b	Set PTPL state to '1'. Reservations and registrants persist across a power loss.					
Value	Definition															
00b	No change to PTPL state															
01b	Reserved															
10b	Clear PTPL state to '0'. Reservations are released and registrants are cleared on a power on.															
11b	Set PTPL state to '1'. Reservations and registrants persist across a power loss.															
29:05	Reserved															
04	<p><b>Dispersed Namespace Reservation Support (DISNSRS):</b> This bit specifies host support for reservations on dispersed namespaces for a host that supports dispersed namespaces (i.e., for a host that has set the Host Dispersed Namespace Support (HDISNS) field to 1h in the Host Behavior Support feature).</p> <p>If this bit is set to '1', then the host supports reservations on dispersed namespaces (i.e., the host supports receiving a value of FFFDh in the Controller ID (CNTLID) field of a Registered Controller data structure or Registered Controller Extended data structure (refer to section 8.1.9.6)).</p> <p>If this bit is cleared to '0', then the host does not support reservations on dispersed namespaces (i.e., the host does not support receiving a value of FFFDh in the Controller ID (CNTLID) field of a Registered Controller data structure or Registered Controller Extended data structure (refer to section 8.1.9.6)). If the HDISNS field is set to 1h in the Host Behavior Support feature and the host submits the command to a dispersed namespace with this bit cleared to '0', then the controller aborts the command with a status code of Namespace Is Dispersed as described in section 8.1.9.6.</p>															
03	<p><b>Ignore Existing Key (IEKEY):</b> If this bit is set to a '1', then Reservation Register Action (RREGA) field values that use the Current Reservation Key (CRKEY) shall succeed regardless of the value of the Current Reservation Key field in the command (i.e., the current reservation key, if any, is not checked, and absence of a current reservation key does not cause an error).</p>															
02:00	<p><b>Reservation Register Action (RREGA):</b> This field specifies the registration action that is performed by the command.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> <th>Reference</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>Register Reservation Key</td> <td>8.1.22.3</td> </tr> <tr> <td>001b</td> <td>Unregister Reservation Key</td> <td>8.1.22.4</td> </tr> <tr> <td>010b</td> <td>Replace Reservation Key</td> <td>8.1.22.3</td> </tr> <tr> <td>011b to 111b</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Value	Definition	Reference	000b	Register Reservation Key	8.1.22.3	001b	Unregister Reservation Key	8.1.22.4	010b	Replace Reservation Key	8.1.22.3	011b to 111b	Reserved	
Value	Definition	Reference														
000b	Register Reservation Key	8.1.22.3														
001b	Unregister Reservation Key	8.1.22.4														
010b	Replace Reservation Key	8.1.22.3														
011b to 111b	Reserved															

**Figure 574: Reservation Register Data Structure**

Bytes	Description
07:00	<p><b>Current Reservation Key (CRKEY):</b> If the Reservation Register Action is 001b (i.e., Unregister Reservation Key) or 010b (i.e., Replace Reservation Key), then this field contains the current reservation key associated with the host. For all other Reservation Register Action values, this field is reserved.</p> <p>The controller ignores the value of this field when the Ignore Existing Key (IEKEY) bit is set to '1'.</p>
15:08	<p><b>New Reservation Key (NRKEY):</b> If the Reservation Register Action field is cleared to 000b (i.e., Register Reservation Key) or 010b (i.e., Replace Reservation Key), then this field contains the new reservation key associated with the host. For all other Reservation Register Action values, this field is reserved.</p>

### 7.6.1 Command Completion

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

### 7.7 Reservation Release command

The Reservation Release command is used to release or clear a reservation held on a namespace.

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

**Figure 575: Reservation Release – Data Pointer**

Bits	Description
127:00	<p><b>Data Pointer (DPTR):</b> This field specifies the location of a data buffer where data is transferred from. Refer to Figure 92 for the definition of this field.</p>

**Figure 576: Reservation Release – Command Dword 10**

Bits	Description
31:16	Reserved
15:08	<p><b>Reservation Type (RTYPE):</b> If the Reservation Release Action field is cleared to 000b (i.e., Release), then this field specifies the type of reservation that is being released. The reservation type in this field shall match the current reservation type. If the reservation type in this field does not match the current reservation type, then the controller should return a status code of Invalid Field in Command. This field is defined in Figure 571.</p>
07:05	Reserved
04	<p><b>Dispersed Namespace Reservation Support (DISNSRS):</b> This bit specifies host support for reservations on dispersed namespaces for a host that supports dispersed namespaces (i.e., for a host that has set the Host Dispersed Namespace Support (HDISNS) field to 1h in the Host Behavior Support feature).</p> <p>If this bit is set to '1', then the host supports reservations on dispersed namespaces (i.e., the host supports receiving a value of FFFDh in the Controller ID (CNTLID) field of a Registered Controller data structure or Registered Controller Extended data structure (refer to section 8.1.9.6)).</p> <p>If this bit is cleared to '0', then the host does not support reservations on dispersed namespaces (i.e., the host does not support receiving a value of FFFDh in the Controller ID (CNTLID) field of a Registered Controller data structure or Registered Controller Extended data structure (refer to section 8.1.9.6)). If the HDISNS field is set to 1h in the Host Behavior Support feature and the host submits the command to a dispersed namespace with this bit cleared to '0', then the controller aborts the command with a status code of Namespace Is Dispersed as described in section 8.1.9.6.</p>
03	<p><b>Ignore Existing Key (IEKEY):</b> If this bit is set to a '1', then the controller shall return an error of Invalid Field in Command. If this bit is cleared to '0', then the Current Reservation Key is checked.</p>



**Figure 576: Reservation Release – Command Dword 10**

Bits	Description												
02:00	<p><b>Reservation Release Action (RRELA):</b> This field specifies the reservation action that is performed by the command.</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> <th>Reference</th> </tr> </thead> <tbody> <tr> <td>000b</td> <td>Release</td> <td>8.1.22.6</td> </tr> <tr> <td>001b</td> <td>Clear</td> <td>8.1.22.8</td> </tr> <tr> <td>010b to 111b</td> <td>Reserved</td> <td></td> </tr> </tbody> </table>	Value	Definition	Reference	000b	Release	8.1.22.6	001b	Clear	8.1.22.8	010b to 111b	Reserved	
Value	Definition	Reference											
000b	Release	8.1.22.6											
001b	Clear	8.1.22.8											
010b to 111b	Reserved												

**Figure 577: Reservation Release Data Structure**

Bytes	O/M	Description
7:0	M	<b>Current Reservation Key (CRKEY):</b> The field specifies the current reservation key associated with the host.

### 7.7.1 Command Completion

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

### 7.8 Reservation Report command

The Reservation Report command returns a Reservation Status data structure to memory that describes the registration and reservation status of a namespace.

If the namespace is not a dispersed namespace, then the size of the Reservation Status data structure is a function of the number of registrants of the namespace (i.e., there is a Registrant data structure and/or Registrant Extended data structure for each such registrant). If the namespace is a dispersed namespace that is able to be accessed by controllers in multiple participating NVM subsystems, then the size of the Reservation Status data structure is a function of the number of registrants of the namespace in the NVM subsystem containing the controller processing the command and the number of registrants of the namespace in each separate participating NVM subsystem. The controller returns the data structure in Figure 581 if the host has selected a 64-bit Host Identifier and the data structure in Figure 582 if the host has selected a 128-bit Host Identifier (refer to section 5.1.25.1.28).

For controllers compliant with NVM Express Base Specification, Revision 2.0 and earlier, registrants of the namespace that are not associated with any controller in the NVM subsystem may or may not be reported by this command.

If a 64-bit Host Identifier has been specified and the Extended Data Structure bit is set to ‘1’ in Command Dword 11, then the controller shall abort the command with the status code of Host Identifier Inconsistent Format. If a 128-bit Host Identifier has been specified and the Extended Data Structure bit is cleared to ‘0’ in Command Dword 11, then the controller shall abort the command with the status code of Host Identifier Inconsistent Format.

The command uses Command Dword 10 and Command Dword 11. If the command uses PRPs for the data transfer, then PRP Entry 1 and PRP Entry 2 fields are used. If the command uses SGLs for the data transfer, then the SGL Entry 1 field is used. All other command specific fields are reserved.

**Figure 578: Reservation Report – Data Pointer**

Bits	Description
127:00	<b>Data Pointer (DPTR):</b> This field specifies the location of a data buffer where data is transferred to. Refer to Figure 92 for the definition of this field.

**Figure 579: Reservation Report – Command Dword 10**

Bits	Description
31:00	<p><b>Number of Dwords (NUMD):</b> This field specifies the number of dwords of the Reservation Status data structure to transfer. This is a 0's based value.</p> <p>If this field corresponds to a length that is less than the size of the Reservation Status data structure, then only that specified portion of the data structure is transferred. If this field corresponds to a length that is greater than the size of the Reservation Status data structure, then the entire contents of the data structure are transferred and no additional data is transferred.</p>

**Figure 580: Reservation Report – Command Dword 11**

Bits	Description
31:02	Reserved
01	<p><b>Dispersed Namespace Reservation Support (DISNSRS):</b> This bit specifies host support for reservations on dispersed namespaces for a host that supports dispersed namespaces (i.e., for a host that has set the Host Dispersed Namespace Support (HDISNS) field to 1h in the Host Behavior Support feature).</p> <p>If this bit is set to '1', then the host supports reservations on dispersed namespaces (i.e., the host supports receiving a value of FFFDh in the Controller ID (CNTLID) field of a Registrant data structure or Registrant Extended data structure (refer to section 8.1.9.6)).</p> <p>If this bit is cleared to '0', then the host does not support reservations on dispersed namespaces (i.e., the host does not support receiving a value of FFFDh in the Controller ID (CNTLID) field of a Registrant data structure or Registrant Extended data structure (refer to section 8.1.9.6)). If the HDISNS field is set to 1h in the Host Behavior Support feature and the host submits the command to a dispersed namespace with this bit cleared to '0', then the controller aborts the command with a status code of Namespace Is Dispersed as described in section 8.1.9.6.</p>
00	<p><b>Extended Data Structure (EDS):</b> If this bit is set to '1', then the controller returns the extended data structure defined in Figure 582. If this bit is cleared to '0', then the controller returns the data structure defined in Figure 581.</p>

**Figure 581: Reservation Status Data Structure**

Bytes	Description
<b>Header</b>	
03:00	<p><b>Generation (GEN):</b> This field contains a 32-bit wrapping counter that is incremented any time any one the following occur:</p> <ul style="list-style-type: none"> <li>a Reservation Register command completes successfully on any controller associated with the namespace;</li> <li>a Reservation Release command with Reservation Release Action (RRELA) set to 001b (i.e., Clear) completes successfully on any controller associated with the namespace; and</li> <li>a Reservation Acquire command with Reservation Acquire Action (RACQA) set to 001b (Preempt) or 010b (Preempt and Abort) completes successfully on any controller associated with the namespace.</li> </ul> <p>If the value of this field is FFFFFFFFh, then the field shall be cleared to 0h when incremented (i.e., rolls over to 0h).</p>
04	<p><b>Reservation Type (RTYPE):</b> This field indicates whether a reservation is held on the namespace. A value of 0h indicates that no reservation is held on the namespace. A non-zero value indicates a reservation is held on the namespace and the reservation type is defined in Figure 571.</p>
06:05	<p><b>Number of Registrants (REGSTRNT):</b> This field indicates the number of registrants of the namespace. This indicates the number of Registrant data structures or Registrant Extended data structures contained in this data structure.</p> <p>Note: This field was formerly named Number of Registered Controllers (REGCTL).</p>
08:07	Reserved

**Figure 581: Reservation Status Data Structure**

Bytes	Description						
09	<b>Persist Through Power Loss State (PTPLS):</b> This field indicates the Persist Through Power Loss State associated with the namespace.						
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Reservations are released and registrants are cleared on a power on.</td> </tr> <tr> <td>1</td> <td>Reservations and registrants persist across a power loss.</td> </tr> </tbody> </table>	Value	Definition	0	Reservations are released and registrants are cleared on a power on.	1	Reservations and registrants persist across a power loss.
	Value	Definition					
0	Reservations are released and registrants are cleared on a power on.						
1	Reservations and registrants persist across a power loss.						
23:10	Reserved						
<b>Registered Controller Data Structure List</b>							
47:24	<b>Registrant Data Structure 0</b> Note: This field was formerly named Registered Controller Data Structure 0.						
...	...						
24*n+47: 24*(n+1)	<b>Registrant Data Structure n</b>						

**Figure 582: Reservation Status Extended Data Structure**

Bytes	Description
<b>Header</b>	
23:00	<b>Reservation Status Header (RSHDR):</b> Refer to the Reservation Status Header in Figure 581 for definition.
63:24	Reserved
<b>Registered Controller Extended Data Structure List</b>	
127:64	<b>Registrant Extended Data Structure 0</b> Note: This field was formerly named Registered Controller Extended Data Structure 0.
...	...
64*(n+1)+63: 64*(n+1)	<b>Registrant Extended Data Structure n</b>

**Figure 583: Registered Controller Data Structure**

Bytes	Description						
01:00	<b>Controller ID (CNTLID):</b> If a registrant of the namespace is associated with a controller in the NVM subsystem, then this field contains the controller ID (i.e., the value of the CNTLID field in the Identify Controller data structure) of the controller that is associated with the registrant whose host identifier is indicated in the Host identifier field of this Registrant data structure.						
	If a registrant of the namespace is not associated with any controller in the NVM subsystem, then the controller processing the command shall set this field to FFFFh.						
	If the namespace is a dispersed namespace and the controller is not contained in the same participating NVM subsystem as the controller processing the command, then the Controller ID field is set to FFFDh, as described in section 8.1.9.6.						
02	<b>Reservation Status (RCSTS):</b> This field indicates the reservation status of the registrant associated with this data structure (i.e., the registrant whose host identifier is indicated in the Host Identifier field of this Registrant data structure).						
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td><b>Association with Host Holding Reservation (AHR):</b> If this bit is set to '1', then the registrant associated with this data structure holds a reservation on the namespace. If this bit is cleared to '0', then the controller is not associated with a host that holds a reservation on the namespace.</td> </tr> </tbody> </table>	Bits	Description	7:1	Reserved	0	<b>Association with Host Holding Reservation (AHR):</b> If this bit is set to '1', then the registrant associated with this data structure holds a reservation on the namespace. If this bit is cleared to '0', then the controller is not associated with a host that holds a reservation on the namespace.
	Bits	Description					
7:1	Reserved						
0	<b>Association with Host Holding Reservation (AHR):</b> If this bit is set to '1', then the registrant associated with this data structure holds a reservation on the namespace. If this bit is cleared to '0', then the controller is not associated with a host that holds a reservation on the namespace.						
07:03	Reserved						
15:08	<b>Host Identifier (HOSTID):</b> This field contains the 64-bit Host Identifier of the registrant of the namespace described by this data structure.						

**Figure 583: Registered Controller Data Structure**

Bytes	Description
23:16	<b>Reservation Key (RKEY):</b> This field contains the reservation key of the registrant described by this data structure.

**Figure 584: Registered Controller Extended Data Structure**

Bytes	Description
01:00	<b>Controller ID (CNTLID):</b> Refer to Figure 583 for definition.
02	<b>Reservation Status (RCSTS):</b> Refer to Figure 583 for definition.
07:03	Reserved
15:08	<b>Reservation Key (RKEY):</b> Refer to Figure 583 for definition.
31:16	<b>Host Identifier (HOSTID):</b> This field contains the 128-bit Host Identifier of the registrant of the namespace described by this data structure.
63:32	Reserved

### 7.8.1 Command Completion

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

## 8 Extended Capabilities

This section describes extended capabilities that are optional. Section 8.1 describes extended capabilities that are common to all transport models. Section 8.2 describes extended capabilities that are specific to the Memory-based transport model. Section 8.3 describes extended capabilities that are specific to the Message-based transport model.

### 8.1 Common Extended Capabilities

This section describes extended capabilities that are common to all transport models.

#### 8.1.1 Asymmetric Namespace Access Reporting

##### 8.1.1.1 Asymmetric Namespace Access Reporting Overview

Asymmetric Namespace Access (ANA) occurs in environments where namespace access characteristics (e.g., performance or ability to access the media) may vary based on:

- the controller used to access the namespace (e.g., Fabrics); and
- the internal configuration of the NVM subsystem. Asymmetric Namespace Access Reporting is used to indicate to the host information about those access characteristics.

Shared namespaces may be accessed through controllers via multiple PCIe ports or fabric ports (refer to section 2.4.1). The controllers that provide access to a shared namespace may provide identical access characteristics through all controllers (i.e., symmetric access), or may provide different access characteristics through some controllers (i.e., asymmetric access).

Private namespaces are accessed by only one controller at a time. The access characteristics of the namespace through that controller may be impacted as a result of changes to the internal configuration of the NVM subsystem. If the access characteristics of the namespace through that controller are impacted by the internal configuration of the NVM subsystem, then asymmetric access occurs.

Symmetric access to a namespace occurs when:

- the access characteristics using one controller are identical to the access characteristics when using a different controller; and
- changes to the internal configuration of the NVM subsystem do not impact the access characteristics.

Asymmetric access to a namespace occurs when:

- the access characteristics using one controller may differ from the access characteristics when using a different controller; or
- changes to the internal configuration of the NVM subsystem may impact the access characteristics.

While commands may be sent to a shared namespace through any attached controller with asymmetric access, the characteristics (e.g., performance or ability to access the media) may differ based on which controller is used; as a result, the host should consider those characteristics when selecting which controller to use for each command that accesses the namespace. The NVM subsystem may perform autonomous internal reconfiguration that results in a change to the access characteristics.

If an NVM subsystem supports Asymmetric Namespace Access Reporting, then all controllers in that NVM subsystem shall:

- set the Asymmetric Namespace Access Reporting Support (ANARS) bit to '1' in the Controller Multi-path I/O and Namespace Sharing Capabilities (CMIC) field in the Identify Controller data structure (refer to Figure 312) to indicate support for Asymmetric Namespace Access Reporting;
- set the Report ANA Optimized State (RANAOS) bit to '1' in the Asymmetric Namespace Access Capabilities (ANACAP) field in the Identify Controller data structure to indicate that the ANA Optimized state is able to be reported;

- set the Report ANA Non-Optimized State (RANANOS) bit to '1' in the ANACAP field in the Identify Controller data structure if ANA Non-Optimized state is able to be reported;
- set the Report ANA Inaccessible State (RANAIS) bit to '1' in the ANACAP field in the Identify Controller data structure if ANA Inaccessible state is able to be reported;
- set the Report ANA Persistent Loss State (RANAPLS) bit to '1' in the ANACAP field in the Identify Controller data structure if ANA Persistent Loss state is able to be reported;
- set the Report ANA Change State (RANACS) bit to '1' in the ANACAP field in the Identify Controller data structure if ANA Change state is able to be reported;
- support Asymmetric Namespace Access Change Notices (refer to section 5.1.25.1.5); and
- support the Asymmetric Namespace Access log page (refer to section 5.1.12.1.13).

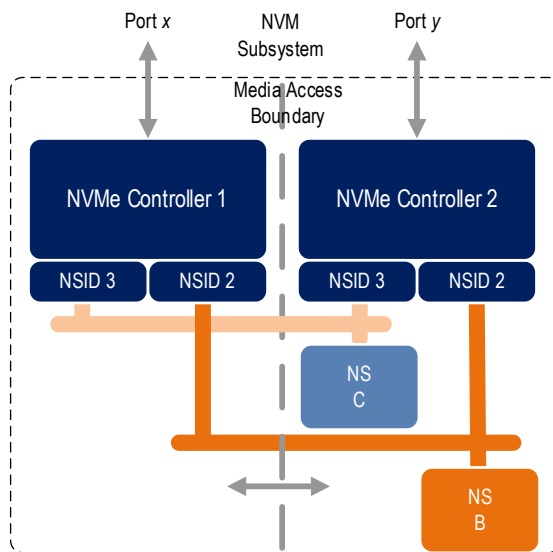
Namespaces attached to a controller that supports Asymmetric Namespace Access Reporting shall:

- be members of an ANA Group; and
- supply a valid ANA Group Identifier in the ANA Group Identifier (ANAGRPID) field in the Identify Namespace data structure (refer to the applicable I/O Command Set specification).

A controller that supports Asymmetric Namespace Access Reporting may also support multiple domains (refer to section 3.2.5).

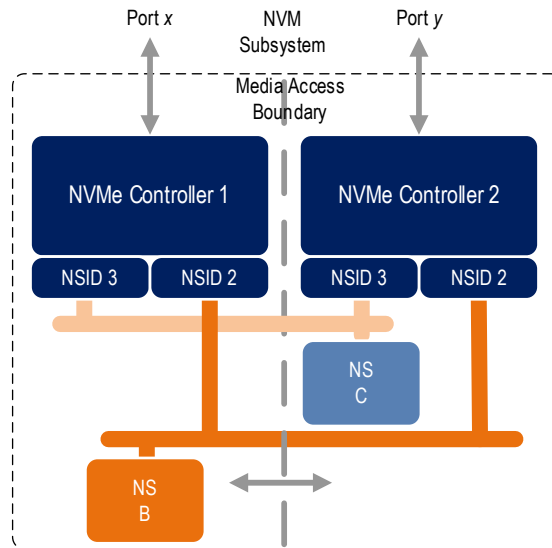
Figure 585 shows an example of an NVM subsystem where access characteristics vary as a result of the presence of two independent domains. In this example, the non-volatile storage media for namespace B and for namespace C are contained within the same domain that contains controller 2. As a result, controller 2 provides optimized access to namespace B and to namespace C while controller 1 does not provide optimized access to namespace B or to namespace C. In an NVM subsystem that supports multiple domains (refer to section 3.2.5), the Media Access Boundary shown in Figure 585 may be a Communication boundary as shown in Figure 71 and Figure 72.

**Figure 585: Namespace B and C optimized through Controller 2**



To provide optimized access to namespace B through controller 1, the NVM subsystem may be administratively reconfigured, or may perform autonomous internal reconfiguration actions that change the access characteristics of namespace B when accessed through controller 1 and controller 2 as shown in Figure 586. Controller 2 provides optimized access to namespace C while controller 1 provides optimized access to namespace B. In an NVM subsystem that supports multiple domains (refer to section 3.2.5), the Media Access Boundary shown in Figure 586 may be a Communication boundary as shown in Figure 71 and Figure 72.

**Figure 586: Namespace B optimized through Controller 1**



### 8.1.1.2 ANA Groups

Namespaces that are members of the same ANA Group perform identical asymmetric namespace access state transitions. The ANA Group maintains the same asymmetric namespace access state for all namespaces that are members of that ANA Group (i.e., a change in the asymmetric namespace access state of one namespace only occurs as part of a change in the asymmetric namespace access state of all namespaces that are members of that ANA Group). Namespaces that are members of the same ANA Group shall be members of the same domain (refer to section 2.3.1). The method for assigning namespaces to ANA Groups is outside the scope of this specification.

An ANA Group may contain zero or more namespaces, zero or more NVM Sets, or zero or more Endurance Groups. The mapping of namespaces, NVM Sets, and Endurance Groups to ANA Groups is vendor specific.

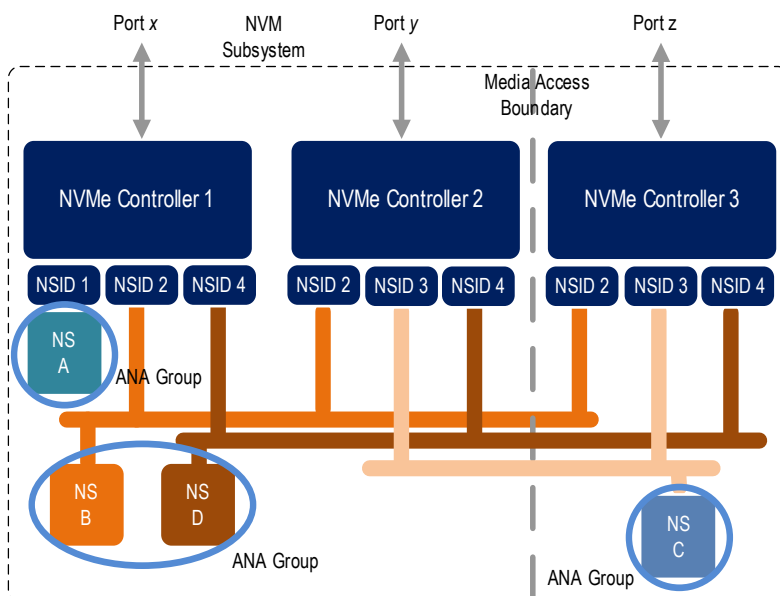
A valid ANA Group Identifier is a non-zero value that is less than or equal to ANAGRPMAX (refer to Figure 312).

The ANA Group Identifier (ANAGRPID) for each ANA Group shall be unique within the NVM subsystem. If the ANA Group ID Change Support (ANAGIDCS) bit in the ANACAP field in the Identify Controller data structure is set to '1', then the ANA Group Identifier shall not change while the namespace is attached to any controller in the NVM subsystem. If the ANAGIDCS bit is cleared to '0', then the ANA Group Identifier may change while the namespace is attached to any controller in the NVM subsystem. If the ANA Group Identifier changes, the controller shall issue the Asymmetric Namespace Access Change Notice as described in 8.1.1.9.

If two or more participating NVM subsystems provide access to a dispersed namespace, and that dispersed namespace is a member of an ANA Group on any participating NVM subsystem, then each participating NVM subsystem shall contain an ANA Group whose members include that dispersed namespace. The ANAGRPID for a dispersed namespace may or may not be the same on each participating NVM subsystem.

Figure 587 shows the following four namespaces:

- the private namespace A in a first ANA Group;
- namespace B and namespace D, that are in the same second ANA Group; and
- namespace C that is in a third ANA Group.

**Figure 587: Multiple Namespace groups**

### 8.1.1.3 Asymmetric Namespace Access states

The Asymmetric Namespace Access State indicates information about the characteristics of the relationship between a controller and an ANA Group. The following asymmetric namespace access states are defined:

- ANA Optimized (refer to section 8.1.1.4);
- ANA Non-Optimized (refer to section 8.1.1.5);
- ANA Inaccessible (refer to section 8.1.1.6);
- ANA Persistent Loss (refer to section 8.1.1.7); and
- ANA Change (refer to section 8.1.1.8).

### 8.1.1.4 ANA Optimized state

While the relationship between the controller and an ANA group is in this state, the characteristic of that relationship to each namespace in that group is optimized. Commands processed by a controller that reports this state for an ANA Group provide optimized access characteristics to any namespace in that ANA Group. A controller that supports ANA Reporting shall support reporting this state.

While in this state, all commands, functions, and operations supported by the namespace shall perform as described in this specification.

### 8.1.1.5 ANA Non-Optimized state

While the relationship between the controller and an ANA group is in this state, the characteristic of that relationship to each namespace in that group is non-optimized. Commands processed by a controller that reports this state for an ANA Group provide non-optimized access characteristics (e.g., the processing of some commands, especially those involving data transfer, may operate with lower performance or may use NVM subsystem resources less effectively than if a controller is used that reports the optimized state) to any namespace in that ANA Group. Support for reporting this state is optional.

While in this state, all commands, functions, and operations supported by the namespace shall perform as described in this specification.



#### **8.1.1.6 ANA Inaccessible state**

While the relationship between the controller and an ANA group is in this state, the characteristic of that relationship to each namespace in that group is inaccessible. Commands processed by a controller that reports this state for an ANA Group are not able to access user data of namespaces in that ANA Group. The namespaces may become accessible through the controller reporting this state at a future time (i.e., a subsequent ANA state transition may occur). Support for reporting this state is optional.

While in this state, accurate namespace related capacity information may not be available. As a result, some namespace capacity information returned in the Identify Namespace data structure (e.g., the NUSE field and the NVMCAP field), are cleared to 0h. For that namespace capacity information, hosts should use the Identify Namespace data structure returned from a controller that reports the relationship between the controller and the namespace to be in the ANA Optimized state or in the ANA Non-Optimized state.

A controller shall abort commands, other than those described in section 8.1.1.10, with a status code of Asymmetric Access Inaccessible if those commands are submitted while the relationship between the specified namespace and the controller processing the command is in this state.

While ANA Inaccessible state is reported by a controller for the namespace, the host should retry the command on a different controller that is reporting ANA Optimized state or ANA Non-Optimized state. If no controllers are reporting ANA Optimized state or ANA Non-Optimized state, then a transition may be occurring such that a controller reporting the Inaccessible state may become accessible and the host should retry the command on the controller reporting Inaccessible state for at least ANATT seconds (refer to Figure 312). Refer to section 8.1.2.2.

#### **8.1.1.7 ANA Persistent Loss state**

While the relationship between the controller and an ANA group is in this state, the characteristic of that relationship to each namespace in that group is persistently inaccessible. Commands processed by a controller that reports this state for an ANA Group are persistently not able to access user data of namespaces in that ANA Group. The relationship between a controller and an ANA Group in this state shall not transition to any other ANA state. Support for reporting this state is optional.

While in this state, accurate namespace related capacity information may not be available. As a result, some namespace capacity information returned in the Identify Namespace data structure (e.g., the NUSE field and the NVMCAP field), are cleared to 0h. For that namespace capacity information, hosts should use the Identify Namespace data structure returned from a controller that reports the relationship between the controller and the namespace to be in the ANA Optimized state or in the ANA Non-Optimized state.

A controller shall abort commands, other than those described in section 8.1.1.10, with a status code of Asymmetric Access Persistent Loss if those commands are submitted while the relationship between the specified namespace and the controller processing the command is in this state.

While ANA Persistent Loss state is reported by a controller for the namespace, the host should retry the command on a different controller that is reporting ANA Optimized state or ANA Non-Optimized state. If no controllers are reporting ANA Optimized state or ANA Non-Optimized state, then a transition may be occurring such that a controller reporting the Inaccessible state may become accessible and the host should retry the command on the controller reporting Inaccessible state for at least ANATT seconds (refer to Figure 312).

#### **8.1.1.8 ANA Change state**

The change from one asymmetric namespace access state to another asymmetric namespace access state is called a transition. Transitions may occur in such a way that the ANA Change state is not visible to the host (i.e., the ANA Change state may or may not be reported in the Asymmetric Namespace Access State field in the Asymmetric Namespace Access log page (refer to section 5.1.12.1.13)). Support for reporting this state is optional.

A controller shall abort commands, other than those described in 8.1.1.10, with a status code of Asymmetric Access Transition if those commands are submitted while the relationship between the specified namespace and the controller processing the command is in this state.

While ANA Change state is reported by a controller for the namespace, the host should:

- a) after a short delay, retry the command on the same controller for at least ANATT (refer to Figure 312) seconds (e.g., if ANATT is 30, perform 3 retries at 10 s intervals, or 10 retries at 3 s intervals); or
- b) retry the command on a different controller that is reporting ANA Optimized state or ANA Non-Optimized state.

### 8.1.1.9 Asymmetric Namespace Access Change Notifications

If Asymmetric Namespace Access Change Notices are enabled on a controller, then an Asymmetric Namespace Access Change Notice shall be sent as described in section 5.1.25.1.5 by the controllers where the change occurred:

- a) if an ANA Group Identifier (refer to Figure 312) changes;
- b) if an asymmetric namespace access state transition fails (e.g., a transition begins, but does not complete and the controller returns to the state that existed before the transition began); or
- c) upon entry to the following ANA states, unless the state entry is a result of a namespace attachment:
  - ANA Optimized State;
  - ANA Non-Optimized State;
  - ANA Inaccessible State; and
  - ANA Persistent Loss State.

### 8.1.1.10 Asymmetric Namespace Access States Command Processing Effects

Processing of Admin commands that:

- are not NVM Command Set specific commands; and
- do not use the Namespace Identifier (i.e., Figure 141 – “Namespace Identifier Used” column indicates “No”),

are not affected by ANA states, except as specified in Figure 588.

Figure 588 describes Asymmetric Namespace Access effects on command processing.

**Figure 588: ANA effects on Command Processing**

Command	ANA State	Effects on command processing
Get Features	ANA Inaccessible, ANA Persistent Loss, or ANA Change	The following feature identifiers are not available <sup>1</sup> : <ul style="list-style-type: none"> <li>a) Reservation Notification Mask (i.e., 82h);</li> <li>b) Reservation Persistence (i.e., 83h); and</li> <li>c) I/O Command Set specific feature identifiers<sup>2</sup>.</li> </ul>
Get Log Page	ANA Inaccessible, ANA Persistent Loss, or ANA Change	The following log pages are affected: <ul style="list-style-type: none"> <li>a) Error Information (i.e., 01h): The log page is not required to contain entries for namespaces whose relationship to the controller processing the command is in the:                             <ul style="list-style-type: none"> <li>a. ANA Inaccessible state (refer to section 8.1.1.6);</li> <li>b. the ANA Persistent Loss state (refer to section 8.1.1.7); or</li> <li>c. the ANA Change state (refer to section 8.1.1.8).</li> </ul> </li> </ul> The following log pages are not available <sup>1</sup> : <ul style="list-style-type: none"> <li>a) Media Unit Status log page (refer to section 5.1.12.1.16); and</li> <li>b) Supported Capacity Configuration List log page (refer to section 5.1.12.1.17).</li> </ul>

**Figure 588: ANA effects on Command Processing**

Command	ANA State	Effects on command processing
Identify	ANA Inaccessible or ANA Persistent Loss	Capacity fields in the Identify Namespace data structure (refer to the applicable I/O Command Set specification) information are cleared to 0h.
Set Features	ANA Inaccessible	The saving of features shall not be supported and the following feature identifiers are not available <sup>1</sup> : a) Reservation Notification Mask (i.e., 82h); b) Reservation Persistence (i.e., 83h); and I/O Command Set specific feature identifiers <sup>2</sup> . If the NSID is set to FFFFFFFFh, then the command shall abort <sup>3</sup> with a status code of Asymmetric Access Inaccessible (refer to section 8.1.1.6).
	ANA Change	The saving of features shall not be supported and the following feature identifiers are not available <sup>1</sup> : a) Reservation Notification Mask (i.e., 82h); b) Reservation Persistence (i.e., 83h); and c) I/O Command Set specific feature identifiers <sup>2</sup> . If the NSID is set to FFFFFFFFh, then the command shall abort <sup>3</sup> with a status code of Asymmetric Access Transition (refer to section 8.1.1.8).
	ANA Persistent Loss	The command shall abort with a status code of Asymmetric Access Persistent Loss (refer to section 8.1.1.7).
Notes: 1. If the ANA state is ANA Inaccessible State, then commands that use feature identifiers or log pages that are not available shall abort with a status code of Asymmetric Access Inaccessible. If the ANA state is ANA Persistent Loss State, then commands that use feature identifiers or log pages that are not available shall abort with a status code of Asymmetric Access Persistent Loss. If the ANA state is ANA Change State, then commands that use feature identifiers or log pages that are not available shall abort with a status code of Asymmetric Access Transition. 2. I/O Command Set specific definition. Refer to each I/O Command Set specification for applicability and additional details, if any. 3. If any namespace that is attached to the controller is in an ANA Group that is in the ANA Inaccessible state, the ANA Persistent Loss state, or the ANA Change state, then the command shall abort with the indicated status. Depending on the ANA state of the ANA Group that contains a namespace (e.g., an ANA state changes during the processing of the command), the specified feature identifier may be altered for some attached namespaces and not altered for other attached namespaces.		

## 8.1.2 Asymmetric Namespace Access Reporting – Host Considerations (Informative)

### 8.1.2.1 Host ANA Normal Operation

The host determines if ANA is supported by examining the Asymmetric Namespace Access Reporting Support (ANARS) bit in the CMIC field in the Identify Controller data structure (refer to Figure 312). The NSID or identifier (refer to section 4.5) is used to determine when multiple paths to the same namespace that is not a dispersed namespace are available. For dispersed namespaces, globally unique namespace identifiers (refer to section 8.1.9.2) are used to determine when multiple paths are available to the same namespace, as described in section 8.1.9.5. The host examines the ANA log page (refer to section 5.1.12.1.13) for each controller to determine the ANA state of each group of namespaces attached to that controller.

To send a command to a namespace, the host should select a controller that reports the ANA Optimized State (refer to section 8.1.1.4) and send the command to that controller. If more than one controller that

reports the ANA Optimized state for a namespace are found, then the host may use all of those controllers to send commands.

If there are no controllers that report the ANA Optimized state for a namespace, then the host should select a controller that reports ANA Non-Optimized State (refer to section 8.1.1.5) for that namespace and send the command to that controller. If more than one controller that reports ANA Non-Optimized state for a namespace are found, then the host may use all of those controllers to send commands.

If multiple controllers are being used, then the algorithm for determining which controller to use next is outside the scope of this specification (e.g., the host may select a simple round robin algorithm, a queue depth weighted algorithm, a transfer length weighted algorithm, or any other algorithm).

If there are no controllers that report the ANA Optimized state for a namespace and there are no controllers that report the ANA Non-Optimized state for that namespace, then the host should examine controllers that report the ANA Inaccessible state as described in section 8.1.2.2.

#### **8.1.2.2 Host ANA Inaccessible Operation**

If the ANA log page reports an ANA state of ANA Inaccessible State for an ANA Group or a command returns a status code of Asymmetric Access Inaccessible, then the host should:

- not use that controller to send commands to any namespace in that ANA Group; and
- select a different controller for sending commands to all namespaces in that ANA Group.

If there are no controllers that report the ANA Optimized state for a namespace and there are no controllers that report the ANA Non-Optimized state, then a transition may be occurring that also impacts controllers that are reporting the ANA Inaccessible state. As a result, the host should use the methods described for Host ANA Transition operation (refer to section 8.1.2.5) to determine if the controller reporting ANA Inaccessible state transitions during the ANATT time interval to an ANA state that enables commands to be processed by that controller.

#### **8.1.2.3 Host ANA Persistent Loss Operation**

If the ANA log page reports an ANA state of ANA Persistent Loss State for an ANA Group or a command returns a status code of Asymmetric Access Persistent Loss, then the host should not use that controller to send commands to any namespace in that ANA Group, and select a different controller for sending commands to any namespace in that ANA Group. If the controller supports the Namespace Management capability (refer to section 8.1.15), then the namespaces in an ANA Group reporting this state should be detached.

#### **8.1.2.4 Host ANA Change Notice Operation**

If the ANA log page reports an ANA state of ANA Change State for an ANA Group or a command returns a status code of Asymmetric Access Transition, then the host should temporarily not use that controller to send commands to any namespace in that ANA Group. If only controllers reporting ANA Inaccessible State are available, then the host should follow these procedures to determine which controller to use. To use a controller, the host may:

- a) if Asymmetric Namespace Access Change Notices are enabled (refer to section 5.1.25.1.5) on the controller, wait for an Asymmetric Namespace Access Change Notice from that controller. Upon receipt of that notice, the host should examine the ANA log page to determine the new ANA state and resume sending commands based on the new ANA state. Such notice should occur within the ANATT time (refer to Figure 312); or
- b) delay and retry the command during the ANATT time interval. The host should not immediately retry, but rather, divide the ANATT time into equal intervals for command retry purposes (e.g., if ANATT is 30, perform 3 retries at 10 s intervals, or 10 retries at 3 s intervals). During or upon completion of the ANATT time interval, the new ANA state of the ANA Group should be known (e.g., one of the command retries returned a different status that indicates completion of the transition to a new ANA state). If the retried command did not complete without error, the ANA log

page should be examined on each controller that provides access to the namespace and the host should resume sending commands based on the new ANA state.

If the ANATT time interval expires, then the host should use a different controller for sending commands to the namespaces in that ANA Group. The ANATT interval reported by the controller should prevent this type of timer expiration from occurring.

#### **8.1.2.5 Host ANA Transition Operation**

Receipt of an Asymmetric Namespace Access Change Notice from a controller may indicate:

- a) that the ANA state reported in one or more ANA Group Descriptors has changed;
- b) a new NSID has been added to one or more of the ANA Group Descriptors;
- c) an NSID has been removed from one or more of the ANA Group Descriptors; and/or
- d) the NSID of a namespace has moved from one ANA Group Descriptor to a different ANA Group Descriptor (i.e., the ANAGRPID field in the Identify Namespace data structure for that namespace has changed), if the ANA Group ID Change Support (ANAGIDCS) bit in the ANACAP field is cleared to '0' in the Identify Controller data structure (refer to Figure 312).

As a result of receiving an Asymmetric Namespace Access Change Notice, the host should read the ANA log page (refer to section 5.1.12.1.13) to check for each of those possible changes.

#### **8.1.2.6 All Paths Down Condition**

An all paths down condition occurs when there are no paths available on the host to access the namespaces in an ANA Group (i.e., the NVM media). To determine whether an all paths down condition has occurred, the host may examine the ANA log page on each controller that provides access to the namespaces in a particular ANA Group. All paths that are not in the ANA Persistent Loss state should be checked. If no paths to the namespaces in that ANA Group become available (i.e., transition to the ANA Optimized state or the ANA Non-Optimized state) for the duration of an ANATT time interval, then an all paths down condition has occurred for the namespaces in that ANA Group.

### **8.1.3 Boot Partitions**

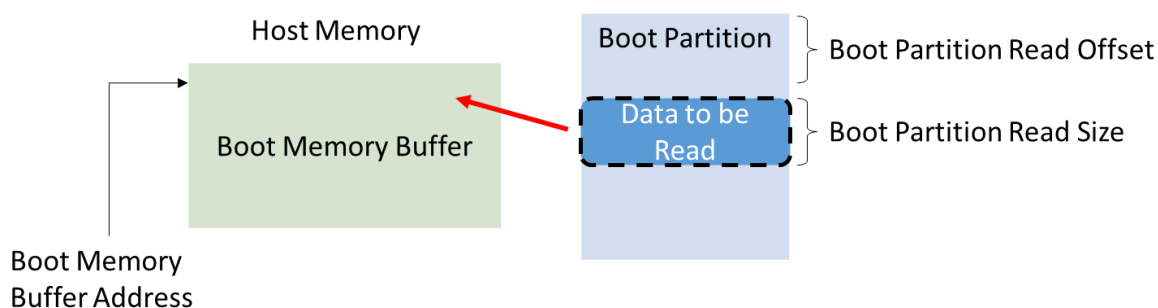
Boot Partitions provide an optional area of NVM storage that may be read by the host without the host initializing queues or enabling the controller. The simplified interface to access Boot Partitions may be used for platform initialization code (e.g., a bootloader that is executed from host ROM) to boot to a pre-OS environment (e.g., UEFI) instead of storing the image on another non-volatile storage medium (e.g., SPI flash). Refer to section 8.1.3.1 for the procedure to read the contents of a Boot Partition.

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

The contents of Boot Partitions are only modified using the Firmware Image Download and Firmware Commit commands (refer to section 8.1.3.2) and may be secured using either the Boot Partition Write Protection Config feature or the Replay Protected Memory Block to prevent unauthorized modifications (refer to section 8.1.3.3).

#### **8.1.3.1 Reading from a Boot Partition**

A Boot Partition is a continuous block of data as shown in Figure 589, that the host may read via NVMe properties.

**Figure 589: Boot Partition Overview**

To read the contents of a Boot Partition using NVMe properties, the host allocates a Boot Partition Memory Buffer in host memory for the controller to copy contents from a Boot Partition. The host initializes the Boot Partition Memory Buffer Base Address. The host sets the Boot Partition ID, Boot Partition Read Size, and Boot Partition Read Offset to initiate the Boot Partition read operation. The host may continue reading from the Boot Partition until the entire Boot Partition has been read.

A portion of the Boot Partition may be read by the host any time the NVM subsystem is powered (i.e., whether or not CC.EN is set to '1'). The host shall not modify transport specific properties (described in the applicable NVMe Transport binding specification), reset, or shutdown the controller while a Boot Partition read is in progress.

To read data from a Boot Partition, the host follows these steps:

1. Initialize the transport (e.g., PCIe link), if necessary;
2. Determine if Boot Partitions are supported by the controller (CAP.BPS);
3. Determine which Boot Partition is active (BPINFO.ABPID) and the size of the Boot Partition (BPINFO.BPSZ);
4. Allocate a physically contiguous memory buffer in the host to store the contents of a Boot Partition;
5. Initialize the address (BPMBL.BMBBA) into the memory buffer where the contents should be copied;
6. Initiate the transfer of data from a Boot Partition by writing to the Boot Partition Read Select (BPRSEL) property. This includes setting the Boot Partition identifier (BPRSEL.BPID), size of Boot Partition Read Size (BPRSEL.BPRSZ) and Boot Partition Read Offset (BPRSEL.BPROF). The controller sets the Boot Read Status (BPINFO.BRS) field while transferring the Boot Partition contents to indicate a Boot Partition read operation is in progress; and
7. Wait for the controller to completely transfer the requested portion of the Boot Partition, indicated in the status field (BPINFO.BRS). If BPINFO.BRS is set to 10b, the requested Boot Partition data has been transferred to the Boot Partition Memory Buffer. If BPINFO.BRS is set to 11b, there was an error transferring the requested Boot Partition data and the host may request the Boot Partition data again.

In constrained memory environments, the host may read the contents of a Boot Partition with a small Boot Partition Memory Buffer by reading a small portion of a Boot Partition, moving the data out of the Boot Memory Buffer to another memory location, and then reading another portion of the Boot Partition until the entire Boot Partition has been read.

If the Boot Partition log page is supported (refer to section 5.1.12.1.1), then the Boot Partition can be accessed through the Boot Partition log page (refer to section 5.1.12.1.21).

### 8.1.3.2 Writing to a Boot Partition

Boot Partition contents may be modified by the host using the Firmware Image Download and Firmware Commit commands while the controller is enabled (CC.EN set to '1').

The process for updating a Boot Partition is:

1. The host issues a Firmware Image Download command to download the contents of the Boot Partition to a controller. There may be multiple portions of the Boot Partition to download, thus the offset for each portion of the Boot Partition being downloaded is specified in the Firmware Image Download command. A host shall send the Boot Partition image in order starting with the beginning of the Boot Partition;
2. The host transitions the Boot Partition that is to be updated to the Write Unlocked State (refer to section 8.1.3.3);
3. The host submits a Firmware Commit command (refer to section 5.1.8) on that controller with a Commit Action of 110b which specifies that the downloaded image replaces the contents of the Boot Partition specified in the Boot Partition ID field;
4. The controller completes the Firmware Commit command. The following actions are taken in certain error scenarios:
  - a. If the firmware activation was not successful because the Boot Partition could not be written, then the controller reports an error of Boot Partition Write Prohibited;
5. (Optional) The host reads the contents of the Boot Partition to verify they are correct (refer to section 8.1.3.1). A host updates the active Boot Partition ID by issuing a Firmware Commit command with a Commit Action of 111b; and
6. The host transitions the Boot Partition to either the Write Locked State or Write Locked Until Power Cycle State to prevent further modification (refer to section 8.1.3.3).

If an internal error, reset, or power loss condition occurs while committing the downloaded image to a Boot Partition, the contents of the Boot Partition may contain the old contents, new contents, or a mixture of both. Host software should verify the contents of a Boot Partition before marking that Boot Partition active to ensure the active Boot Partition is stable.

Host software should not read the contents of a Boot Partition while writing to the Boot Partition. The controller may return a combination of new and old data if the host attempts to perform a Boot Partition read operation while overwriting the contents.

Host software should not overlap firmware/boot partition image update command sequences (refer to section 1.5.41). During a boot partition image update command sequence, if a Firmware Image Download command or a Firmware Commit command is submitted for another firmware/boot partition image update command sequence, the results of both that command and the in-progress firmware image update are undefined.

Host software should use the same controller or Management Endpoint (refer to the NVM Express Management Interface Specification) for all commands that are part of a boot partition image update command sequence. If the commands for a single firmware/boot partition image update command sequence are submitted to more than one controller and/or Management Endpoint, the controller may abort the Firmware Commit command with Invalid Firmware Image status.

### 8.1.3.3 Boot Partition Write Protection

A controller that supports both Boot Partitions and RPMB shall support at least one of the following Boot Partition write protection mechanisms:

- Set Features Boot Partition Write Protection (refer to section 8.1.3.3.1); or
- RPMB Boot Partition Write Protection (refer to section 8.1.3.3.2).

It is not recommended that a controller support both Set Features Boot Partition Write Protection and RPMB Boot Partition Write Protection.

A controller that supports Boot Partitions and does not support RPMB shall support Set Features Boot Partition Write Protection.

A host is able to determine whether Set Features Boot Partition Write Protection is supported by checking the value of the Set Features Boot Partition Write Protection Support bit of the Boot Partition Capabilities field in the Identify Controller Data Structure (refer to Figure 312).

A host is able to determine whether RPMB Boot Partition Write Protection is supported by checking the RPMB Boot Partition Write Protection Support field of the Boot Partition Capabilities field in the Identify

Controller data structure (refer to Figure 312). Only controllers compliant with NVM Express Base Specification, Revision 2.0 and earlier are allowed to report a value of 00b (i.e., support not specified) in that field. If a controller reports the 00b value, a host is able to determine if RPMB Boot Partition Write Protection is supported by checking whether the controller supports both Boot Partitions (refer to section 3.1.4.1) and RPMB (refer to section 5.1.13.2.1) because a controller compliant with NVM Express Base Specification, Revision 2.0 and earlier is required to support RPMB Boot Partition Write Protection when the controller supports both Boot Partitions and RPMB.

If Set Features Boot Partition Write Protection is supported and either:

- 1) RPMB Boot Partition Write Protection is not supported; or
- 2) RPMB Boot Partition Write Protection is not enabled,

then the Boot Partition write protection states are able to be configured via the Boot Partition Write Protection Config feature. Section 8.1.3.3.1 covers the case when only the Boot Partition Write Protection Config feature is supported.

If RPMB Boot Partition Write Protection is supported and enabled, then the Boot Partition write protection states are able to be configured using RPMB (refer to section 8.1.21). Section 8.1.3.3.2 covers the case where only RPMB Boot Partition Write Protection is supported.

Only one mechanism controls the Boot Partition write protection states at a time. Section 8.1.3.3.3 covers considerations when both Boot Partition write protection mechanisms are supported.

If any Boot Partition is shared across multiple controllers (refer to section 8.1.3), then the write protection state of the Boot Partition shall be enforced by all controllers that share that Boot Partition.

**8.1.3.3.1 Set Features Boot Partition Write Protection**

Figure 590 shows an overview of the Boot Partition write protection states for each Boot Partition when only Set Features Boot Partition Write Protection is supported.

**Figure 590: Set Features Boot Partition Write Protection State Machine Model**

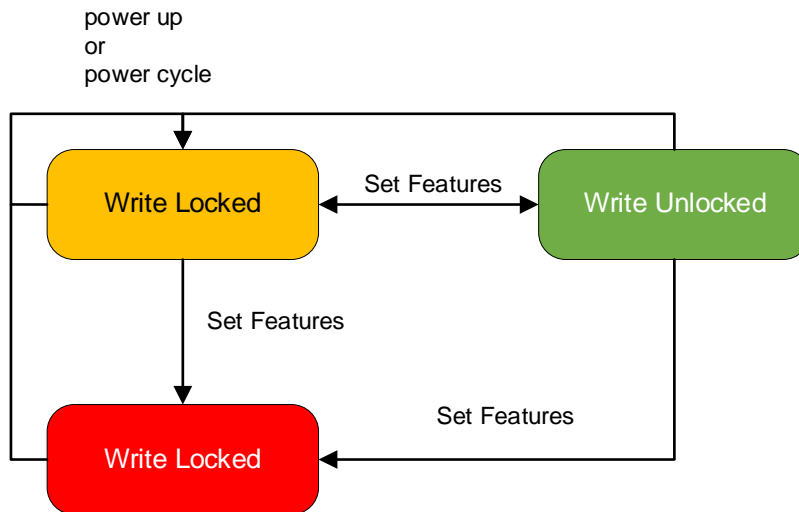


Figure 591 defines the write protection states per Boot Partition if Set Features Boot Partition Write Protection is supported.



**Figure 591: Set Features Boot Partition Write Protection State Definitions**

State	Definition	Persistent Across	
		Power Cycles	Controller Level Resets
Write Unlocked	The Boot Partition is not write locked.	No	Yes
Write Locked	The Boot Partition is write locked.	Yes	Yes
Write Locked Until Power Cycle	The Boot Partition is write locked until the next power cycle.	No	Yes

If Set Features Boot Partition Write Protection is supported, then the default state for all Boot Partitions is the Write Locked state. In this state, a host may read from a Boot Partition but is unable to modify that Boot Partition. To enable modification of a Boot Partition, a host has to first transition the Boot Partition to the Write Unlocked state by setting the appropriate Boot Partition Write Protection State to Write Unlocked using the Boot Partition Write Protection Config feature via the Set Features command.

In the Write Unlocked state, a host may read from and modify a Boot Partition. Any Boot Partition in a Write Unlocked state transitions to the Write Locked state when the controller undergoes a power cycle.

If Set Features Boot Partition Write Protection is supported, then both Boot Partitions support a Write Locked Until Power Cycle state. In this state, the Boot Partition can be read from but is prohibited from being modified. Additionally, once a Boot Partition enters the Write Locked Until Power Cycle state, the Boot Partition remains in this state until the controller is power cycled.

The Write Locked Until Power Cycle state is prohibited in multi-domain NVM subsystems with Boot Partitions shared across controllers (e.g., since clearing that state requires simultaneous power cycle of all controllers that share the Boot Partitions). The result of a command that attempts to use that state in a multi-domain NVM subsystem is specified in section 5.1.25.1.32.

#### **8.1.3.3.2 RPMB Boot Partition Write Protection**

Figure 592 shows an overview of the Boot Partition write protection states for each Boot Partition when only RPMB Boot Partition Write Protection is supported.

**Figure 592: RPMB Boot Partition Write Protection State Machine Model**

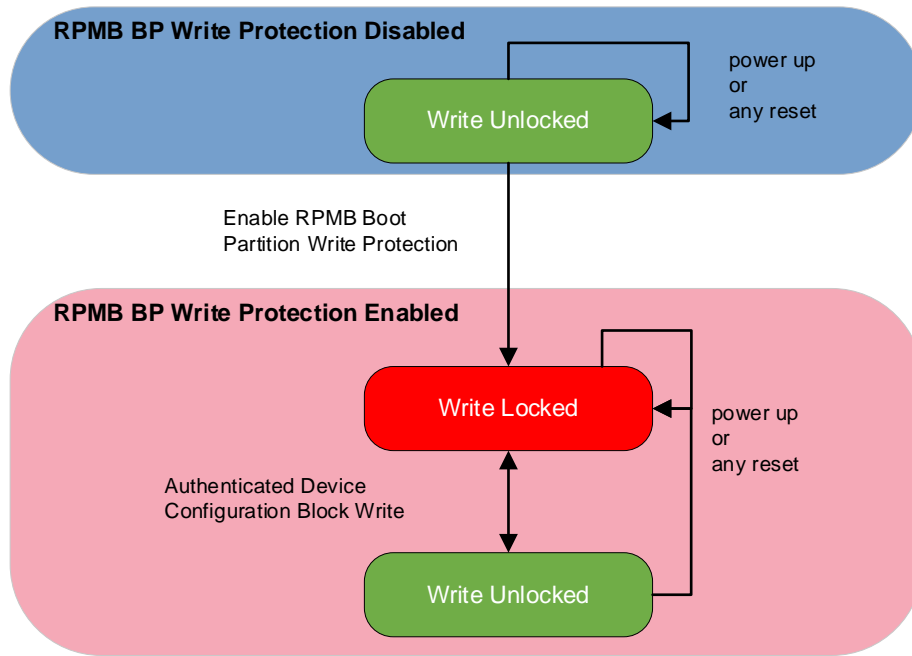


Figure 594 defines the write protection states per Boot Partition if RPMB Boot Partition Write Protection is supported.

**Figure 593: RPMB Boot Partition Write Protection State Definitions**

State	Definition	Persistent Across	
		Power Cycles	Controller Level Resets
<b>RPMB Boot Partition Write Protection Disabled</b>			
Write Unlocked	The Boot Partition is not write locked.	Yes	Yes
<b>RPMB Boot Partition Write Protection Enabled</b>			
Write Unlocked	The Boot Partition is not write locked.	No	No
Write Locked	The Boot Partition is write locked.	Yes	Yes

If Set Features Boot Partition Write Protection is not supported by the controller, then prior to enabling RPMB Boot Partition Write Protection, the default state for all Boot Partitions is the Write Unlocked state. If Set Features Boot Partition Write Protection is also supported by the controller, then the default state for all Boot Partitions is the Write Locked state, regardless of whether RPMB Boot Partition Write Protection has been enabled or not. Refer to section 8.1.3.3.1 for more details on Boot Partition write protection when RPMB Boot Partition Write Protection is disabled and section 8.1.3.3.3 for additional considerations when both Boot Partition write protection capabilities are supported.

If Set Features Boot Partition Write Protection is not supported by the controller, then all Boot Partitions remain unlocked until RPMB Boot Partition Write Protection is enabled by the host. A host enables RPMB Boot Partition Write Protection by setting the Boot Partition Write Protection Enabled bit in the RPMB Device Configuration Block data structure (refer to section 8.1.21). Once RPMB Boot Partition Write Protection is enabled, the controller shall reject Authenticated Device Configuration Block Writes that attempt to disable the RPMB Boot Partition Write Protection mechanism (i.e., enabling RPMB Boot Partition Write Protection

is permanent). Once RPMB Boot Partition Write Protection is enabled, Boot Partitions are able to be modified only after write unlocking the Boot Partition using RPMB.

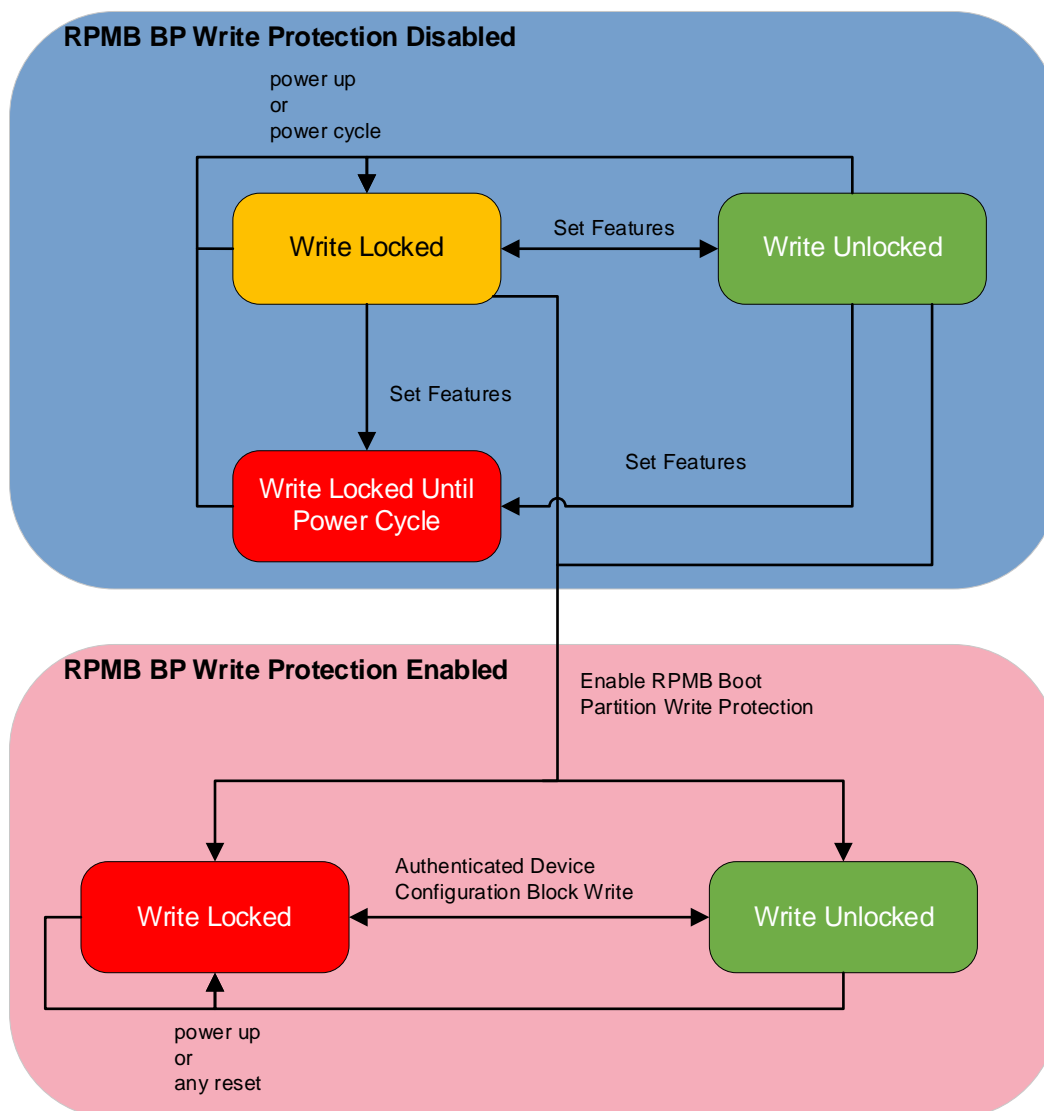
After enabling RPMB Boot Partition Write Protection:

- a) The default state for all Boot Partitions is the Write Locked state. In this state, a host may read a Boot Partition. In this state, the controller rejects attempts to write to a Boot Partition using the Firmware Commit command;
- b) Each Boot Partition may be locked or unlocked independently using the corresponding bit in the Device Configuration Block data structure. A Boot Partition may be unlocked in the same command that enables RPMB Boot Partition Write Protection; and
- c) If any Boot Partition has been unlocked, a power cycle or Controller Level Reset event results in that Boot Partition becoming write locked.

#### **8.1.3.3.3 Interactions between Boot Partition Write Protection Mechanism**

Figure 594 shows an overview of the Boot Partition write protection states for each Boot Partition when both Boot Partition write protection mechanisms are supported. Supporting both Boot Partition write protection mechanisms is discouraged, as specified in section 8.1.3.3.

Figure 594: Boot Partition Write Protection State Machine Model



If both Boot Partition write protection mechanisms are supported by the controller, only one Boot Partition write protection mechanism controls the write protection states of the Boot Partitions for the controller at any time. If RPMB Boot Partition Write Protection is enabled, then RPMB Boot Partition Write Protection controls the Boot Partition write protection states (refer to section 8.1.3.3.2 and section 8.1.21). If RPMB Boot Partition Write Protection is disabled, then Set Features Boot Partition Write Protection controls the Boot Partition write protection states (refer to section 8.1.3.3.1 and section 5.1.25.1.32). Control of the Boot Partition write protection states transitions from Set Features Boot Partition Write Protection to RPMB Boot Partition Write Protection by enabling RPMB Boot Partition Write Protection when the Boot Partitions are in either a Write Unlocked or Write Locked state.

If both Boot Partition write protection capabilities are supported and an RPMB authentication key has not been programmed for RPMB target 0, there is a possibility of malicious bypass of a Boot Partition's Write Locked state. In order to prevent malicious bypass of a Boot Partition's Write Locked state, a controller that supports both Boot Partition write protection mechanisms is required to prevent host attempts to enable RPMB Boot Partition Write Protection when either Boot Partition

is in a Write Locked Until Power Cycle state. Refer to section 8.1.21 for the specific behavior that the controller exhibits under this condition.

## 8.1.4 Capacity Management

### 8.1.4.1 Overview

Capacity Management is a capability for organizing physical media into Endurance Groups and NVM Sets. There are two different forms of Capacity Management, Fixed Capacity Management and Variable Capacity Management. A controller that supports Capacity Management shall support at least one form.

Capacity Management commands shall not be supported by Exported NVM Subsystems.

The host uses Fixed Capacity Management to create Endurance Groups and NVM Sets by selecting a configuration which explicitly allocates Media Units (refer to section 8.1.4.2) to Endurance Groups and NVM Sets.

The host uses Variable Capacity Management to:

- create a single Endurance Group by specifying the desired capacity;
- create a single NVM Set by specifying the desired capacity;
- delete a single Endurance Group; and
- delete a single NVM Set.

The Capacity Adjustment Factor is the ratio between the capacity consumed by an Endurance Group from the Unallocated NVM Capacity field in the Identify Controller data structure and the total NVM capacity in that Endurance Group, i.e.:

$$\text{Capacity Adjustment Factor} = \text{INTEGER} \left( \frac{\text{Capacity Consumed}}{\text{Endurance Group Capacity}} * 100 \right)$$

(E.g., the value 200 indicates that creation of an Endurance Group with a total NVM capacity of 5 GiB consumes 10 GiB of the Unallocated NVM Capacity indicated by the controller).

If an Endurance Group is created, then the controller performs the following actions as an atomic operation:

- a) the value indicated by the Unallocated NVM Capacity (UNVMCAP) field of the Identify Controller data structure is changed based on the requested capacity, the Capacity Adjustment Factor of the created Endurance Group, and the granularity at which the controller allocates NVM capacity; and
- b) the Endurance Group Identifier is added to the Endurance Group List.

If an Endurance Group is deleted, then the controller performs the following actions in sequence:

- 1) the Endurance Group Identifier is removed from the Endurance Group List;
- 2) if the Media Unit Status log page is supported, then the Endurance Group Identifier field is cleared to 0h in all Media Unit Status Descriptors, if any, that indicate the deleted Endurance Group;
- 3) every NVM Set in the Endurance Group is deleted; and
- 4) the value indicated by the Unallocated NVM Capacity (UNVMCAP) field of the Identify Controller data structure is increased by the value that was indicated by the Total Endurance Group Capacity (TEGCAP) field of the Endurance Group Information log page of the deleted Endurance Group.

If any of the entities modified by the above sequence are accessed after the sequence begins and before it completes, then the results are indeterminate.

If an NVM Set is created, then the controller performs the following actions as an atomic operation:

- a) the NVM Set Identifier is added to the NVM Set List; and
- b) the Unallocated Endurance Group Capacity indicated by the Endurance Group Information log page (refer to Figure 218) is decreased by the amount of capacity allocated to the NVM Set; the controller may allocate NVM capacity in units such that the requested size for an NVM Set may be rounded up to the next unit boundary.

If an NVM Set is deleted, then the following actions are performed in sequence:

- 1) the NVM Set Identifier is removed from the NVM Set List;
  - 2) if the Media Unit Status log page is supported, then the NVM Set Identifier field is cleared to 0h in all Media Unit Status Descriptors, if any, that indicated the deleted NVM Set;
  - 3) for each namespace in the deleted NVM Set:
    - a. all commands targeting the namespace are handled as described for namespace deletion in section 8.1.15;
    - b. the namespace identifier is removed from the Allocated Namespace ID list;
    - c. the namespace is deleted;
    - d. for each controller to which the namespace was attached when that namespace was deleted:
      - i. the namespace identifier is added to the Changed Attached Namespace List log page for that controller; and
      - ii. an Attached Namespace Attribute Changed asynchronous event is generated by that controller as described in section 8.1.15.2;
    - e. the namespace identifier is added to the Changed Allocated Namespace List log page for each controller in the NVM subsystem; and
    - f. an Allocated Namespace Attribute Changed asynchronous event is generated for each controller that has Allocated Namespace Attribute Notices enabled as described in section 8.1.15.2;
- and
- 4) the Unallocated Endurance Group Capacity indicated by the Endurance Group Information log page is increased by the amount of capacity formerly allocated to the NVM Set.

If any of the entities modified by the above sequence are accessed after the sequence begins and before it completes, then the results are indeterminate.

If an NVM Set is created or deleted, the value indicated by the Unallocated NVM Capacity (UNVMCAP) field of the Identify Controller data structure is not changed.

#### 8.1.4.2 Fixed Capacity Management

A Media Unit represents a component of the underlying media in a domain. An implementation may choose to represent each die as a separate Media Unit; however, this is not required. A Media Unit is the smallest media component for which the controller reports wear information (refer to the Available Spare field and the Percentage Used field in the Media Unit Status Descriptor, Figure 255).

Two or more I/O operations to a Media Unit at the same time may interfere with each other as they contend for resources internal to or shared by that Media Unit.

A controller supporting Fixed Capacity Management:

- a) shall support the Media Unit Status log page (refer to section 5.1.12.1.16);
- b) shall support Endurance Groups (refer to section 3.2.3);
- c) may support NVM Sets (refer to section 3.2.2);
- d) shall set the Fixed Capacity Management bit to '1' in the CTRATT field of the Identify Controller data structure (refer to Figure 312);
- e) shall support the Supported Capacity Configuration List log page (refer to section 5.1.12.1.17); and
- f) shall support the Select Capacity Configuration operation of the Capacity Management command (refer to section 5.1.3).

Media Units are accessed by way of Channels that represent communication pathways that may be a source of contention. This information is reported to facilitate the composition of NVM Sets that minimize interference between independent writers competing for this type of resource.

The host allocates the Media Units in a domain to Endurance Groups and NVM Sets by:

- 1) reading the Supported Capacity Configuration List log page (refer to Figure 256) and selecting the desired configuration; and
- 2) issuing a Capacity Management command specifying the Select Capacity Configuration operation and the Capacity Configuration Identifier of the desired configuration.

Following successful completion of the command, each Media Unit is allocated to one Endurance Group and to one NVM Set. The resulting configuration of Media Units is reported by the Media Unit Status log page (refer to section 5.1.12.1.16).

#### 8.1.4.3 Variable Capacity Management

Variable Capacity Management allows the dynamic creation and deletion of Endurance Groups and NVM Sets.

A controller supporting Variable Capacity Management:

- a) may support the Media Unit Status log page;
- b) shall support Endurance Groups;
- c) may support NVM Sets;
- d) shall set the Variable Capacity Management bit to '1' in the CTRATT field of the Identify Controller data structure (refer to Figure 312);
- e) shall support the Create Endurance Group operation of the Capacity Management command;
- f) may support the Delete Endurance Group operation of the Capacity Management command; and
- g) if NVM Sets are supported:
  - a. shall support the Create NVM Set operation of the Capacity Management command;
  - b. shall report non-zero values in the Total Endurance Group Capacity field and the Unallocated Endurance Group Capacity field in the Endurance Group Information log page (refer to Figure 218); and
  - c. may support the Delete NVM Set operation of the Capacity Management command.

If a controller supports the Delete Endurance Group operation of the Capacity Management command, then it shall set the Delete Endurance Group bit to '1' in the CTRATT field of the Identify Controller data structure.

If a controller supports the Delete NVM Set operation of the Capacity Management command, then it shall set the Delete NVM Set bit to '1' in the CTRATT field of the Identify Controller data structure.

A typical sequence of operations for allocating capacity is:

- 1) determine the available capacities in each domain (refer to section 3.2.5);
- 2) create Endurance Group with desired capacity (refer to section 5.1.3);
- 3) create NVM Set with desired capacity in the Endurance Group (refer to section 5.1.3); and
- 4) create namespace with desired capacity in the NVM Set (refer to section 5.1.21).

A typical sequence of operations for deallocating capacity is:

- 1) delete namespace that contains formatted storage, if any, (refer to section 5.1.21);
- 2) delete NVM Set, if any, (refer to section 5.1.30); and
- 3) delete Endurance Group (refer to section 5.1.3).

If there is insufficient unallocated capacity in an Endurance Group for the controller to create an NVM Set, then the host can delete one or more NVM Sets in that Endurance Group and create the new NVM Set using some or all of the available capacity.

If there is insufficient unallocated capacity in a domain for the controller to create an Endurance Group, then the host can delete one or more Endurance Groups in that domain and create a new Endurance Group using some or all of the available capacity.

### 8.1.5 Command and Feature Lockdown

The Command and Feature Lockdown capability is used to prohibit the execution of commands submitted to NVM Express controllers and/or Management Endpoints in an NVM subsystem. Within this feature, commands and Feature Identifiers are defined with the following scopes:

- Admin Command Set commands defined by the Opcode field;
- Set Features command features defined by the Feature Identifier field;
- Management Interface Command Set commands defined by the Opcode field (refer to the NVM Express Management Interface Specification); and
- PCIe Command Set commands defined by the Opcode field (refer to the NVM Express Management Interface Specification).

Admin Command Set commands and Feature Identifiers are defined to be prohibitable by this feature, however it is vendor specific which of the Command Set commands and Feature Identifiers are prohibitable from execution, including the Lockdown command itself.

The prohibition of commands or Feature Identifiers on an interface is specified in the Interface field of the Lockdown command (refer to section 5.1.15). The prohibition applies to all applicable:

- NVM Express controllers; and
- Management Endpoints,

in the NVM subsystem, as specified in the Interface field.

The Lockdown command is used to specify commands that are prohibited from execution (i.e., locked down) and may be used further to then again allow that command to be executed.

The prohibiting of execution of a command as part of this feature shall persist until:

- a) power cycle of the NVM subsystem; or
- b) further being allowed by a follow-on Lockdown command.

If a prohibited Admin Command Set command or Feature Identifier is processed by a controller in the NVM subsystem, then that command shall be aborted with a status code of Command Prohibited by Command and Feature Lockdown.

If a prohibited Management Interface Command Set command or PCIe Command Set command is processed by a management endpoint in the NVM subsystem, then that command shall be aborted and send a Response Message with an Access Denied Error Response (refer to the NVM Express Management Interface Specification).

The prohibition or allowance of the execution of a command is based on the interface on which the command is received (i.e., the Admin Submission Queue, or an out-of-band Management Endpoint). For example, a command is able to be prohibited if received on an Admin Submission Queue but allowed if received on an out-of-band Management Endpoint, if supported. The Interface field in the Lockdown command (refer to 5.1.15) is used to specify this behavior.

A host may use the Command and Feature Lockdown log page (refer to section 5.1.12.1.20) to determine the commands and Feature Identifiers that are allowed to be prohibited from execution. A Get Log Page command specifying the Command and Feature Lockdown log page returns a list of Admin command opcodes or Feature Identifiers depending on the scope specified in the Get Log Page command. The returned list of opcodes or Feature Identifiers are the opcodes or Feature Identifiers that are:

- a) supported as being prohibitable from execution using the Lockdown command;
- b) currently prohibited from execution if received on an Admin Submission Queue; or
- c) currently prohibited from execution if received out-of-band on a Management Endpoint.

If the Command and Feature Lockdown capability is supported (i.e., the CFLS bit in the OACS field in Figure 312 is set to '1'), then the controller shall support the Lockdown command and the Command and Feature Lockdown log page.



### 8.1.6 Controller Data Queue

A Controller Data Queue (CDQ) is used to post information from the controller to the host that is specific to the type of queue being created (refer to Figure 165) by the Controller Data Queue command that specifies the Create Controller Data Queue management operation (refer to section 5.1.4.1.1). A CDQ is a circular buffer with a fixed slot size with entries posted by a controller.

A CDQ is deleted:

- by a Controller Data Queue command that specifies the Delete Controller Data Queue management operation (refer to section 5.1.4.1.2);
- on a Controller Level Reset on the controller that processed the Controller Data Queue command that created the CDQ; or
- by a Migration Send command that specifies the Suspend management operation and the Delete User Data Migration Queue (DUDMQ) bit is set to '1' (refer to section 5.1.17.1.1).

On the successful completion of a Controller Data Queue command, a Controller Data Queue Identifier (CDQID) is returned in the completion queue entry (refer to Figure 168). A CDQID is used to identify the created Controller Data Queue on the controller that processed the command. The returned value for that CDQID is any value except the CDQID values for any Controller Data Queue that exists on the controller that processed the command (i.e., a controller is permitted to return the value of a CDQID value of a Controller Data Queue that was previously deleted). The scope of a CDQID is per controller.

The CDQ Head pointer is updated by the host by issuing a Set Features command specifying the Controller Data Queue feature and the Controller Data Queue Identifier (refer to section 5.1.25.1.23) after processing an entry indicating the last free CDQ slot. A Controller Data Queue Phase Tag (CDQP) bit is defined in the CDQ entry (refer to the applicable I/O command set specification) to indicate whether an entry has been newly posted without the host relying on the Controller Data Queue Tail Pointer event (refer to Figure 154). The Controller Data Queue Phase Tag bit enables the host to determine whether entries are new or not.

#### 8.1.6.1 Controller Data Queue Usage

The controller uses the current Tail entry pointer to identify the next open CDQ slot. The controller increments the Tail entry pointer after placing the new entry to the next open CDQ slot. If the Tail entry pointer increment exceeds the CDQ size, then the Tail entry pointer shall roll to zero. The controller may continue to place entries in free CDQ slots as long as the Full queue condition is not met (refer to section 3.3.1.5). The controller shall take CDQ wrap conditions into account.

The host uses the current Head entry pointer to identify the slot containing the next entry to be consumed. The host increments the Head entry pointer after consuming the next entry from the CDQ. If the Head entry pointer increment exceeds the CDQ size, the Head entry pointer shall roll to zero. The host may continue to consume entries from the CDQ as long as the Empty queue condition is not met (refer to section 3.3.1.4).

Note: The host shall take CDQ wrap conditions into account.

A host issues a Set Features command specifying the Controller Data Queue feature to communicate a new value of the Head entry pointer to the controller. If the host specifies an invalid value to the Head entry pointer, then that Set Features command is aborted. This condition may be caused by a host attempting to remove an entry from an empty CDQ.

A host checks a posted entries Controller Data Queue Phase Tag (CDQP) bits in memory to determine whether new CDQ entries have been posted (refer to section 8.1.6.2). The CDQ Tail pointer is only used internally by the controller and is not visible to the host.

An entry is posted to the CDQ when the controller write of that entry to the next free CDQ slot inverts the Controller Data Queue Phase Tag (CDQP) bit from its previous value in memory (refer to section 8.1.6.2). The controller generates a Controller Data Queue Tail Pointer event (refer to Figure 154) to the host to indicate that the CDQ slot specified by the host in the Controller Data Queue feature (refer to section 5.1.25.1.23) has been posted if in the Controller Data Queue feature associated with the CDQ:

- the Tail pointer value matches the value in the Tail Pointer Trigger (TPT) field; and

- the Enable Tail Pointer Trigger (ETPT) bit is set to '1'.

A CDQ entry has been consumed by the host when the host submits a Set Features command specifying the Controller Data Queue feature with a new value that indicates that the CDQ Head Pointer has moved past the slot in which that CDQ entry was placed. A Set Features command specifying the Controller Data Queue feature may indicate that one or more CDQ entries have been consumed.

Altering a CDQ entry after that entry has been posted but before that entry has been consumed results in undefined behavior.

Refer to the specific type of CDQ to determine the behavior of a CDQ that has become full.

Refer to section 3.3.1.4 for the definition of an empty CDQ. Refer to section 3.3.1.5 for the definition of a full CDQ.

If a CDQ entry is constructed via multiple writes, the Controller Data Queue Phase Tag (CDQP) bit shall be updated in the last write of that CDQ entry.

Refer to the applicable I/O Command Set specifications for any specific requirements on the use of CDQs

### 8.1.6.2 Controller Data Queue Phase Tag

The Controller Data Queue Phase Tag (CDQP) bit indicates whether a CDQ entry is new. The CDQP bit for each CDQ entry in the CDQ shall be initialized to '0' by the host before creating the CDQ by submitting a Controller Data Queue command (refer to section 5.1.4) specifying the Create Queue management operation (refer to section 5.1.4.1.1).

When the controller posts a new CDQ entry to the CDQ, the controller shall invert the CDQP bit in that CDQ entry (i.e., the inverting of the CDQP bit enables the host to detect the new CDQ entry).

When a CDQ entry is posted to a CDQ slot in the CDQ for the first time after the CDQ is created, the CDQP bit for that completion queue entry is set to '1'.

This continues for each CDQ entry that is posted until the controller posts a CDQ entry to the highest numbered CDQ slot and wraps to CDQ slot number 0 as described in section 8.1.6.1. When that CDQ wrap condition occurs, the CDQP bit is then cleared to '0' in each CDQ entry that is posted. This continues until another CDQ wrap condition occurs. Each time a CDQ wrap condition occurs, the value of the CDQP bit is inverted (i.e., changes from '1' to '0' or changes from '0' to '1').

### 8.1.7 Device Self-test Operations

A device self-test operation is a diagnostic testing sequence that tests the integrity and functionality of the controller and may include testing of the media associated with namespaces or refresh operations. The operation is broken down into a series of segments, where each segment is a set of vendor specific tests or refresh operations. The segment number in the Self-test Result data structure (refer to section 5.1.12.1.7) is used for reporting purposes to indicate where a test failed, if any. The test performed in each segment may be the same for the short device self-test operation and the extended device self-test operation.

A device self-test operation is performed in the background allowing concurrent processing of some commands and requiring suspension of the device self-test operation to process other commands. Which commands may be processed concurrently versus require suspension of the device self-test operation is vendor specific.

If the controller receives any command that requires suspension of the device self-test operation to process and complete, then the controller shall:

- 1) suspend the device self-test operation;
- 2) process and complete that command; and
- 3) resume the device self-test operation.

During a device self-test operation, the performance of the NVM subsystem may be degraded (e.g., controllers not performing the device self-test operation may also experience degraded performance).

The following device self-test operations are defined:

- a) short device self-test operation (refer to section 8.1.7.1);
- b) extended device self-test operation (refer to section 8.1.7.2); and
- c) Host-Initiated Refresh operation (refer to section 8.1.11).

Figure 595 is an informative example of a device self-test operation with the associated segments and tests performed in each segment.

**Figure 595: Example Device Self-test Operation (Informative)**

Segment		Test Performed	Failure Criteria
1 – RAM Check		Write a test pattern to RAM, followed by a read and compare of the original data.	Any uncorrectable error or data miscompare
2 – SMART Check		Check SMART or health status for Critical Warning bits set to '1' in SMART / Health Information Log.	Any Critical Warning bit set to '1' fails this segment
3 – Volatile memory backup		Validate volatile memory backup solution health (e.g., measure backup power source charge and/or discharge time).	Significant degradation in backup capability
4 – Metadata validation		Confirm/validate all copies of metadata.	Metadata is corrupt and is not recoverable
5 – NVM integrity		Write/read/compare to reserved areas of each NVM. Ensure also that every read/write channel of the controller is exercised.	Data miscompare
Extended only	6 – Data Integrity	Perform background housekeeping tasks, prioritizing actions that enhance the integrity of stored data.	Metadata is corrupt and is not recoverable
		Exit this segment in time to complete the remaining segments and meet the timing requirements for extended device self-test operation indicated in the Identify Controller data structure.	
7 – Media Check		Perform random reads from every available good physical block.  Exit this segment in time to complete the remaining segments. The time to complete is dependent on the type of device self-test operation.	Inability to access a physical block
8 – Drive Life		End-of-life condition: Assess the drive's suitability for continuing write operations.	The Percentage Used is set to 255 in the SMART / Health Information Log or an analysis of internal key operating parameters indicates that data is at risk if writing continues
9 – SMART Check		Same as 2 – SMART Check	

### 8.1.7.1 Short Device Self-Test Operation

A short device self-test operation should complete in two minutes or less. The percentage complete of the short device self-test operation is indicated in the Current Percentage Complete field in the Device Self-test log page (refer to section 5.1.12.1.7).

A short device self-test operation:

- a) shall be aborted by any Controller Level Reset that affects the controller on which the device self-test is being performed;
- b) shall be aborted by a Format NVM command as described in Figure 596;
- c) shall be aborted when a sanitize operation is started (refer to section 5.1.22);
- d) shall be aborted if a Device Self-test command with the Self-Test Code field set to Fh is processed; and
- e) may be aborted if the specified namespace is removed from the namespace inventory.

**Figure 596: Format NVM command Aborting a Device Self-Test Operation**

SES	FNS Bit	SENS Bit	NSID in Format NVM command	NSID in Device Self-test command	Abort Device Self-Test operation?
000b (i.e., not a secure erase)	0	N/A	Any allocated NSID value (refer to section 3.2.1.3)	Any active NSID value (refer to section 3.2.1.4)	Yes, if the NSID values are the same
	0		FFFFFFFFh	Any active NSID value (refer to section 3.2.1.4)	Yes
	0		Any allocated NSID value (refer to section 3.2.1.3)	FFFFFFFFh	Optional
	0		FFFFFFFFh	FFFFFFFFh	Yes
	1		Ignored	Ignored	Yes
001b or 010b (i.e., secure erase)	N/A	0	Any allocated NSID value (refer to section 3.2.1.3)	Any active NSID value (refer to section 3.2.1.4)	Yes, if the NSID values are the same
		0	FFFFFFFFh	Any active NSID value (refer to section 3.2.1.4)	Yes
		0	Any allocated NSID value (refer to section 3.2.1.3)	FFFFFFFFh	Optional
		0	FFFFFFFFh	FFFFFFFFh	Yes
		1	Ignored	Ignored	Yes
Key: Optional = The device self-test operation is not required to be aborted but may be aborted.					

### 8.1.7.2 Extended Device Self-Test Operation

An extended device self-test operation should complete in the time indicated in the Extended Device Self-test Time field in the Identify Controller data structure or less. The percentage complete of the extended device self-test operation is indicated in the Current Percentage Complete field in the Device Self-test log page (refer to section 5.1.12.1.7).

An extended device self-test operation shall persist across any Controller Level Reset and shall resume after completion of the reset or any restoration of power, if any. The segment where the extended device self-test operation resumes is vendor specific, but implementations should only have to perform tests again within the last segment that was being tested prior to the reset.

An extended device self-test operation:

- a) shall be aborted by a Format NVM command as described in Figure 596;
- b) shall be aborted when a sanitize operation is started (refer to section 5.1.22);
- c) shall be aborted if a Device Self-test command with the Self-Test Code field set to Fh is processed;  
and
- d) may be aborted if the specified namespace is removed from the namespace inventory.

### 8.1.8 Directives

Directives is a mechanism to enable host and NVM subsystem or controller information exchange. The Directive Receive command (refer to section 5.1.6) is used to transfer data related to a specific Directive Type from the controller to the host. The Directive Send command (refer to section 5.1.7) is used to transfer data related to a specific Directive Type from the host to the controller. Other commands may include a Directive Specific value specific for a given Directive Type (e.g., the Write command in the NVM Command Set).

Support for Directives is optional and is indicated by the Directives Supported (DIRS) bit in the Optional Admin Command Support (OACS) field in the Identify Controller data structure (refer to Figure 312).

If a controller supports Directives, then the controller shall:

- Indicate support for Directives in the Optional Admin Command Support (OACS) field by setting the DIRS bit to '1' in the Identify Controller data structure;
- Support the Directive Receive command;
- Support the Directive Send command; and
- Support the Identify Directive (i.e., Type 00h).

The Directive Types that may be supported by a controller are defined in Figure 597.

The Directive Specific field and Directive Operation field are dependent on the Directive Type specified in the command (e.g., Directive Send, Directive Receive, or I/O command).

**Figure 597: Directive Types**

Directive	Directive Type Value	Reference	I/O Command Directive
Identify	00h	8.1.8.2	No
Streams	01h	8.1.8.3	Yes
Data Placement	02h	8.1.8.4	Yes
Vendor Specific	0Fh		Yes

If a Directive is not supported or is supported and disabled, then all Directive Send commands and Directive Receive commands with that Directive Type shall be aborted with a status code of Invalid Field in Command.

Support for a specific Directive Type is indicated using the Return Parameters operation of the Identify Directive. A specific Directive may be enabled or disabled using the Enable operation of the Identify Directive. Before using a specific Directive, the host should determine if that Directive is supported and should enable that Directive using the Identify Directive.

#### 8.1.8.1 Directive Use in I/O Commands

I/O Command Directives are the subset of Directive Types that may be used as part of I/O commands. For example, a Write command in the NVM Command Set may specify a Directive Type and an associated Directive Specific value. I/O Command Directives shall have a Directive Type value that is less than or equal to 0Fh due to the size of the Directive Type field in I/O commands. When a Directive Type is specified in an I/O command, the most significant four bits are assumed to be 0h. A Directive Type of 00h in an I/O command specifies that the I/O command is not using Directives.

In an I/O command, if the Directive Type (DTYPE) field is set to an I/O Command Directive, then the Directive Specific (DSPEC) field includes additional information for the associated I/O command (refer to Figure 598).

**Figure 598: Directive Specific Field Interpretation**

Directive Type Value	Directive Specific Field Definition
00h (Directives not in use)	Field not used.
01h (Streams)	Specifies the identifier of the stream associated with the data.
02h (Data Placement)	Specifies the Placement Identifier used to determine where to write the user data within non-volatile storage (refer to Figure 282 and Figure 283) of the Endurance Group associated with the namespace.
03h to 0Eh	Reserved
0Fh (Vendor Specific)	Vendor specific

In an I/O command:

- if no I/O Command Directive is enabled or the DTYPE field is cleared to 00h, then the DTYPE field and the DSPEC field are ignored; and

- if one or more I/O Command Directives is enabled and the DTYPE field is set to a value that is not supported or not enabled, then the controller shall abort the command with a status code of Invalid Field in Command.

For the Streams Directive (i.e., DTYPE field set to 01h), if the DSPEC field is cleared to 0h in an I/O command that supports the Streams Directive, then that I/O command shall be processed normally (i.e., as if DTYPE field is cleared to 00h).

### 8.1.8.2 Identify (Directive Type 00h)

The Identify Directive is used to determine the Directive Types that the controller supports and to enable use of the supported Directives. If Directives are supported, then this Directive Type shall be supported.

The Directive operations that shall be supported for the Identify Directive are listed in Figure 599.

**Figure 599: Identify Directive – Directive Operations**

Directive Command	Directive Operation Name	Directive Operation Value	Reference
Directive Receive	Return Parameters	01h	8.1.8.2.1.1
	Reserved	All other values	
Directive Send	Enable Directive	01h	8.1.8.2.2.1
	Reserved	All other values	

#### 8.1.8.2.1 Directive Receive

This section defines operations used with the Directive Receive command for the Identify Directive.

##### 8.1.8.2.1.1 Return Parameters (Directive Operation 01h)

This operation returns a data structure that contains a bit vector specifying the Directive Types supported by the controller and a bit vector specifying the Directive Types enabled for the namespace. The data structure returned is defined in Figure 600. If an NSID value of FFFFFFFFh is specified, then the controller shall abort the command with a status code of Invalid Field in Command. The DSPEC field in command Dword 11 is not used for this operation.

**Figure 600: Identify Directive – Return Parameters Data Structure**

Bytes	Bits	Description
<b>Directives Supported</b>		
31:00	255:16	Reserved
	15	<b>Vendor Specific Directive (VSDIRS):</b> This bit is set to '1' if the Vendor Specific Directive is supported. This bit is cleared to '0' if the Vendor Specific Directive is not supported.
	14:03	Reserved
	02	<b>Data Placement Directive (DPDIRS):</b> This bit is set to '1' if the Data Placement Directive is supported. This bit is cleared to '0' if the Data Placement Directive is not supported.
	01	<b>Streams Directive (SDIRS):</b> This bit is set to '1' if the Streams Directive is supported. This bit is cleared to '0' if the Streams Directive is not supported.
	00	<b>Identify Directive (IDIRS):</b> This bit shall be set to '1' to indicate that the Identify Directive is supported.
<b>Directives Enabled</b>		
63:32	255:16	Reserved
	15	<b>Vendor Specific Directive (VSDIRE):</b> This bit is set to '1' if the Vendor Specific Directive is enabled. This bit is cleared to '0' if the Vendor Specific Directive is not enabled.
	14:03	Reserved
	02	<b>Data Placement Directive (DPDIRE):</b> This bit is set to '1' if the Data Placement Directive is enabled. This bit is cleared to '0' if the Data Placement Directive is not enabled.
	01	<b>Streams Directive (SDIRE):</b> This bit is set to '1' if the Streams Directive is enabled. This bit is cleared to '0' if the Streams Directive is not enabled.
	00	<b>Identify Directive (IDIRE):</b> This bit shall be set to '1' to indicate that the Identify Directive is enabled.
<b>Directive Persistent Across Controller Level Resets</b>		

**Figure 600: Identify Directive – Return Parameters Data Structure**

Bytes	Bits	Description
95:64	255:16	Reserved
	15	<p><b>Vendor Specific Directive (VSDIRCLR):</b> If the Vendor Specific Directive is supported, then this bit is:</p> <ul style="list-style-type: none"> <li>set to '1' to indicate that the host specified Data Placement Directive state is preserved across Controller Level Resets; or</li> <li>cleared to '0' to indicate that the host specified Data Placement Directive state is not preserved across Controller Level Resets.</li> </ul> <p>If the Vendor Specific Directive is not supported, then this bit shall be cleared to '0'.</p>
	14:03	Reserved
	02	<b>Data Placement Directive (DPDIRCLR):</b> If the Data Placement Directive is supported, then this bit shall be set to '1' to indicate that the host specified Data Placement Directive state is preserved across Controller Level Resets. If the Data Placement Directive is not supported, then this bit shall be cleared to '0'.
	01	<b>Streams Directive (SDIRCLR):</b> This bit shall be cleared to '0' to indicate that the Streams Directive state is not preserved across Controller Level Resets.
	00	<b>Identify Directive (IDIRCLR):</b> This bit shall be set cleared to '0' as the host is not able to change the state of the Identify Directive.
4095:96	n/a	Reserved

### 8.1.8.2.2 Directive Send

This section defines operations used with the Directive Send command for the Identify Directive.

#### 8.1.8.2.2.1 Enable Directive (Directive Operation 01h)

The Enable Directive operation is used to enable a specific Directive for use within a namespace by all controllers that are associated with the same Host Identifier. The DSPEC field in command Dword 11 is not used for this operation. The Identify Directive is always enabled. The enable state of each Directive on each shared namespace attached to enabled controllers associated with the same non-zero Host Identifier is the same. If the Directive is not the Data Placement Directive and an NSID value of FFFFFFFFh is specified, then the Enable Directive operation applies to the NVM subsystem (i.e., all namespaces and all controllers associated with the NVM subsystem). If the Directive is the Data Placement Directive and an NSID value of FFFFFFFFh is specified, then the controller shall abort the command with a status code of Invalid Namespace or Format.

On a Controller Level Reset:

- all Directives other than the Identify Directive that have the Directive Persistent Across Controller Level Resets bit cleared to '0' are disabled for that controller; and
- if there is an enabled controller associated with the Host Identifier for the controller that was reset, then for namespaces attached to enabled controllers associated with that Host Identifier, Directives are not disabled.

If a host sets the Host Identifier of a controller to the same non-zero Host Identifier as one or more other controllers in the NVM subsystem, then setting that Host Identifier shall result in each shared namespace attached to that controller having the same enable state for each Directive as the enable state for each Directive for that namespace attached to other controllers associated with that Host Identifier.

If a host enables a controller that has the same non-zero Host Identifier as one or more other controllers in the NVM subsystem, then enabling that controller shall result in each shared namespace attached to that controller having the same enable state for each Directive as the enable state for each Directive for that namespace attached to other controllers associated with that Host Identifier.

For all controllers in an NVM subsystem that have the same non-zero Host Identifier, if a host changes the enable state of any Directive for a shared namespace attached to a controller by a means other than a

Controller Level Reset, then that change shall be made to the enable state of that Directive for that namespace attached to any other controller associated with that Host Identifier.

If the Host Identifier value is 0h and the Host Identifier is required to be set to a non-zero value before a host enables a Directive (i.e., the Directive Type is set to Streams and the SRNZID bit in the NVM subsystem Stream Capability field (refer to Figure 607) is set to '1') then the Directive Send command shall be aborted with a status code of Host Identifier Not Initialized.

Refer to the sections defining each Directive Type for restrictions on enabling that directive.

**Figure 601: Enable Directive – Command Dword 12**

Bits	Description
31:16	Reserved
15:08	<b>Directive Type (DTYPE):</b> This field specifies the Directive Type to enable or disable. If this field specifies the Identify Directive (i.e., 00h), then a status code of Invalid Field in Command shall be returned.
07:01	Reserved
00	<b>Enable Directive (ENDIR):</b> If this bit is set to '1' and the Directive Type is supported, then the Directive is enabled, unless otherwise specified. If this bit is cleared to '0', then the Directive is disabled. If this bit is set to '1' for a Directive that is not supported, then a status code of Invalid Field in Command shall be returned.

**Figure 602: Enable Directive – Command Specific Status Values**

Value	Description
7Fh	<b>Stream Resource Allocation Failed:</b> The controller failed to enable the Streams Directive for the specified namespace and Host Identifier due to insufficient resources.

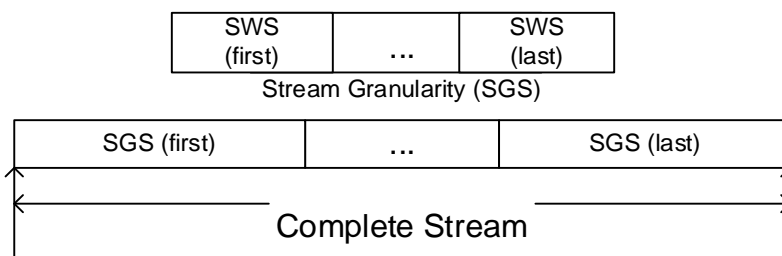
### 8.1.8.3 Streams (Directive Type 01h, Optional)

The Streams Directive enables the host to indicate (i.e., by using the stream identifier) to the controller that the specified user data in a User Data Out command (e.g., logical blocks in a write command) are part of one group of associated data. This information may be used by the controller to store related data in associated locations or for other performance enhancements.

The controller provides information in response to the Return Parameters operation about the configuration of the controller that indicates Stream Write Size, Stream Granularity Size, and stream resources at the NVM subsystem and namespace levels.

Data that is aligned to and in multiples of the Stream Write Size (SWS) provides optimal performance of the write commands to the controller. The SWS unit of granularity is defined independently for each I/O Command Set. The Stream Granularity Size indicates the size of the media that is prepared as a unit for future allocation for write commands and is a multiple of the Stream Write Size. The controller may allocate and group together a stream in Stream Granularity Size (SGS) units. Refer to Figure 603.

**Figure 603: Directive Streams – Stream Alignment and Granularity**



One example of this is if the host issues an NVM Command Set Dataset Management command (refer to the Dataset Management command section of the NVM Command Set Specification) to deallocate logical blocks that are associated with a stream, that host should specify a starting LBA and length that is aligned



to and in multiples of the Stream Granularity Size. This provides optimal performance and endurance of the media.

Stream resources are the resources in the NVM subsystem that are necessary to track operations associated with a specified stream identifier. There are a maximum number of stream resources that are available in an NVM subsystem as indicated by the Max Stream Limit (MSL) field in the Return Parameters data structure (refer to Figure 607).

Available NVM subsystem stream resources are stream resources that are not allocated for exclusive use in any namespace. Available NVM subsystem stream resources are reported in the NVM Subsystem Streams Available (NSSA) field (refer to Figure 607) and may be used by any host in any namespace that:

- has the Streams Directive enabled;
- has not been allocated exclusive stream resources by that host if the Shared Stream Identifiers (SSID) bit is cleared to '0' in the NSSC field; and
- has not been allocated exclusive stream resources by any host if the SSID bit is set to '1'.

Each time stream resources are allocated for exclusive use in a specified namespace, the available NVM subsystem stream resources reported in the NSSA field are reduced.

For a given namespace:

- a) a host allocates stream resources to that namespace for the exclusive use of that host(s) by issuing the Allocate Resources operation;
- b) other hosts may concurrently allocate stream resources to that namespace for their exclusive use; and
- c) hosts which have not allocated stream resources to that namespace may use available NVM subsystem stream resources for access to that namespace.

If an NVM subsystem has streams resources allocated to a host with a Host Identifier value of 0h and the Host Identifier is subsequently changed to a non-zero value, then those streams resources remain associated with the Host with a Host Identifier value of 0h and are not associated with the host with the non-zero Host Identifier.

The Directive operations that shall be supported if the Streams Directive is supported are listed in Figure 604. The Directive Specific field in a command is referred to as the Stream Identifier when the Directive Type field is set to the Streams Directive.

**Figure 604: Streams – Directive Operations**

Directive Command	Directive Operation Name	Directive Operation Value	Reference
Directive Receive	Return Parameters	01h	8.1.8.3.1.1
	Get Status	02h	8.1.8.3.1.2
	Allocate Resources	03h	8.1.8.3.1.3
	Reserved	All other values	
Directive Send	Release Identifier	01h	8.1.8.3.2.1
	Release Resources	02h	8.1.8.3.2.2
	Reserved	All other values	

Stream identifiers are assigned by the host and may be in the range 0001h to FFFFh. The host may specify a sparse set of stream identifiers (i.e., there is no requirement for the host to use Stream Identifiers in any particular order).

The host may access a namespace through multiple controllers in the NVM subsystem. The controllers in an NVM subsystem indicate in the SSID bit (refer to Figure 607) if a stream identifier is unique based on the Host Identifier (i.e., the same stream identifier used to access the same namespace by a host that has registered a different Host Identifier is referencing a different stream), or if a stream identifier may be used by multiple Host Identifiers (i.e., the same stream identifier used to access the same namespace by a host

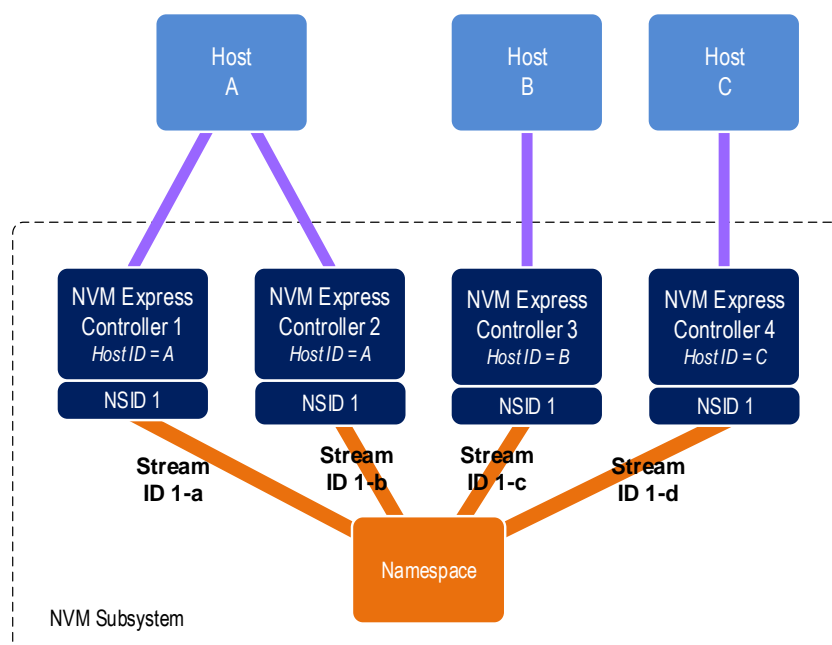
that has registered a different Host Identifier is referencing the same stream). All controllers in an NVM subsystem shall report the same value in the NSSC field.

If multiple controllers receive a registration of a Host Identifier (refer to section 5.1.25.1.28) that has the same non-zero value, then that value represents a single host that is accessing the namespace through those controllers and a stream identifier is used across those controllers to access the same stream on the namespace. If a Host Identifier has a unique non-zero value, then each value represents a unique host that is accessing the namespace and:

- if the SSID bit is cleared to '0', then the same stream identifier on controllers with different non-zero Host Identifiers does not have the same meaning for a particular namespace (i.e., the stream identifier is not used across controllers with different non-zero Host Identifiers to access the same stream on the namespace); and
- if the SSID bit is set to '1', then the same stream identifier on any controller with a non-zero Host Identifier has the same meaning for a particular namespace (i.e., the stream identifier is used across controllers to access the same stream on the namespace).

If a Host Identifier is cleared to 0h, then a unique host is accessing the namespace and the stream identifier does not have the same meaning for a particular namespace.

**Figure 605: Example Multi-Stream and NSSC**



In the example shown in Figure 605, if the SSID bit is cleared to '0', then there are three streams as follows:

- Stream ID 1-a and Stream ID 1-b have the same meaning;
- Stream ID 1-c has a different meaning; and
- Stream ID 1-d has a different meaning.

In the example shown in Figure 605, if the SSID bit is set to '1', then there is one stream as follows:

- Stream ID 1-a, Stream ID 1-b, Stream ID 1-c, and Stream ID 1-d have the same meaning.

The controller(s) recognized by the NVM subsystem as being associated with a specific host or hosts and attached to a specific namespace either:

- utilizes a number of stream resources allocated for exclusive use of that namespace as returned in response to an Allocate Resources operation; or
- utilizes resources from the NVM subsystem stream resources.

The value of Namespace Streams Allocated (NSA) indicates how many resources for individual stream identifiers have been allocated for exclusive use for the specified namespace by the associated controllers. This indicates the maximum number of stream identifiers that may be open at any given time in the specified namespace by the associated controllers. To request a different number of resources than are currently allocated for exclusive use by the associated controllers for a specific namespace, all currently allocated resources are first required to be released using the Release Resources operation. There is no mechanism to incrementally increase or decrease the number of allocated resources for a given namespace.

Streams are opened by the controller when the host issues a Write command that specifies a stream identifier that is not currently open. While a stream is open the controller maintains context for that stream (e.g., buffers for associated data). The host may determine the streams that are open using the Get Status operation.

For a namespace that has a non-zero value of Namespace Streams Allocated (NSA), if the host submits a Write command specifying a stream identifier not currently in use and stream resources are exhausted, then an arbitrary stream identifier for that namespace is released by the controller to free the stream resources associated with that stream identifier for the new stream. The host may ensure the number of open streams does not exceed the allocated stream resources for the namespace by explicitly releasing stream identifiers as necessary using the Release Identifier operation.

For a namespace that has zero namespace stream resources allocated, if the host submits an I/O command specifying a stream identifier not currently in use and:

- NVM subsystem streams available are exhausted, then an arbitrary stream identifier for an arbitrary namespace that is using NVM subsystem stream resources is released by the NVM subsystem to free the stream resources associated with that stream identifier for the new stream; or
- all NVM subsystem stream resources have been allocated for exclusive use for specific namespaces, then the Write command is treated as a normal Write command that does not specify a stream identifier.

The host determines parameters associated with stream resources using the Return Parameters operation. The host may get a list of open stream identifiers using the Get Status operation.

If the Streams Directive becomes disabled for use by a host within a namespace, then all stream resources and stream identifiers shall be released for that host for the affected namespace. If the host issues a Format NVM command, then all stream identifiers for all open streams for affected namespaces shall be released. If the host deletes a namespace, then all stream resources and all stream identifiers for that namespace shall be released. If the write protection state of a namespace changes such that the namespace becomes write protected (refer to section 8.1.16), then the controller shall release all stream resources and stream identifiers for that namespace.

Streams Directive defines the command specific status values specified in Figure 606.

**Figure 606: Streams Directive – Command Specific Status Values**

Value	Definition
7Fh	<b>Stream Resource Allocation Failed:</b> The controller was not able to allocate stream resources for exclusive use of the specified namespace and no NVM subsystem stream resources are available.

The Streams Directive is not allowed to be enabled in any namespace that is contained in an Endurance Group with Flexible Data Placement enabled. If the specified namespace is contained in an Endurance Group that has Flexible Data Placement enabled, then the controller shall not enable the Streams Directive. If:

- a Directive Send command specifies the Streams Directive in the DTYPE field and the Directive Operation field is set to 01h (i.e., Enable Directive); and
- the namespace specified by the NSID field is contained in an Endurance Group that has Flexible Data Placement enabled,

then the controller shall:

- abort that Directive Send command with a status code of Invalid Field in Command; and
- indicate the Directive Operation field in the Parameter Error Location field (refer to Figure 205) if an entry is added to the Error Information log page due to aborting that Directive Send command.

### 8.1.8.3.1 Directive Receive

This section defines operations used with the Directive Receive command for the Streams Directive.

#### 8.1.8.3.1.1 Return Parameters (Directive Operation 01h)

The Return Parameter operation returns a data structure that specifies the features and capabilities supported by the Streams Directive, including namespace specific values. The DSPEC field in command Dword 11 is not used for this operation. The data structure returned is defined in Figure 607. If an NSID value of FFFFFFFFh is specified, then the controller:

- returns the NVM subsystem specific values;
- may return any namespace specific values that are the same for all namespaces (e.g., SWS); and
- clears all other namespace specific fields to 0h.

**Figure 607: Streams Directive – Return Parameters Data Structure**

Bytes	Description								
<b>NVM Subsystem Specific Fields</b>									
01:00	<b>Max Streams Limit (MSL):</b> This field indicates the maximum number of concurrently open streams that the NVM subsystem supports. This field returns the same value independent of specified namespace.								
03:02	<b>NVM Subsystem Streams Available (NSSA):</b> This field indicates the number of NVM subsystem stream resources available. These are the stream resources that are not allocated for the exclusive use by a host in any specific namespace. This field returns the same value independent of specified namespace.								
05:04	<b>NVM Subsystem Streams Open (NSSO):</b> This field indicates the number of open streams in the NVM subsystem that are not associated with a namespace for which resources were allocated using an Allocate Resources operation. This field returns the same value independent of specified namespace.								
06	<b>NVM Subsystem Stream Capability (NSSC):</b> This field indicates the stream capabilities of the NVM subsystem.								
	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:2</td> <td>Reserved</td> </tr> <tr> <td>1</td> <td><b>Streams Require Non-Zero Host Identifier (SRNZID):</b> This bit indicates whether the Host Identifier is required to be set to a non-zero value before the Streams Directive is able to be enabled. If this bit is cleared to '0', then the Host Identifier is not required to be set to a non-zero value before the Streams Directive is able to be enabled. If this bit is set to '1', then the Host Identifier is required to be set to a non-zero value before the Streams Directive is able to be enabled.</td> </tr> <tr> <td>0</td> <td><b>Shared Stream Identifiers (SSID):</b> This bit indicates whether stream identifiers may be shared by multiple Host Identifiers, or if a stream identifier is associated with a single Host Identifier. If this bit is cleared to '0', then the stream identifier is associated with a single non-zero Host Identifier. If this bit is set to '1', then the stream identifier may be associated with multiple non-zero Host Identifiers.</td> </tr> </tbody> </table>	Bits	Description	7:2	Reserved	1	<b>Streams Require Non-Zero Host Identifier (SRNZID):</b> This bit indicates whether the Host Identifier is required to be set to a non-zero value before the Streams Directive is able to be enabled. If this bit is cleared to '0', then the Host Identifier is not required to be set to a non-zero value before the Streams Directive is able to be enabled. If this bit is set to '1', then the Host Identifier is required to be set to a non-zero value before the Streams Directive is able to be enabled.	0	<b>Shared Stream Identifiers (SSID):</b> This bit indicates whether stream identifiers may be shared by multiple Host Identifiers, or if a stream identifier is associated with a single Host Identifier. If this bit is cleared to '0', then the stream identifier is associated with a single non-zero Host Identifier. If this bit is set to '1', then the stream identifier may be associated with multiple non-zero Host Identifiers.
	Bits	Description							
7:2	Reserved								
1	<b>Streams Require Non-Zero Host Identifier (SRNZID):</b> This bit indicates whether the Host Identifier is required to be set to a non-zero value before the Streams Directive is able to be enabled. If this bit is cleared to '0', then the Host Identifier is not required to be set to a non-zero value before the Streams Directive is able to be enabled. If this bit is set to '1', then the Host Identifier is required to be set to a non-zero value before the Streams Directive is able to be enabled.								
0	<b>Shared Stream Identifiers (SSID):</b> This bit indicates whether stream identifiers may be shared by multiple Host Identifiers, or if a stream identifier is associated with a single Host Identifier. If this bit is cleared to '0', then the stream identifier is associated with a single non-zero Host Identifier. If this bit is set to '1', then the stream identifier may be associated with multiple non-zero Host Identifiers.								
15:07	Reserved								
<b>Namespace Specific Fields</b>									
19:16	<b>Stream Write Size (SWS):</b> This field indicates the alignment and size of the optimal stream write as a number for the specified namespace where the unit of granularity is specified by the applicable I/O Command Set. The size indicated should be less than or equal to Maximum Data Transfer Size (MDTS) that is specified in units of minimum memory page size. SWS may change if the namespace is reformatted with a different User Data Format. If the NSID value is set to FFFFFFFFh, then this field may be cleared to 0h if a single user data size cannot be indicated.  Refer to the applicable I/O Command Set specification for how this field is utilized to optimize performance and endurance.								

**Figure 607: Streams Directive – Return Parameters Data Structure**

Bytes	Description
21:20	<p><b>Stream Granularity Size (SGS):</b> This field indicates the stream granularity size for the specified namespace in Stream Write Size (SWS) units. If the NSID value is set to FFFFFFFFh, then this field may be cleared to 0h.</p> <p>Refer to the applicable I/O Command Set specification for how this field is utilized to optimize performance and endurance.</p>
<b>Namespace and Host Identifier Specific Fields</b>	
23:22	<p><b>Namespace Streams Allocated (NSA):</b> This field indicates the number of stream resources allocated for exclusive use of the specified namespace.</p> <p>If the SSID bit is cleared to '0' in the NSSC field, then those exclusive stream resources are shared by the controller processing the Return Parameters operation and by all other controllers that share the same non-zero Host Identifier, and are attached to the specified namespace.</p> <p>If the SSID bit is set to '1', then those exclusive stream resources are shared by all controllers that are associated with any non-zero Host Identifier and are attached to this namespace.</p> <p>If this value is non-zero, then the namespace may have up to NSA number of concurrently open streams. If this field is cleared to 0h, then no stream resources are currently allocated to this namespace and the namespace may have up to NSSA number of concurrently open streams.</p>
25:24	<p><b>Namespace Streams Open (NSO):</b> This field indicates the number of open streams in the specified namespace.</p> <p>If the SSID bit is cleared to '0', then this field indicates the number of streams that were opened by the controller processing the Return Parameters operation and by all other controllers that share the same non-zero Host Identifier, and are attached to this namespace.</p> <p>If the SSID bit is set to '1', then this field indicates the number of streams that were opened by the controller processing the Return Parameters operation and all other controllers that are associated with any non-zero Host Identifier and are attached to this namespace.</p> <p>Note: It is not possible for a host to retrieve the number of open streams using resources allocated to the specified namespace by other hosts.</p>
31:26	Reserved

**8.1.8.3.1.2 Get Status (Directive Operation 02h)**

The Get Status operation returns information about the status of currently open streams for the specified namespace and the host issuing the Get Status operation. The DSPEC field in command Dword 11 is not used for this operation.

If the SSID bit is cleared to '0' in the NSSC field, then the information returned describes only those resources for the specified namespace that are associated with hosts that are registered with the same non-zero Host Identifier value as the host issuing the Get Status operation. If the SSID bit is set to '1', then the information returned describes the resources for the specified namespace that are associated with hosts that are registered with any non-zero Host Identifier.

If an NSID value of FFFFFFFFh is specified, then the controller shall return information about the status of currently open streams in the NVM subsystem that use resources which are not allocated for the exclusive use of a particular namespace. If a stream identifier value being returned is in use by different namespaces, then that stream identifier shall be returned only once.

Stream Identifier 1 (i.e., returned at offset 03:02) contains the value of the open stream of lowest numerical value. Each subsequent field contains the value of the next numerically greater stream identifier of an open stream.

The data structure returned is defined in Figure 608. All fields are specific to the specified namespace if the NSID value was not set to FFFFFFFFh.

**Figure 608: Streams Directive – Get Status Data Structure**

Bytes	Description
01:00	<b>Open Stream Count (OSC):</b> This field specifies the number of streams that are currently open.
03:02	<b>Stream Identifier 1 (SID1):</b> This field specifies the stream identifier of the first (numerically lowest) open stream.
05:04	<b>Stream Identifier 2 (SID2):</b> This field specifies the stream identifier of the second open stream.
...	...
131071: 131070	<b>Stream Identifier 65,535 (SID655355):</b> This field specifies the stream identifier of the 65,535th open stream.

#### 8.1.8.3.1.3 Allocate Resources (Directive Operation 03h)

The Allocate Resources operation indicates the number of streams that the host requests for the exclusive use for the specified namespace. If the SSID bit is cleared to '0' in the NSSC field, then those resources are for the exclusive use of hosts that are registered with the same Host Identifier as the host that made the request. If the SSID bit is set to '1', then those resources are for the exclusive use of any host that is registered with any non-zero Host Identifier. The DSPEC field in command Dword 11 is not used for this operation. The operation returns the number of streams allocated in Dword 0 of the completion queue entry. The value allocated may be less than or equal to the number requested. The allocated resources shall be reflected in the Namespace Streams Allocated field of the Return Parameters data structure.

If the controller is unable to allocate any stream resources for the exclusive use for the specified namespace, then the controller shall:

- return a status value of Stream Resource Allocation Failed; or
- if NVM subsystem stream resources are available, then clear NSA to 0h in the completion queue entry to indicate that the host may use stream resources from the NVM subsystem for this namespace.

If the specified namespace already has stream resources allocated for the exclusive use of the host issuing the Allocate Resources operation, then the controller shall return a status code of Invalid Field in Command. To allocate additional streams resources, the host should release resources and request a complete set of resources.

No data transfer occurs.

**Figure 609: Allocate Resources – Command Dword 12**

Bits	Description
31:16	Reserved
15:00	<b>Namespace Streams Requested (NSR):</b> This field specifies the number of stream resources the host is requesting be allocated for exclusive use by the specified namespace.

**Figure 610: Allocate Resources – Completion Queue Entry Dword 0**

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

#### 8.1.8.3.2 Directive Send

This section defines operations used with the Directive Send command for the Streams Directive.

##### 8.1.8.3.2.1 Release Identifier (Directive Operation 01h)

The Release Identifier operation specifies that the stream identifier specified in the DSPEC field in command Dword 11 is no longer in use by the host. Specifically, if the host uses that stream identifier in a

future operation, then that stream identifier is referring to a different stream. If the specified identifier does not correspond to an open stream for the specified namespace, then the Directive Send command should not fail as a result of the specified identifier. If there are stream resources allocated for the exclusive use of the specified namespace, then those exclusive stream resources remain allocated for this namespace and may be re-used in a subsequent write command. If there are no stream resources allocated for the exclusive use of the specified namespace, then the stream resources are returned to the NVM subsystem stream resources for future use by a namespace without exclusive allocated stream resources. If an NSID value of FFFFFFFFh is specified, then the controller shall abort the command with a status code of Invalid Field in Command.

No data transfer occurs.

#### **8.1.8.3.2 Release Resources (Directive Operation 02h)**

The Release Resources operation is used to release all streams resources allocated for the exclusive use of the namespace attached to all controllers:

- associated with the same non-zero Host Identifier of the controller that processed the operation if the SSID bit is cleared to '0' in the NSSC field is cleared to '0'; and
- associated with any non-zero Host Identifier if the SSID bit is set to '1'.

On successful completion of this command, the exclusive allocated stream resources are released and the Namespace Streams Allocated (refer to Figure 607) field is cleared to 0h for the specified namespace. If this command is issued when no streams resources are allocated for the exclusive use of the namespace, then the Directive Send command shall take no action and shall not fail as a result of no allocated stream resources.

No data transfer occurs.

#### **8.1.8.4 Data Placement (Directive Type 02h, Optional)**

The Data Placement Directive enables the host to specify to the controller the Reclaim Unit (refer section 8.1.10) to place the user data in I/O commands specified by the appropriate I/O Command Set specification.

The Data Placement Directive has no Directive Operations defined. Any Directive Receive command or Directive Send command that specifies a Data Placement Directive Type shall be aborted by the controller with a status code of Invalid Field in Command.

If the specified namespace is not contained in an Endurance Group with Flexible Data Placement enabled, then the controller shall abort the Directive Send command with a status code of FDP Disabled.

#### **8.1.9 Dispersed Namespaces**

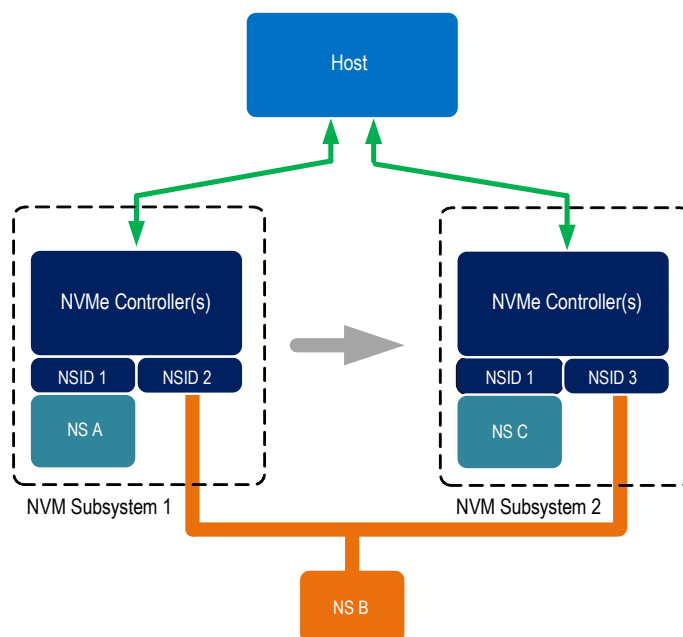
Dispersed namespaces provide hosts with access to the same namespace using multiple participating NVM subsystems. A typical Dispersed Namespace implementation uses Fabrics based NVM subsystems. Two prominent scenarios that require namespace access from multiple participating NVM subsystems are:

1. online data migration; and
2. data replication.

Online data migration is used to transparently move the contents of one or more namespaces across participating NVM subsystems without disrupting host access to those namespaces during the migration. For each namespace whose contents are migrated, one or more paths to the namespace in the destination NVM subsystem are added to the multi-path I/O on the host, in addition to the existing paths to the namespace in the source NVM subsystem. Once the online data migration completes (i.e., all of the contents of all namespaces being migrated has been successfully migrated from the source NVM subsystem to the destination NVM subsystem), all of the paths to the namespace in the source NVM subsystem are subsequently removed from the multi-path I/O on the host. During the online data migration, the host and participating NVM subsystems must manage the paths so that there is no interruption to I/O. Figure 611 shows an example of online data migration, where the contents of namespace B are migrated from one participating NVM subsystem to another participating NVM subsystem. In this figure, the lines

from the host to the NVMe controllers may represent multiple connections from the host to multiple controllers in each participating NVM subsystem.

**Figure 611: Online Data Migration**



Data replication is used to replicate the contents of one or more dispersed namespaces across participating NVM subsystems such that each participating NVM subsystem is able to provide host access to the same namespace data. Other behavior (e.g., persistent reservations) must also be maintained between hosts and each participating NVM subsystem. Figure 612, Figure 613, and Figure 614 show different examples of data replication, where the contents of namespace B are replicated across two participating NVM subsystems. In these figures, the lines from hosts to NVMe controllers may represent multiple connections from the hosts to multiple controllers in each participating NVM subsystem. Figure 612 shows an example of single NVM subsystem host connectivity, where two hosts access the same dispersed namespace being replicated across two participating NVM subsystems via controllers in only one NVM subsystem.



**Figure 612: Data Replication Example 1**

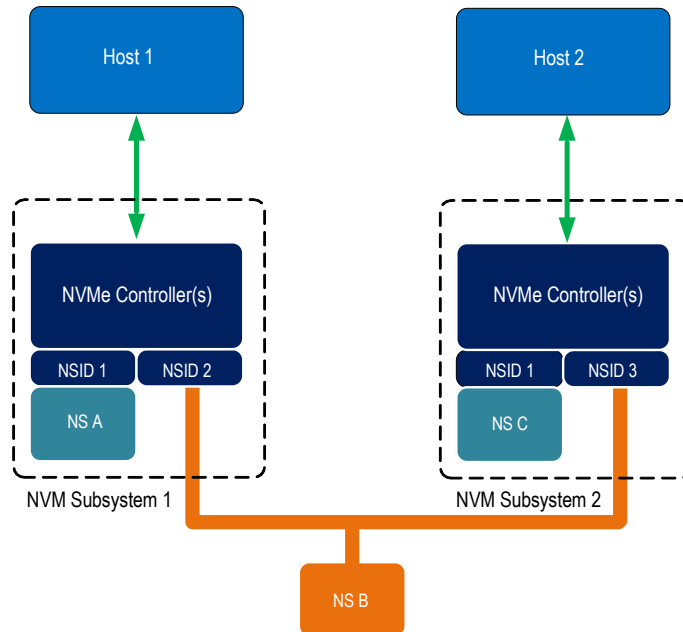


Figure 613 shows another example of single NVM subsystem host connectivity, where two hosts access the same dispersed namespace being replicated across two participating NVM subsystems via controllers in only one NVM subsystem, but the host connected to NVM Subsystem 2 only accesses namespace B in the event of a failure scenario.

**Figure 613: Data Replication Example 2**

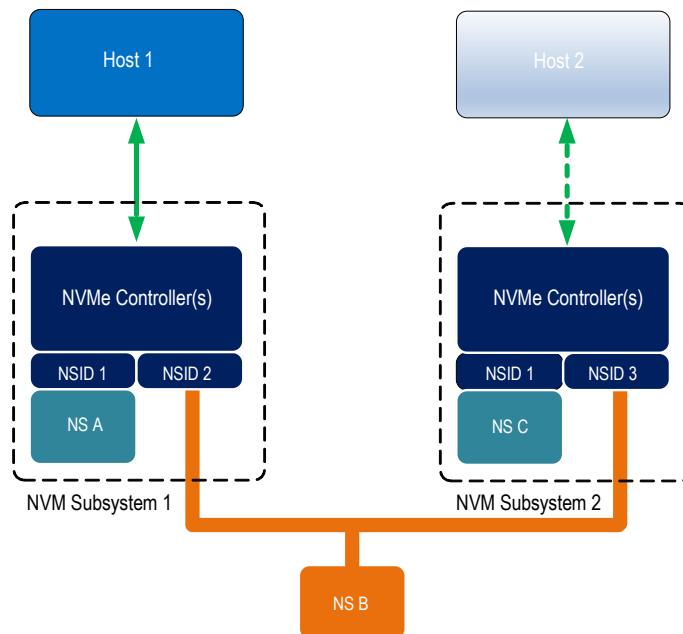
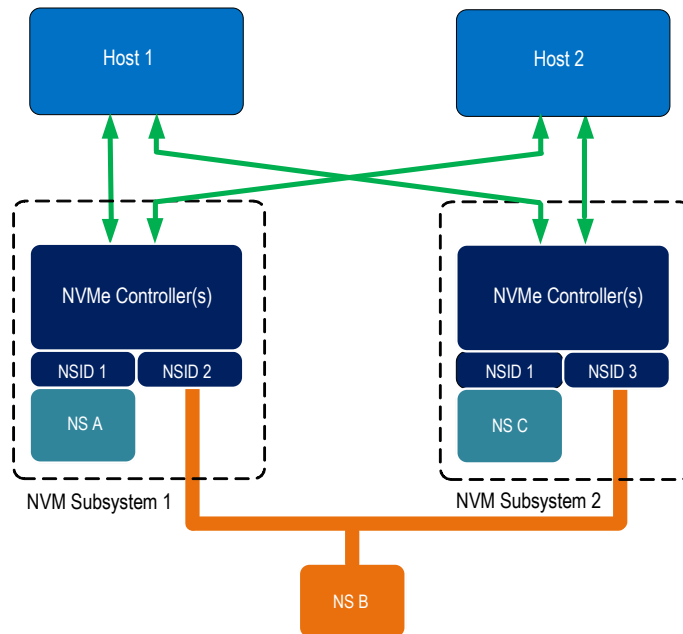


Figure 614 shows an example of multiple NVM subsystem host connectivity, where two hosts access the same dispersed namespace being replicated across two participating NVM subsystem via controllers in both NVM subsystems.

**Figure 614: Data Replication Example 3**

### 8.1.9.1 Dispersed Namespace Management

The Namespace Management command (refer to section 5.1.21) may be used to create a namespace that is capable of being accessed using controllers contained in two or more participating NVM subsystems concurrently (i.e., may be used to create a namespace that is capable of being a dispersed namespace). The methods for dispersing a namespace (e.g., attaching that namespace to controllers contained in separate participating NVM subsystems) are outside the scope of this specification. The Namespace Management or Capacity Management command may be used to delete a dispersed namespace on the participating NVM subsystem containing the controller processing the command. Deleting a dispersed namespace on one participating NVM subsystem may or may not affect that namespace on other participating NVM subsystems (e.g., a deletion from one participating NVM subsystem may have no effect on the namespace on other participating NVM subsystems, or a deletion from one participating NVM subsystem may delete the namespace from all participating NVM subsystems).

The Namespace Management command may be used to create a shared namespace that is not a dispersed namespace which is later converted to a dispersed namespace. The method of converting a namespace that is not a dispersed namespace to a dispersed namespace is outside the scope of this specification. NVM subsystems should not convert private namespaces to dispersed namespaces. Whenever a namespace that is not a dispersed namespace is converted to a dispersed namespace or a dispersed namespace is converted to a namespace that is not a dispersed namespace, the controller reports a Namespace Attribute Changed event as described in Figure 151.

The host may attach or detach a dispersed namespace from the controller by using the Namespace Attachment command (refer to section 5.1.20).

### 8.1.9.2 NSID and Globally Unique Namespace Identifier Usage

The NSID for a dispersed namespace is unique for all controllers in a participating NVM subsystem, as described in section 3.2.1.6. The NSID for a dispersed namespace may be different on different participating NVM subsystems.

A dispersed namespace has a globally unique namespace identifier (refer to section 4.7.1.4) that uniquely identifies the namespace in all participating NVM subsystems. The globally unique namespace identifier has the same value for that namespace in all participating NVM subsystems and shall be used to determine which NSIDs on different participating NVM subsystems refer to the same namespace. The globally unique

namespace identifier that uniquely identifies the dispersed namespace in all participating NVM subsystems shall be either:

- a) a Namespace Globally Unique Identifier (NGUID); or
- b) a Universally Unique Identifier (UUID).

If the dispersed namespace has an NGUID and does not have a UUID, then:

- a) that NGUID shall be the globally unique namespace identifier that uniquely identifies the dispersed namespace in all participating NVM subsystems;
- b) all participating NVM subsystems shall use the same NGUID value for the dispersed namespace; and
- c) the dispersed namespace may have an EUI64 that shall not be the globally unique namespace identifier that uniquely identifies the dispersed namespace in all participating NVM subsystems.

If the dispersed namespace has a UUID and does not have an NGUID, then:

- a) that UUID shall be the globally unique namespace identifier that uniquely identifies the dispersed namespace in all participating NVM subsystems;
- b) all participating NVM subsystems shall use the same UUID value for the dispersed namespace; and
- c) the dispersed namespace may have an EUI64 that shall not be the globally unique namespace identifier that uniquely identifies the dispersed namespace in all participating NVM subsystems.

If the dispersed namespace has an NGUID and has a UUID, then:

- a) the NGUID shall be the globally unique namespace identifier that uniquely identifies the dispersed namespace in all participating NVM subsystems;
- b) all participating NVM subsystems shall use the same NGUID value for the dispersed namespace; and
- c) the dispersed namespace may have an EUI64 that shall not be the globally unique namespace identifier that uniquely identifies the dispersed namespace in all participating NVM subsystems.

### 8.1.9.3 Dispersed Namespace Access

The host may discover if a namespace is capable of being accessed using controllers contained in two or more participating NVM subsystems concurrently (i.e., may discover if a namespace may be a dispersed namespace) by issuing an Identify command (i.e., CNS 00h if supported by the I/O Command Set or CNS 08h) to the controller for the specified NSID. If the namespace is capable of being accessed using controllers contained in two or more participating NVM subsystem concurrently, then the controller shall set the Dispersed Namespace (DISNS) bit to '1' in the Namespace Multi-path I/O and Namespace Sharing Capabilities (NMIC) field of the returned I/O Command Set Independent Identify Namespace data structure (refer to Figure 319) or Identify Namespace data structure (refer to the NVM Command Set Specification).

All controllers in each participating NVM subsystem that are able to provide access to a specific dispersed namespace shall support the I/O Command Set associated with that dispersed namespace.

The host may discover the NQNs of participating NVM subsystems which contain controllers that are able to provide access to a dispersed namespace by issuing a Get Log Page command for the Dispersed Namespace Participating NVM Subsystems log page (i.e., the Log Page Identifier (LID) field set to 17h (refer to section 5.1.12.1.23)).

Hosts specify support for dispersed namespaces by setting the Host Dispersed Namespace Support (HDISNS) field to 1h in the Host Behavior Support feature by issuing a Set Features command with Feature Identifier (FID) set to 16h (refer to Figure 407).

If the HDISNS field is set to 1h, the host specifies support for:

- treating all instances of a dispersed namespace as the same namespace, based upon the shared globally unique namespace identifier (refer to section 8.1.9.2) when accessing that dispersed namespace through multiple controllers on the same participating NVM subsystem;

- treating all instances of a dispersed namespace as the same namespace, based upon the shared globally unique namespace identifier when accessing that dispersed namespace through controllers on multiple participating NVM subsystems; and
- specifying a Host Identifier that is unique across all hosts that connect to any participating NVM subsystem.

If the HDISNS field is set to 1h, the controller shall not abort any command that the host submits to dispersed namespaces with a status code of Host Dispersed Namespace Support Not Enabled.

A participating NVM subsystem shall support prohibiting host access to dispersed namespaces when the HDISNS field is cleared to 0h in the Host Behavior Support feature, as described in section 8.1.9.4. A participating NVM subsystem may either:

- allow host access to a dispersed namespace when the HDISNS field is cleared to 0h in the Host Behavior Support feature if the host associated with the Host NQN in the connection is only able to access the dispersed namespace from a single participating NVM subsystem; or
- prohibit host access to dispersed namespaces when the HDISNS field is cleared to 0h in the Host Behavior Support feature.

A participating NVM subsystem that allows host access to a dispersed namespace when the HDISNS field is cleared to 0h in the Host Behavior Support feature if the host associated with the Host NQN in the connection is only able to access the dispersed namespace from a single participating NVM subsystem shall implement a method to determine if the host is able to access the dispersed namespace from multiple participating NVM subsystems. The method used to determine if the host is able to access the dispersed namespace from multiple participating NVM subsystems is outside the scope of this specification. If a participating NVM subsystem determines that a Connect command requests a host with the HDISNS field cleared to 0h in the Host Behavior Support feature having access to the same dispersed namespace through multiple participating NVM subsystems, then that participating NVM subsystem shall either:

- a) abort a Connect command with a status code of Connect Invalid Host if that Connect command requests the host associated with the Host NQN in the connection having access to a dispersed namespace through multiple participating NVM subsystems. Aborting a Connect command for this reason may result in the host being unable to access other namespaces (i.e., namespaces that are unrelated to the dispersed namespace) on that NVM subsystem. This enforces the requirement that a host that has not set the HDISNS field to 1h in the Host Behavior Support feature be allowed to access a dispersed namespace on only one participating NVM subsystem by preventing all access to any other participating NVM subsystems (including all other namespaces on those other participating NVM subsystems); or
- b) allow a Connect command that requests the host associated with the Host NQN in the connection having access to a dispersed namespace through multiple participating NVM subsystems, but prohibit host access to that dispersed namespace from multiple participating NVM subsystems (e.g., prohibit host access to that dispersed namespace on the second participating NVM subsystem) as described in section 8.1.9.4. This enforces the requirement that a host that has not set the HDISNS field to 1h in the Host Behavior Support feature be allowed to access a dispersed namespace on only one participating NVM subsystem by, for each dispersed namespace, preventing access to that dispersed namespace on any other participating NVM subsystems (and allowing access to other namespaces that are not dispersed namespaces on those other participating NVM subsystems).

#### **8.1.9.4 Prohibiting Host Access to Dispersed Namespaces**

If a participating NVM subsystem prohibits host access to dispersed namespaces when the Host Dispersed Namespace Support (HDISNS) field is cleared to 0h in the Host Behavior Support feature, then the controller shall abort the following commands with a status code of Host Dispersed Namespace Support Not Enabled:

- any I/O commands (refer to section 7 in this specification and the I/O Commands section in the appropriate I/O Command Set specification) that specify the NSID of a dispersed namespace in the command; or

- any of the Admin commands listed in Figure 615 that specify the NSID of a dispersed namespace in the command.

The Additional Restrictions column in Figure 615 lists additional restrictions that apply while the HDISNS field is cleared to 0h in the Host Behavior Support feature for Admin commands that are capable of affecting dispersed namespaces without specifying the NSID of a dispersed namespace.

**Figure 615: Dispersed Namespaces Command Restrictions - Prohibited Admin Commands**

Admin Command	Additional Restrictions
Directive Receive	None.
Directive Send	None.
Set Features	This command is prohibited if a dispersed namespace is attached to the controller and an NSID of FFFFFFFFh is specified for a feature capable of affecting all namespaces attached to that controller.
Format NVM	This command is prohibited if a dispersed namespace exists in the participating NVM subsystem and the scope of the command includes all namespaces that exist in that NVM subsystem (refer to Figure 188).  This command is prohibited if a dispersed namespace is attached to the controller and the scope of the command includes all namespaces attached to that controller (refer to Figure 188).

### 8.1.9.5 ANA Considerations

If more than one participating NVM subsystem contains controllers that provide a host with access to a dispersed namespace, then Asymmetric Namespace Access (ANA) and Asymmetric Namespace Access Reporting (refer to section 8.1.1) should be supported in all participating NVM subsystems. If ANA is used with dispersed namespaces, then globally unique namespace identifiers (refer to section 8.1.9.2) are used to determine when multiple paths are available to the same dispersed namespace. Hosts specify to controllers that their host software (e.g., multipath I/O software) uses globally unique namespace identifiers to determine when multiple paths are available to the dispersed namespace by setting the HDISNS field to 1h in the Host Behavior Support feature, as described in section 8.1.9.3.

ANA Group usage with dispersed namespaces is described in section 8.1.1.2.

### 8.1.9.6 Reservation Considerations

Reservations for a dispersed namespace that is able to be accessed by controllers in multiple participating NVM subsystems are intended to be coordinated between each participating NVM subsystem. To ensure the uniqueness of a host's identity and prevent potential data corruption, each host that sets the Host Dispersed Namespace Support (HDISNS) field to 1h in the Host Behavior Support feature specifies a Host Identifier that is unique across all hosts that connect to any participating NVM subsystem, as described in section 8.1.9.3.

If a host is a reservation registrant on a dispersed namespace, and that host submits a Reservation Report command to that namespace, then for reservations established using controllers that are not contained in the same participating NVM subsystem as the controller processing the Reservation Report command, the Controller ID (CNTLID) field shall be set to FFFDh in the Registered Controller data structure (refer to Figure 583) or Registered Controller Extended data structure (refer to Figure 584). Multiple identical Registered Controller or Registered Controller Extended data structures with the CNTLID field set to FFFDh shall not be returned by a single Reservation Report command; as a result, a Registered Controller or Registered Controller Extended data structure with the CNTLID field set to FFFDh indicates one or more controllers are associated with a host that is a registrant of the dispersed namespace in another participating NVM subsystem. A Registered Controller or Registered Controller Extended data structure with the CNTLID field set to FFFDh is returned for each host (i.e., as identified by the HOSTID field in that data structure) that is a registrant of the dispersed namespace in any other participating NVM subsystem.

The Dispersed Namespace Reservation Support (DISNSRS) bit in a command is set to '1' by hosts that support reservations on dispersed namespaces (i.e., hosts that support receiving a value of FFFDh in the

CNTLID field of a Registered Controller data structure or Registered Controller Extended data structure). A controller that supports dispersed namespaces and supports reservations (i.e., a controller that has the Reservations Support (RESERVS) bit set to '1' in the Optional NVM Command Support (ONCS) field of the Identify Controller data structure (refer to Figure 312)) shall support the DISNSRS bit being set to '1' in each command in the following list:

- Reservation Acquire (refer to section 7.5);
- Reservation Register (refer to section 7.6);
- Reservation Release (refer to section 7.7); and
- Reservation Report (refer to section 7.8).

If the HDISNS field is set to 1h in the Host Behavior Support feature and the DISNSRS bit is cleared to '0' in any of the commands in the preceding list when submitted to a dispersed namespace, then the controller shall abort that command with a status code of Namespace Is Dispersed.

## 8.1.10 Flexible Data Placement

### 8.1.10.1 Flexible Data Placement Overview

The Flexible Data Placement (FDP) capability is an optional capability which allows the host to reduce write amplification by aligning user data usage to physical media. The controller supports log pages which indicate the status of FDP, statistics about the FDP operation, and information the host uses to detect and correct usage patterns that increase write amplification.

The scope of the Flexible Data Placement capability is an Endurance Group.

The host enables or disables Flexible Data Placement by issuing a Set Features command specifying the Flexible Data Placement feature (refer to section 5.1.25.1.20) and the FDP configuration (refer to section 5.1.12.1.28) to be applied to an Endurance Group. The host is required to delete all namespaces associated with the specified Endurance Group before modifying the value of the Flexible Data Placement feature (e.g., transitioning from disabled to enabled).

If a Set Features command specifies:

- the Flexible Data Placement feature;
- an Endurance Group in which one or more namespaces exist; and
- a different value from the current value of that feature for that Endurance Group,

then the controller shall abort the command with a status code of Command Sequence Error.

If a Set Features command is successful and changes the Feature value, then:

- all events in the FDP Events log page (refer to section 5.1.12.1.31) are cleared; and
- all fields in the FDP Statistics log page (refer to section 5.1.12.1.30) are cleared to 0h.

Refer to section 3.2.4 for the logical view of the non-volatile storage capacity for the Endurance Group with Flexible Data Placement enabled.

The Namespace Management command (refer to section 8.1.15) is used to create a namespace within the Endurance Group. All namespaces in an Endurance Group with Flexible Data Placement enabled shall be associated with the NVM Command Set (refer to Figure 311) (i.e., the Command Set Identifier (CSI) field in the Namespace Management command shall be cleared to 00h). The capacity for the user data stored by a write command to a namespace is allocated from the Reclaim Unit referenced by the Reclaim Unit Handle and Reclaim Group specified by that write command. The user data for a namespace is allowed to be in any stored Reclaim Unit within any Reclaim Group within the Endurance Group (refer to Figure 15).

The Namespace Management command specifies a Placement Handle List (refer to the Namespace Management command in the appropriate I/O Command Set specification). Each entry in that list specifies the Reclaim Unit Handle associated with the Placement Handle for that entry. The Placement Handles are numbered from 0h to the number of entries in the list minus 1. Figure 616 shows an example where Namespace A associates Reclaim Unit Handle 1 with Placement Handle 0. A specific Reclaim Unit Handle

is not allowed to be associated with more than one Placement Handle per namespace. There are no other requirements on which Reclaim Unit Handle is associated with which Placement Handle.

Namespaces are allowed to utilize the same (i.e., share) Reclaim Unit Handle but may have restrictions defined in the applicable I/O Command Set specification.

The host is required to enable the Data Placement Directive (refer to section 8.1.8.4) to submit write commands that specify a Placement Identifier (refer to Figure 282 and Figure 283) in the DSPEC field. That Placement Identifier uniquely specifies the Reclaim Unit in which to place the user data (refer to Figure 616). The controller determines the Reclaim Unit by using the Reclaim Unit Handle associated with the Placement Handle that was specified by the host when the namespace was created. For example, if the write command in Figure 616 specifies a Reclaim Group Identifier value of 2h and a Placement Handle value of 1h, then the host is targeting the user data for that write command to be placed into the Reclaim Unit associated with Reclaim Group 2h and Reclaim Unit Handle *NRUH-1*.

A Placement Identifier is invalid if:

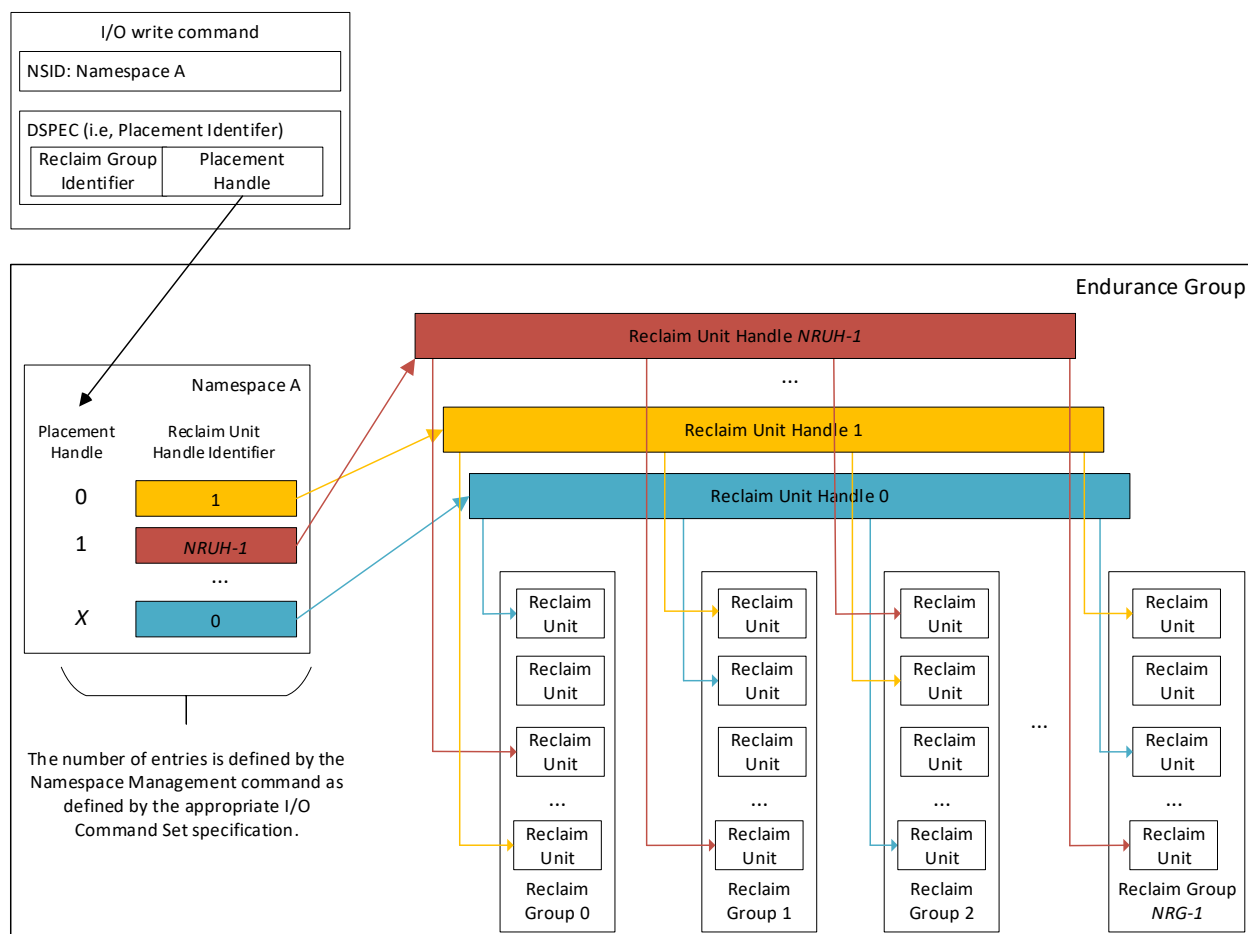
- the Reclaim Group Identifier value is greater than or equal to the number of Reclaim Groups supported by the enabled FDP configuration (refer to the NRG field in Figure 280); or
- the Placement Handle is greater than or equal to the number of Placement Handles supported by the Namespace Management command of the applicable I/O Command Set specification.

If the Placement Identifier in the DSPEC field of a write command is invalid, then the controller shall:

- select the Reclaim Group and Reclaim Unit Handle accessible by the specified namespace to determine the placement of the user data for that write command; and
- generate an Invalid Placement Identifier FDP Event if that event is enabled for the selected Reclaim Unit Handle. To receive the FDP event, the host should enable the Invalid Placement Identifier FDP Event on all Reclaim Unit Handles.

Any write command to a namespace that exists in an Endurance Group with Flexible Data Placement enabled that does not specify the Data Placement Directive uses the Placement Handle value 0h and the controller selects the Reclaim Group for that command. For example, if the write command does not specify the Data Placement Directive, then, as illustrated in Figure 616, Placement Handle value of 0h is associated with Reclaim Unit Handle 1h and the controller may place the user data into the Reclaim Unit associated any Reclaim Group using Reclaim Unit Handle 1h.

Figure 616: Flexible Data Placement Model



A write command that places user data into a Reclaim Unit causes reduction of the remaining capacity of that Reclaim Unit. If that Reclaim Unit is written to capacity, then the controller modifies the Reclaim Unit Handle referencing that Reclaim Unit to reference a different Reclaim Unit within the same Reclaim Group.

A controller may modify a Reclaim Unit Handle to reference a different Reclaim Unit as part of performing a sanitize operation (refer to section 8.1.24).

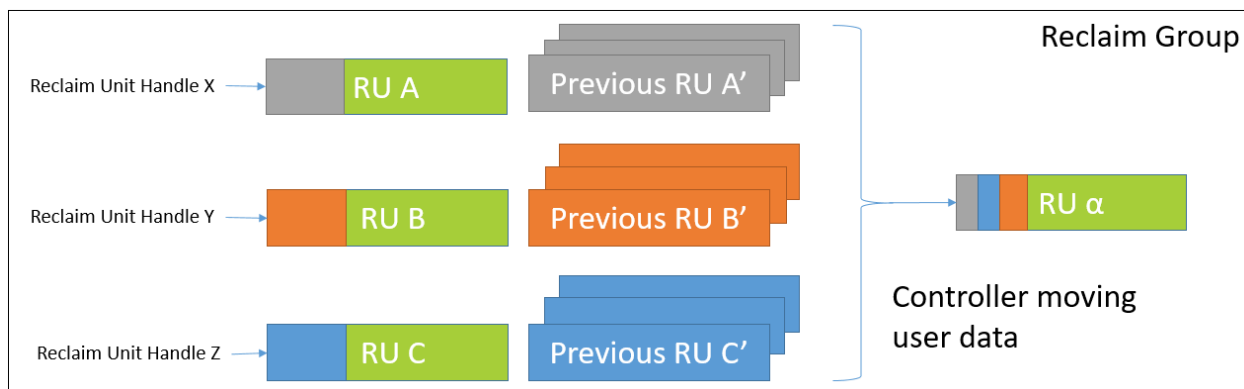
A Reclaim Unit shall only be referenced by one Reclaim Unit Handle. Therefore, a write command that utilizes a Reclaim Unit Handle results in the user data from that write command being initially isolated from the user data from any other write command that utilizes a different Reclaim Unit Handle. Each Reclaim Unit Handle has a type as defined in Figure 281 that identifies the isolation requirements of the user data moved by the controller to a different Reclaim Unit due to vendor specific controller operations (e.g., garbage collection).

A controller is allowed to move user data that was written by the host that utilized a Reclaim Unit Handle with an Initially Isolated type (Figure 281) to a different Reclaim Unit within the same Reclaim Group. In addition, the controller is allowed to move user data written by the host that utilized different Reclaim Unit Handles with an Initially Isolated type to that different Reclaim Unit within the same Reclaim Group as shown in Figure 617.

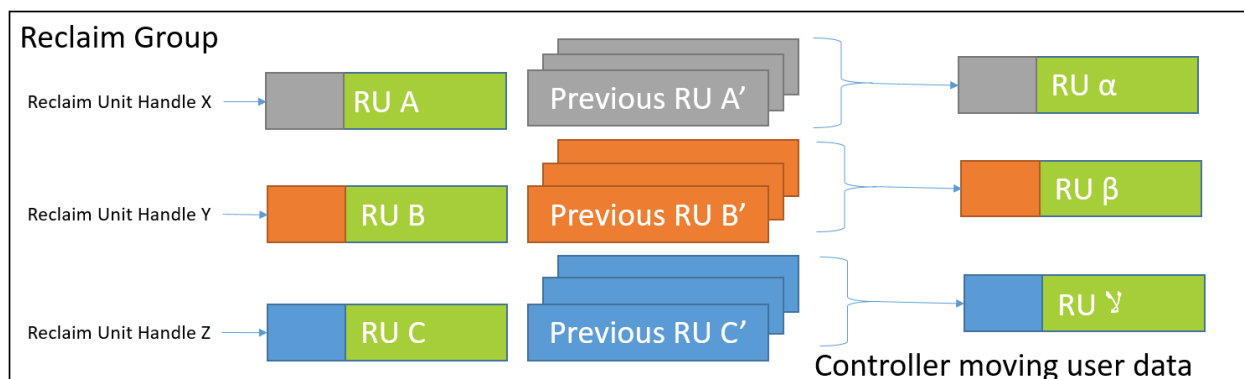
A controller is allowed to move user data that was written by the host that utilized a Reclaim Unit Handle with a Persistently Isolated type (Figure 281) to a different Reclaim Unit within the same Reclaim Group. However, the controller shall only move user data written by the host that utilized the same Reclaim Unit Handles with a Persistently Isolated type to the different Reclaim Unit within the same Reclaim Group as shown in Figure 618.



**Figure 617: Initially Isolated Reclaim Unit Handles**



**Figure 618: Persistently Isolated Reclaim Unit Handle**



The result of a Namespace Management command that creates a namespace within the Endurance Group is that at least one Placement Handle is defined for that namespace (refer to the Namespace Management section in applicable I/O Command Set specification).

For any namespace created in an Endurance Group that has Flexible Data Placement enabled, the host may issue an I/O Management Receive command to obtain a list of Placement Handles and the associated Reclaim Unit Handles that are used by the Data Placement Directive in write commands.

If the Flexible Data Placement capability is supported, then the controller shall support the following:

- a) the Feature Identifiers Supported and Effects log page (refer to section 5.1.12.1.18);
- b) the I/O Management Receive command (refer to section 7.3) and the Reclaim Unit Handle Status Management Operation in that command (refer to Figure 561);
- c) the I/O Management Send command (refer to section 7.4) and the Placement Identifier Update Management Operation in that command (refer to Figure 565);
- d) the Flexible Data Placement feature (refer to section 5.1.25.1.20);
- e) the FDP Configurations log page (refer to section 5.1.12.1.28);
- f) the Reclaim Unit Handle Usage log page (refer to section 5.1.12.1.29);
- g) the FDP Statistics log page (refer to section 5.1.12.1.30);
- h) the Flexible Data Placement Events feature (refer to section 5.1.25.1.21);
- i) the FDP Events log page (refer to section 5.1.12.1.31); and
- j) the Data Placement Directive (refer to section 8.1.8.4).

### 8.1.10.2 Enabling Flexible Data Placement (Informative)

The host prepares an Endurance Group for operation in Flexible Data Placement using the following process:

- 1) Validate that the Flexible Data Placement capability is supported by issuing an Identify command to access the Identify Controller data structure and checking that the Flexible Data Placement Support (FDPS) bit is set to '1'.
- 2) Delete any existing namespaces that exist in the Endurance Group where Flexible Data Placement is to be enabled.
- 3) Issue a Get Log Page command specifying the FDP Configurations log page (refer to section 5.1.12.1.28). Parse the FDP Configurations List in the returned log page to determine the desired FDP configuration.
- 4) Enable Flexible Data Placement utilizing that desired FDP configuration by issuing a Set Features command specifying:
  - a. the Flexible Data Placement feature;
  - b. the Endurance Group Identifier field set to the ENDGID of the Endurance Group in which Flexible Data Placement is to be enabled;
  - c. the FDPE bit set to '1' (i.e., enabling Flexible Data Placement); and
  - d. the Flexible Data Placement Configuration Index field set to the index of the desired FDP configuration from the FDP Configurations List in the FDP Configurations log page.
- 5) Issue Get Features commands for the FDP Events feature (refer to section 5.1.25.1.21) to acquire the list of supported FDP events and the enabled state of each supported FDP event.
- 6) Issue Set Features commands for the FDP Events feature (refer to section 5.1.25.1.21) to specify if FDP events are required for Reclaim Unit Handles.
- 7) Issue an Identify command specifying the Identify Namespace data structure (i.e., CNS 00h as defined in Figure 273) to identify which LBA formats are supported.
- 8) For each namespace to be created in the Endurance Group, issue a Namespace Management command specifying:
  - a. the Select field set to the Create operation (refer to the NVM Express Base Specification);
  - b. the User Data Format; and
  - c. the Placement Handle List used to define the Reclaim Unit Handle associated with each Placement Handle of the namespace (refer to the Namespace Management section of the applicable I/O Command Set specification).
- 9) For each namespace created in an Endurance Group in which the Data Placement Directive is to be utilized by the host by write commands specifying that namespace:
  - a. Issue a Directive Send command specifying:
    - i. the Directive Operation field set to Enable Directive (i.e., 01h);
    - ii. the NSID of that namespace;
    - iii. the DTYPE field in Command Dword 11 set to Identify (i.e., 00h);
    - iv. the DTYPE field in Command Dword 12 set to Data Placement (i.e., 02h); and
    - v. the Enable Directive bit set to '1' (i.e., to enable the Data Placement Directive).
  - b. Issue an Identify command with CNS value 08h (i.e., the I/O Command Set Independent Identify Namespace data structure) to determine if a volatile write cache is present for the namespace.

Following a Controller Level Reset, the host performs the following actions before resuming use of Flexible Data Placement in an Endurance Group:

- 1) Determine the FDP configuration:
  - a. Issue a Get Feature command specifying the Flexible Data Placement feature (i.e., 1Dh) and the ENDGID for the Endurance Group to determine if Flexible Data Placement is enabled and the Flexible Data Placement Configuration Index.
  - b. Issue a Get Log Page command specifying the FDP Configurations log page (i.e., Log Page Identifier 20h).
- 2) Issue Get Features commands for the FDP Events feature (refer to section 5.1.25.1.21) to acquire the list of supported FDP events and the enabled state of each supported FDP event.

- 3) Issue Set Features commands for the FDP Events feature (refer to section 5.1.25.1.21) to specify if supported FDP events are required for Reclaim Unit Handles.
- 4) For each namespace in the Endurance Group that the Data Placement Directive is to be utilized by the host by write commands specifying that namespace:
  - a. Issue an I/O Management Receive command to determine the Placement Handles for that namespace; and
  - b. Issue an Identify command with CNS value 08h (i.e., the I/O Command Set Independent Identify Namespace data structure) to determine if a volatile write cache is present for the namespace.

### 8.1.10.3 Write Commands using Flexible Data Placement (Informative)

Hosts writing user data to a namespace created in an Endurance Group that has Flexible Data Placement enabled have two mechanisms for specifying the placement of the user data with the Endurance Group to the controller.

1. Each namespace has a default Placement Handle. If any write command:
  - a. has the DTYPE field being ignored by the controller (refer to section 8.1.8.4); or
  - b. has the DTYPE field not being ignored by the controller and specifies a DTYPE field value cleared to 00h,

then the Placement Handle value of 0h for the specified namespace is combined with a Reclaim Group selected by the controller to create the Placement Identifier (refer to Figure 282 and Figure 283) utilized for that write command.

2. If any write command has the DTYPE field not being ignored by the controller and specifies a DTYPE field set to Data Placement (i.e., 02h) and specifies a namespace in which the Data Placement Directive is enabled, then the Placement Identifier specified by the DSPEC field specifies the Reclaim Group Identifier and Placement Handle used for the write command.

To reduce write amplification, hosts track the user data written to an entire Reclaim Unit and deallocate that user data by issuing one or more Dataset Management commands specifying the AD bit set to '1' and one or more Range fields specifying the LBA ranges that were written to that Reclaim Unit.

### 8.1.11 Host-Initiated Refresh Operation

The Host-Initiated Refresh operation of the Device Self-test command performs implementation-specific refresh operations that verify the media integrity and ensure access to media (e.g., media in an NVM subsystem that has not been in operation for a long period of time and/or has been subject to extreme environmental conditions). Examples of refresh operations may include read verification, rewrite of underlying media that are exhibiting a high correctable error rate (e.g., to prevent future errors), and other maintenance activities. A Host-Initiated Refresh operation does not change stored user data.

If the Host-Initiated Refresh operation is supported, then the controller shall set the HIRS bit to '1' in the DSTO field in the Identify Controller data structure (refer to Figure 312).

The percentage complete of the Host-Initiated Refresh operation is indicated in the Current Percentage Complete field in the Device Self-test log page (refer to section 5.1.12.1.7).

For Host-Initiated Refresh operation, the NSID field in the Device Self-test command is ignored by the controller and all media in the NVM subsystem is refreshed regardless of whether any portion of the media is or is not part of any namespace.

A Host-Initiated Refresh operation shall be aborted:

- a) if any Controller Level Reset affects the controller on which the device self-test is being performed;
- b) if a Format NVM command is processed by any controller in the NVM subsystem;
- c) if a sanitize operation is started (refer to section 5.1.22); or
- d) if a Device Self-test command with the Self-Test Code field set to Fh is processed by any controller in the NVM subsystem.

A Controller Level Reset on a controller that is not performing the Host-Initiated Refresh operation shall not impact that Host-Initiated Refresh operation.

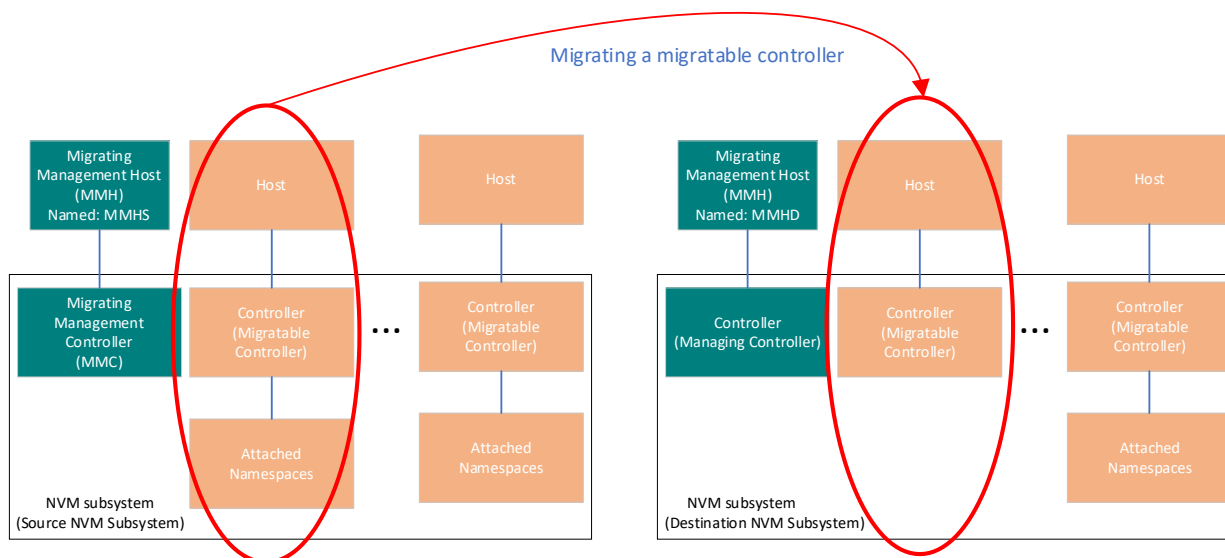
### 8.1.12 Host Managed Live Migration

The migration of a controller, by a host that is using the Host Managed Live Migration capability involves one or more NVM subsystems, multiple hosts, and multiple controllers. In this section, to differentiate between the multiple hosts and controllers, each is given a unique name as illustrated in Figure 619. In an NVM subsystem that supports Host Managed Live Migration, each controller that sets the HMLMS bit to '1' is referred to as a Migration Management Controller (MMC) and each host associated with an MMC is referred to as a Migration Management Host (MMH). In that same NVM subsystem, all other controllers (i.e., those that clear the HMLMS bit to '0') are referred to as Migratable Controllers.

The Host Managed Live Migration capability allows an MMH to use an MMC to migrate a Migratable Controller from a Source NVM Subsystem to a Migratable Controller in a Destination NVM Subsystem. During that migration, a host is actively submitting commands to a Migratable Controller in the Source NVM Subsystem and that controller is processing those commands.

The MMH that is managing the migration of a Migratable Controller from the Source NVM Subsystem is responsible for ensuring that the Migratable Controller in the Destination NVM Subsystem is compatible with controller state of the Migratable Controller from the Source NVM Subsystem. It is also the responsibility of the MMH to transfer the migrating data between the NVM subsystems.

**Figure 619: Host Managed Live Migration Host and Controller Naming**



An MMC is an I/O controller or an Administrative controller that supports the Host Managed Live Migration capability and provides the ability for the MMH to use privileged actions (refer to section 3.10 to:

- suspend the processing of commands on the Migratable Controller in the Source NVM Subsystem being migrated (refer to the Migration Send command in section 5.1.17.1.1);
- obtain the state of the Migratable Controller in the Source NVM Subsystem being migrated (refer to the Migration Receive command in section 5.1.16.1.1);
- set that controller state in the Migratable Controller in the Destination NVM Subsystem (refer to the Migration Send command in section 5.1.17.1.3); and
- resume the operation on the Migratable Controller in the Destination NVM Subsystem (refer to the Migration Send command in section 5.1.17.1.2).

A controller indicates support for the Host Managed Live Migration capability by setting the Host Managed Live Migration Support (HMLMS) bit to '1' in the Optional Admin Command Support (OACS) field in the Identify Controller data structure (refer to Figure 312).

An MMC is not permitted to be migrated and:

- shall support:
    - the Migration Send command with the Management Operations,
      - Suspend (i.e., 0h);
      - Resume (i.e., 1h); and
      - Set Controller State (i.e., 2h);
    - the Migration Receive command with the Management Operations,
      - Controller State (i.e., 1h);
- and
- the CNS value of 21h in the Identify command (i.e., Supported Controller State Formats data structure (refer to section 5.1.13.2.21)).

If the NVM subsystem contains one or more Migration Management Controllers, then:

- each controller (i.e., MMC and Migratable Controller) in that NVM subsystem does not support the Host Memory Buffer (refer to the Host Memory Buffer Preferred Size (HMPRE) field in the Identify Controller data structure in Figure 312); and
- each Migratable Controller in that NVM subsystem is allowed to be migrated and shall not support:
  - the Migration Send command (refer to section 5.1.17);
  - the Migration Receive command (refer to section 5.1.16);
  - the CNS value of 21h in the Identify command (i.e., returning Supported Controller State Formats data structure (refer to section 5.1.13.2.21)); and
  - the Controller Data Queue command with the Create Queue management operation specifying the User Data Migration Queue (refer to section 5.1.4.1.1).

A Controller Level Reset on an MMC shall cause that MMC to:

- delete all User Data Migration Queues on that MMC; and
- stop tracking all host memory changes on that MMC;

A Controller Level Reset on a Migratable Controller shall cause that Migratable Controller to:

- remove a suspended state if that Migratable Controller is suspended by a Migration Send command specifying the Suspend management operation (refer to section 5.1.17.1.1); and
- retain any persistent controller state that has been committed to that Migratable Controller from a Migration Send command specifying the Set Controller State management operation (refer to section 5.1.17.1.3).

During the migration of a Migratable Controller in the Source NVM Subsystem while that Migratable Controller is processing commands, it may be necessary for the MMHS to obtain the user data modifications to namespaces attached to that Migratable Controller and modifications made by that Migratable Controller to host memory in the host. The Track Send command (refer to section 5.1.27) and the Track Receive command (refer to section 5.1.26) allows the host to use privileged actions (refer to section 3.10) to:

- determine user data in namespaces attached to a controller that has changed; and
- determine host memory that has changed due to commands processed by a controller.

A controller indicates support for tracking changes to user data in namespaces attached to a controller by setting the Track User Data Changes Support (TUDCS) bit to '1' in the Tracking Attributes (TRATTR) field in the Identify Controller data structure (refer to Figure 312).

A controller indicates support for tracking changes to host memory (refer to section 1.5.46) due to the processing of commands by a controller by setting the Track Host Memory Changes Support (THMCS) bit to '1' in the TRATTR field.

### 8.1.12.1 Process for Migrating a Controller

The controller This section provides an example sequence of steps to allow the MMHS to migrate a Migratable Controller in the Source NVM Subsystem and the attached namespaces to a Migratable Controller in the Destination NVM Subsystem (refer to Figure 619). Additionally, the steps include obtaining the host memory modifications made to the host due to the Migratable Controller in the Source NVM Subsystem processing commands during the migration.

In this example the MMC in the Source NVM Subsystem has:

- the Track User Data Changes Support (TUDCS) bit set to '1' in the Tracking Attributes (TRATTR) field in the Identify Controller data structure (refer to Figure 312); and
- the Track Host Memory Changes Support (THMCS) bit set to '1' in the Tracking Attributes (TRATTR) field in the Identify Controller data structure.

If attached namespaces are not required to be migrated by the MMHS, then those steps may be ignored. If host memory modifications are not required to be obtained by the MMHS, then those steps may be ignored.

1. The MMHS copies (i.e., migrates) the user data in namespaces attached to the Migratable Controller in the Source NVM Subsystem while the host is submitting commands and the Migratable Controller is processing those submitted commands.
  - a. The MMHS creates a User Data Migration Queue in the MMC in the Source NVM Subsystem by submitting a Controller Data Queue command to that MMC (refer to section 5.1.4) specifying:
    - the Select field set to the Create Queue management operation (i.e., 1h);
    - the Queue Type field set to User Data Migration Queue; and
    - the Controller Identifier field (refer to Figure 166) set to the controller identifier for the Migratable Controller in the Source NVM Subsystem.
  - b. If the Controller Data Queue command is successful, the completion queue entry contains the Controller Data Queue Identifier for the created User Data Migration Queue. The MMHS causes the MMC in the Source NVM Subsystem to post user data modifications to namespaces attached to the Migratable Controller by submitting a Track Send command to that MMC specifying:
    - the Select field set to Log User Data Changes management operation (i.e., 0h);
    - the Logging Action bit set to '1' (i.e., start logging); and
    - the Controller Data Queue Identifier field set to the Controller Data Queue Identifier from the completion queue entry for the Controller Data Queue command.

It is the responsibility of the MMHS to make sure that the User Data Migration Queue is sized properly and there may be more restrictions on that MMHS as specified by the appropriate I/O command set (e.g., the NVM Command Set requires the MMHS to not allow the User Data Migration Queue to become full). Refer to section 8.1.6 for a description of the User Data Migration Queue (i.e., a Controller Data Queue).
  - c. If the Track Send command is successful, then the MMHS starts copying the namespaces attached to the Migratable Controller in the Source NVM Subsystem to the Destination NVM Subsystem. Refer to the applicable I/O Command Set specifications for capabilities that may reduce the amount of user data that is required to be copied (e.g., the Get LBA Status command in the NVM Command Set Specification).
2. The MMHS causes the MMC in the Source NVM Subsystem to start tracking the modifications to host memory in the host due to the Migratable Controller processing submitted commands.

- a. The MMHS submits a Track Send command to the MMC in the Source NVM Subsystem (refer to section 5.1.27) specifying:
  - the Select field set to the Track Memory Changes management operation (i.e., 0h);
  - the Tracking Action (TACT) bit set to '1';
  - the Controller Identifier field specifying the identifier for the Migratable Controller in the Source NVM Subsystem being migrated; and
  - a Track Memory Changes data structure specifying the host memory to be tracked as described by the set of host memory ranges.
- b. If the Track Send command is successful, then the MMHS may query the host memory modifications that have resulted from the Migratable Controller processing commands by submitting a Track Receive command to the MMC in the Source NVM Subsystem specifying:
  - the Select field set to the Track Memory Changes management operation (i.e., 0h); and
  - the Controller Identifier field specifying the identifier for the Migratable Controller in the Source NVM Subsystem being migrated.
3. Once the MMHS has copied the namespaces attached to the Migratable Controller in the Source NVM Subsystem, that MMHS determines a time to suspend that Migratable Controller. During this period, that MMHS is migrating the user data in the namespaces attached to that Migratable Controller that are identified in the posted entries in the User Data Migration Queue. That MMHS may handle the modified host memory in the host as reported by the Track Receive command.
4. The MMHS suspends the Migratable Controller in the Source NVM Subsystem by submitting a Migration Send command to the MMC in the Source NVM Subsystem specifying:
  - the Select field set to the Suspend management operation (i.e., 0h);
  - the Suspend Type field set to Suspend;
  - Delete User Data Migration Queue bit set to '1', if the MMHS desires that the MMC delete the User Data Migration Queue being used to log user data changes of the Migratable Controller being migrated; and
  - the Controller Identifier field (refer to Figure 166) set to the controller identifier for the Migratable Controller to be suspended (i.e., the Migratable Controller being migrated).

Prior to completing that Migration Send command:

- the specified Migratable Controller being suspended:
    - stops fetching commands (i.e.; from the Admin queue and I/O queues); and
    - completes all previously fetched commands;and
  - the MMC processing that Migration Send command:
    - if user data is being logged into a User Data Migration Queue that is associated with the Migratable Controller being migrated, an entry is placed into the User Data Migration Queue that indicates the user data logging has suspended, and then deletes the User Data Migration Queue, if the Delete User Data Migration Queue bit is set to '1'.
5. Since the Migratable Controller in the Source NVM Subsystem is suspended, the MMHS migrates the user data in the namespaces attached to the Migratable Controller that are identified in the posted entries in the User Data Migration Queue. Refer to the appropriate I/O command set specification to determine how the entries identify when entries have completed being posted. That MMHS may handle the modified host memory in the host as reported by the Track Receive command.
  6. The MMHS is able to obtain the controller state of the Migratable Controller in the Source NVM Subsystem by submitting one or more Migration Receive commands to the MMC in the Source NVM Subsystem specifying:

- a. the Select field set to the Get Controller State management operation (i.e., 0h);
  - b. the Sequence Indicator field set to the proper sequence value;
  - c. the Controller Identifier field set to the controller identifier for the Migratable Controller in the Source NVM Subsystem;
  - d. the Controller State Version Index field set to an index into the NVMe Controller State Version list in the Supported Controller State Formats data structure for MMC in the Source NVM Subsystem (refer to Figure 329) if the NVMe defined controller state is required; and
  - e. the Controller State UUID Index field set to an index in the Vendor Specific Controller State UUID Supported list in the Supported Controller State Formats Data Structure for MMC in the Source NVM Subsystem (refer to Figure 329), if vendor specific controller state is required.
7. After the MMHS has transferred the obtained controller state information to the MMHD, the MMHD sets that controller state into a Migratable Controller by submitting a sequence of one or more Migration Send commands to the MMC in the Destination NVM Subsystem specifying:
    - the Select field set to Set Controller State management operation (i.e., 3h);
    - the Sequence Indicator field set to the proper sequence value;
    - the Controller Identifier field set to the controller identifier for the Migratable Controller in the Destination NVM Subsystem;
    - the Controller State Version Index field set to an index into the NVMe Controller State Version list in the Supported Controller State Formats data structure for MMC in the Destination NVM Subsystem (refer to Figure 329) if value of the NVMe Controller State Size (NVMECSS) field is non-zero (refer to Figure 358); and
    - the Controller State UUID Index field set to an index in the Vendor Specific Controller State UUID Supported list in the Supported Controller State Formats Data Structure for MMC in the Source NVM Subsystem (refer to Figure 329), if the value of the Vendor Specific Size (VSS) field is non-zero (refer to Figure 358).
  8. The MMHD resumes operation on the Migratable Controller in the Destination NVM Subsystem (i.e., the migrated controller) by submitting a Migration Send command to the MMC in the Destination NVM Subsystem specifying:
    - the Select field set to Resume management operation (i.e., 2h);
    - the Controller Identifier field (refer to Figure 166) set to the controller identifier for the Migratable Controller to be resumed.

### 8.1.13 Key Per I/O

The Key Per I/O capability provides a mechanism to use encryption keys that have been injected into an NVM subsystem by a host. The mechanism to perform activation of the Key Per I/O capability, encryption key injection, management, and association to encryption key tag is outside the scope of this specification. One mechanism is defined in the TCG Storage Security Subsystem Class: Key Per I/O Specification using the NVMe Admin commands Security Send and Security Receive. If the TCG mechanism is used, any additional modifications to the NVM subsystem as a result of activation of the Key Per I/O Security Provider are defined in the TCG Storage Interface Interactions Specification (SIIS).

Encryption keys injected into the NVM subsystem may be referenced by I/O commands through the use of the encryption key tag (refer to the KEYTAG field in Figure 620) associated with the encryption keys. If an I/O command CETYPE field is set to the KPIOTAG value, then the CEV field (refer to Figure 620) of that I/O command specifies the encryption key tag associated with the encryption key to be used to encrypt or decrypt the data in that I/O command. The association of an encryption key tag to a specific encryption key is outside the scope of this specification. One association mechanism is defined in the TCG Storage Security Subsystem Class: Key Per I/O Specification.

The Key Per I/O Scope bit (refer to Figure 312) indicates if the Key Per I/O capability:

- applies to all read and write commands in all namespaces within the NVM subsystem; or
- independently applies to read and write commands in each namespace within the NVM subsystem.



The Key Per I/O capability does not have any effect on host accesses to RPMBs and Boot Partitions as these features are not addressed through I/O commands that specify a namespace.

A controller that supports the Key Per I/O capability shall set the KPIOS bit to '1' in the Identify Controller data structure (refer to Figure 312).

A namespace that supports the Key Per I/O capability shall set the KPIOSNS bit to '1' in the I/O Command Set Independent Identify Namespace data structure (refer to Figure 319).

The Key Per I/O capability uses the Command Extension Type (CETYPE) and Command Extension Value (CEV) fields in all read and write commands. Definition of the CETYPE fields are shown in Figure 620.

**Figure 620: CETYPE Definition**

Value	Definition	CEV Field Definition
0h	Reserved	
1h	<b>Key Per I/O Tag (KPIOTAG):</b> This command is using the Key Per I/O capability.	<p><b>Key Tag (KEYTAG):</b> Specifies a namespace-specific 16-bit encryption key tag that identifies the encryption key used to encrypt or decrypt the data of the command.</p> <p>The same Key Tag value on different namespaces may or may not identify the same encryption key.</p> <p>Refer to the Maximum Key Tag field in the I/O Command Set Independent Identify Namespace data structure (refer to Figure 319) for the supported values.</p>
2h to Eh	Reserved	
Fh	Vendor Specific	

If:

- the Key Per I/O capability is enabled in a namespace (i.e., the KPIOENS bit set to '1');
- an I/O command supports the CETYPE field; and
- the CETYPE field in that I/O command is set to a value that is reserved,

then the controller shall abort that command with a status code of Invalid Field in Command.

#### 8.1.14 Management Addresses

Various entities on a network are able to request a management agent to perform management operations on NVM subsystems. A controller in an NVM subsystem use the Management Addresses capability to indicate the network addresses of those management agents. When an NVM subsystem is provisioned in a storage system, the management addresses are established in the controller.

Management agents are able to be located in various networked entities, including:

- NVM subsystems;
- Fabric interface managers;
- Embedded management controllers; and
- Host software.

Each management address is represented as a uniform resource indicator (URI; refer to RFC 3986).

The address of a management agent contained in an NVM subsystem (e.g., an Ethernet-attached SSD) is indicated in the Management Address List log page (refer to section 5.1.12.1.24), in a Management Address Descriptor indicating a Management Address Type of 1h. The method by which the address is determined is outside the scope of this specification.

The address of a management agent contained in a fabric interface manager is indicated in the Management Address List log page (refer to section 5.1.12.1.24), in a Management Address Descriptor indicating a Management Address Type of 2h. The method by which the address is determined is outside the scope of this specification.

The address of a management agent contained in an embedded management controller (e.g., a BMC) is set and retrieved using the Embedded Management Controller Address feature (refer to section 5.1.25.1.24). The Embedded Management Controller Address feature is intended to be set only by the storage system containing the embedded management controller. The embedded management controller is able to ensure this by issuing a Lockdown command (refer to section 5.1.15) with the OFI field specifying this Feature, and the PRHBT bit set to '1' and the IFC field set to Admin Submission Queue (i.e., 00b).

The address of a management agent contained in host software is set and retrieved using the Host Management Controller Address feature (refer to section 5.1.25.1.25). The Host Management Agent Address feature is intended to be set only by host software. Host software is able to ensure this by issuing a Lockdown command (refer to section 5.1.15) with the OFI field specifying this Feature, and the PRHBT bit set to '1' and the IFC field set to Out-of-band on a Management Endpoint (i.e., 10b).

### 8.1.15 Namespace Management

The Namespace Management capability consists of the Namespace Management command (refer to section 5.1.21) and the Namespace Attachment command (refer to section 5.1.20). The Namespace Management command is used to create a namespace or delete a namespace. The Namespace Attachment command is used to attach and detach controllers from a namespace. The Namespace Management capability is intended for use during manufacturing or by a system administrator.

In addition to the Namespace Management capability, there are other events and capabilities that are able to affect the number of namespaces that exist within an NVM subsystem (e.g., the Capacity Management capability as described in section 8.1.4).

If the Namespace Management capability is supported, then the controller:

- a) shall support the Namespace Management command and the Namespace Attachment command;
- b) shall set the NMS bit to '1' in the OACS field (refer to Figure 312);
- c) shall support the Attached Namespace Attribute Changed asynchronous event (refer to Figure 151 and section 5.1.25.1.5);
- d) should support the Allocated Namespace Attribute Changed asynchronous event (refer to Figure 151 and section 5.1.25.1.5); and
- e) may support Namespace Granularity (refer to the NVM Command Set Specification).

A controller may support the Namespace Attachment command without supporting the Namespace Management command. Such a controller:

- a) shall not set the Namespace Management Supported (NMS) bit to '1' in the OACS field (refer to Figure 312); and
- b) should support the Namespace Attribute Change asynchronous event.

If a namespace is detached from a controller, then the NSID that referred to that namespace becomes an inactive NSID (refer to section 3.2.1.4) on that controller. If a namespace is deleted from the NVM subsystem, then the NSID that referred to that namespace becomes an unallocated NSID (refer to section 3.2.1.3) in the NVM subsystem. Previously submitted but uncompleted or subsequently submitted commands to the affected NSID are handled by the controller as if they were issued to an inactive NSID (refer to Figure 92).

The size of a namespace that contains formatted storage is based on the size requested in a create operation, the format of the namespace, and any characteristics (e.g., endurance). The controller determines the NVM capacity allocated for that namespace. Namespaces may be created with different usage characteristics (e.g., endurance) that utilize differing amounts of NVM capacity. Namespace characteristics and the mapping of these characteristics to NVM capacity usage are outside the scope of this specification.

Reporting of capacity information for the NVM subsystem, Domain, Endurance Group, and NVM Set are described in section 3.8. For each namespace that contains formatted storage, the NVM Set and the Endurance Group that contain the namespace are reported in the Identify Namespace data structure. The NVM Set to be used for a namespace that contains formatted storage is based on the value in the NVM

Set Identifier field in a create operation. If the NVM Set Identifier field is cleared to 0h in a create operation for a namespace that contains formatted storage, then the controller shall choose the NVM Set from which to allocate capacity to create that namespace.

If the NVM Set Identifier field and the Endurance Group Identifier field are both cleared to 0h in a create operation for a namespace that contains formatted storage, then the controller shall choose the Endurance Group and the NVM Set from which to allocate capacity to create that namespace.

If the NVM Set Identifier field is cleared to 0h and the Endurance Group Identifier field is set to a non-zero value in a create operation for a namespace that contains formatted storage, then the controller shall choose the NVM Set in the specified Endurance Group from which to allocate capacity to create that namespace.

If the NVM Set Identifier field is set to a non-zero value and the Endurance Group Identifier field is cleared to 0h in a create operation for a namespace that contains formatted storage, then the controller shall abort the command with a status code of Invalid Field in Command.

If the NVM Set Identifier field and the Endurance Group Identifier field are both set to non-zero values in a create operation for a namespace that contains formatted storage and the specified NVM Set exists in the specified Endurance Group, then the controller shall allocate capacity for that created namespace from the specified NVM Set.

If the NVM Set Identifier field and the Endurance Group Identifier field are both set to non-zero values in a create operation for a namespace that contains formatted storage and the specified NVM Set does not exist in the specified Endurance Group, then the controller shall abort the command with a status code of Invalid Field in Command.

For each namespace that contains formatted storage, the NVM capacity used for that namespace is reported in the Identify Namespace data structure (refer to the applicable I/O Command Set specification). The controller may allocate NVM capacity in units such that the requested size for a namespace that contains formatted storage may be rounded up to the next unit boundary. The units in which NVM capacity is allocated are reported in the Namespace Granularity List (refer to the NVM Command Set Specification), if supported. For example, when using the NVM Command Set, if host software requests a namespace of 32 logical blocks with a logical block size of 4 KiB for a total size of 128 KiB and the allocation unit for the implementation is 1 MiB, then the NVM capacity consumed may be rounded up to 1 MiB. The NVM capacity fields may not correspond to the logical block size multiplied by the total number of logical blocks.

The method of allocating ANA Group identifiers is outside the scope of this specification. If the ANA Group Identifier (refer to Figure 319 and the Identify Namespace data structure in the NVM Command Set Specification) is cleared to 0h, then the controller shall determine the ANAGRPID that is assigned to that namespace.

### 8.1.15.1 Namespace Management Examples

The following is an example of how a host creates a namespace:

1. the host requests the Identify Namespace data structure that specifies common namespace capabilities (i.e., using an Identify command with the NSID field set to FFFFFFFFh and the CNS field cleared to 0h);
2. if the controller supports reporting of I/O Command Set specific Namespace Management content (refer to the Namespace Management section in the applicable I/O Command Set specification), host software optionally requests that information (e.g., Namespace Granularity).
3. the host determines available resources (e.g., capacity for a namespace that contains formatted storage) (refer to section 3.8);
4. the host creates the data structure defined in Figure 369 (e.g., taking into account the common namespace capabilities, available capacity);
5. the host issues the Namespace Management command specifying the Create operation and the data structure. On successful completion of the command, the Namespace Identifier of the new namespace is returned in Dword 0 of the completion queue entry. At this point, the new namespace is not attached to any controller; and

6. if Allocated Namespace Attribute Notices are enabled on the controller, then the host requests the Identify Namespace data structures (refer to section 1.5.49) for the new namespace to determine all attributes of the namespace upon receiving an Allocated Namespace Attribute Changed asynchronous event from that controller.

The following is an example of how a host attaches a namespace:

1. the host issues the Namespace Attachment command specifying the Controller Attach operation to attach the specified namespace to one or more controllers;
2. the host receives an Attached Namespace Attribute Changed asynchronous event from each controller that has a namespace newly attached to that controller if Attached Namespace Attribute Notices are enabled on that controller; and
3. the host receives an Allocated Namespace Attribute Changed asynchronous event from each controller that has Allocated Namespace Attribute Notices enabled.

The following is an example of how a host detaches a namespace:

1. the host issues the Namespace Attachment command specifying the Controller Detach operation to detach the specified namespace from one or more controllers;
2. the host receives an Attached Namespace Attribute Changed asynchronous event from each controller that has a namespace newly detached from that controller if Attached Namespace Attribute Notices are enabled on that controller; and
3. the host receives an Allocated Namespace Attribute Changed asynchronous event from each controller that has Allocated Namespace Attribute Notices enabled.

The following is an example of how a host deletes a namespace:

1. the host should detach the namespace from all controllers;
2. the host issues the Namespace Management command specifying the Delete operation for the specified namespace. On successful completion of the command, the namespace has been deleted;
3. if the namespace was attached to any controller(s) when deleted:
  - a. the host receives an Attached Namespace Attribute Changed asynchronous event from each of those controllers that has Attached Namespace Attribute Notices enabled as described in section 8.1.15.2; and
  - b. the host receives an Allocated Namespace Attribute Changed asynchronous event from each of those controllers that has Allocated Namespace Attribute Notices enabled as described in section 8.1.15.2;

and

4. if the namespace was not attached to any controller when deleted, the host receives an Allocated Namespace Attribute Changed asynchronous event from each controller that has Allocated Namespace Attribute Notices enabled as described in section 8.1.15.2.

### 8.1.15.2 Namespace Deletion Asynchronous Event Reporting

If a delete operation is requested for a namespace:

- via a command (e.g., a Namespace Management command or Capacity Management command) received on the Admin Submission Queue (e.g., not via a Management Endpoint, refer to the NVM Express Management Interface Specification), then:
  - each controller, to which the namespace is attached, that has Attached Namespace Attribute Notices enabled (refer to Figure 391) other than the controller processing that delete operation, shall issue an Attached Namespace Attribute Changed asynchronous event as part of the delete operation to indicate a namespace change; and
  - each controller that has Allocated Namespace Attribute Notices enabled (refer to Figure 391) other than the controller processing that delete operation, shall issue an Allocated Namespace Attribute Changed asynchronous event as part of the delete operation to indicate a namespace change;

or

- via any other method (e.g., a method outside the scope of this specification or via a Management Endpoint, refer to the NVM Express Management Interface Specification), then:
  - each controller, to which the namespace is attached, that has Attached Namespace Attribute Notices enabled (refer to Figure 391) shall issue an Attached Namespace Attribute Changed asynchronous event as part of the delete operation to indicate a namespace change; and
  - each controller that has Allocated Namespace Attribute Notices enabled (refer to Figure 391) shall issue an Allocated Namespace Attribute Changed asynchronous event as part of the delete operation to indicate a namespace change.

### 8.1.15.3 Namespace Management Considerations for Exported NVM Subsystems (informative)

A host is able to determine whether an Exported NVM Subsystem supports the Namespace Attachment command by reading the Commands Supported and Effects log page (refer to section 5.1.12.1.6). The Namespace Attachment command is used to attach namespaces to controllers and detach namespaces from controllers. The Namespace Management command is not supported by controllers in an Exported NVM subsystem.

If an Exported Namespace is detached from a controller in the Exported NVM Subsystem, then the NSID that referred to that namespace becomes an inactive NSID (refer to section 3.2.1.4) on that controller. If an Underlying Namespace is disassociated from the Exported NVM Subsystem (refer to section 5.3.7.1.2), then the NSID of the Exported Namespace that referred to that namespace becomes an unallocated NSID (refer to section 3.2.1.3) and is not available to any controller in the Exported NVM Subsystem. Previously submitted but uncompleted or subsequently submitted commands to the namespace that is:

- detached from a controller; or
- disassociated from the Exported NVM Subsystem,

are handled by the controller as if they were issued to an inactive NSID (refer to section 3.2.1.4 and Figure 92). Refer to section 8.1.15 for Host actions to attach or detach a specified Namespace.

If an Exported NVM Subsystem Exports an Underlying Namespace that becomes unavailable (e.g., detached or deleted) or is affected by a Capacity Management command (e.g., Endurance Group deletion, NVM Set deletion (refer to section 8.1.4)) then:

- Previously submitted but uncompleted or subsequently submitted commands to the Exported Namespace are handled by the controller as if they were issued to an inactive NSID (refer to section 3.2.1.4 and Figure 92); and
- If Namespace Attribute Notices are enabled controllers affected by Underlying Namespace changes report a Namespace Attribute Changed asynchronous event to the host as described in Figure 151.

### 8.1.16 Namespace Write Protection

Namespace Write Protection is an optional configurable controller capability that enables the host to control the write protection state of a namespace or to determine the write protection state of a namespace. Support for this capability is reported in the Namespace Write Protection Capabilities (NWPC) field in the Identify Controller data structure (refer to Figure 312).

Figure 621 defines the write protection states that may be supported for a namespace. All states persist across power cycles and Controller Level Resets (refer to section 3.7.2) except Write Protect Until Power Cycle state, which transitions to the No Write Protect state on the occurrence of a power cycle.

**Figure 621: Namespace Write Protection State Definitions**

M/O	State	Definition	Persistent Across	
			Power Cycles	Controller Level Resets
M	No Write Protect	The namespace is not write protected.	Yes	Yes

**Figure 621: Namespace Write Protection State Definitions**

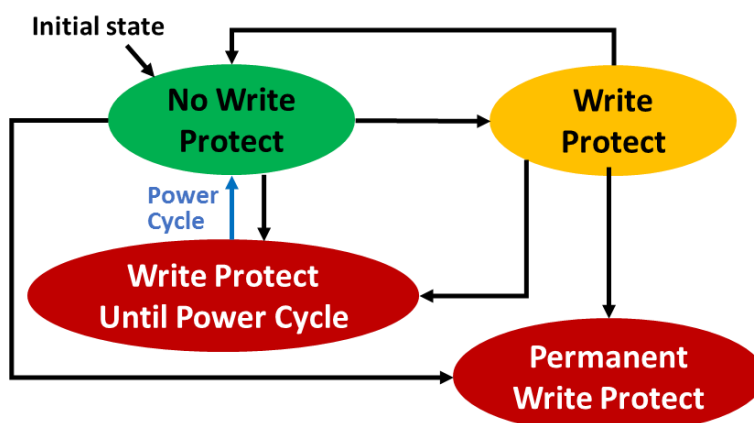
M/O	State	Definition	Persistent Across	
			Power Cycles	Controller Level Resets
M	Write Protect	The namespace is write protected.	Yes	Yes
O	Write Protect Until Power Cycle	The namespace is write protected until the next power cycle.	No	Yes
O	Permanent Write Protect	The namespace is permanently write protected.	Yes	Yes

Notes:  
M – If the Namespace Write Protection capability is supported, then support of this state is mandatory.  
O – If the Namespace Write Protection capability is supported, then support of this state is optional.

The Write Protect Until Power Cycle state shall not be used in multi-domain NVM subsystems because clearing that state requires simultaneous power cycle of the namespace and all controllers to which that namespace is attached. The result of a command that attempts to use that state in a multi-domain NVM subsystem is specified in section 5.1.25.1.31.

Figure 622 defines the transition between write protection states. All state transitions are based on Set Features commands unless otherwise specified. The initial state of a namespace at the time of its creation is the No Write Protect state.

**Figure 622: Namespace Write Protection State Machine Model**



The Write Protect Until Power Cycle and Permanent Write Protect states are subject to the controls defined in the Write Protection Control field (refer to Figure 633), which determines whether the controller processes or aborts Set Features commands which cause a transition into either of these two states (refer to section 8.1.21).

The results of using Namespace Write Protection in combination with an external write protection system (e.g., TCG Storage Interface Interactions Specification) are outside the scope of this specification.

**8.1.16.1 Namespace Write Protection – Theory of Operation**

If Namespace Write Protection is supported by the controller, then the controller shall:

- indicate the level of support for Namespace Write Protection capabilities in the Namespace Write Protection Capabilities (NWPC) field in the Identify Controller data structure by:
  - setting the No Write Protect and Write Protect Support bit to ‘1’ in the NWPC field;
  - setting the Write Protect Until Power Cycle Support bit to ‘1’ in the NWPC field, if the Write Protect Until Power Cycle state is supported; and

- setting the Permanent Write Protect Support bit to ‘1’ in the NWPC field, if the Permanent Write Protect state is supported;

and

- support the Namespace Write Protection Config feature (refer to section 5.1.25.1.31).

If the controller supports the Write Protect Until Power Cycle state or the Permanent Write Protect state, then the controller shall support the Write Protection Control field in the RPMB Device Configuration Block data structure (refer to section 8.1.21).

The controller shall not set the All Media Read-Only bit to ‘1’ in the Critical Warning field (refer to Figure 206) if the read-only condition on the media is a result of a change in the namespace write protection state as defined by the Namespace Write Protection State Machine (refer to Figure 622), or due to any autonomous namespace write protection state transitions (e.g., power cycle). Host software may check the current namespace write protection state of a namespace using the Get Features command with the Namespace Write Protection Config Feature Identifier.

If any controller in the NVM subsystem supports Namespace Write Protection, then the write protection state of a namespace shall be enforced by any controller to which that namespace is attached.

### 8.1.16.2 Namespace Write Protection – Command Interactions

Unless otherwise noted, the commands listed in Figure 623 are processed normally when specifying an NSID for a namespace that is write protected.

**Figure 623: Commands Allowed when Specifying a Write Protected NSID**

Admin Command Set	NVM Command Set
Device Self-test	Compare
Directive Send <sup>1</sup>	Dataset Management <sup>1</sup>
Directive Receive <sup>3</sup>	Read
Get Features	Reservation Register
Get Log Page	Reservation Report
Identify	Reservation Acquire
Namespace Attachment	Reservation Release
Security Receive <sup>1</sup>	Vendor Specific <sup>1</sup>
Security Send <sup>1</sup>	Flush <sup>2</sup>
Set Features <sup>1</sup>	Verify
Vendor Specific <sup>1</sup>	

Notes:

1. The controller shall fail commands if the specified action attempts to modify the non-volatile storage medium of the specified namespace.
2. A Flush command shall complete successfully with no effect. All volatile write cache data and metadata associated with the specified namespace is written to non-volatile storage medium as part of transitioning to the write protected state (refer to section 5.1.25.1.31).
3. A Directive Receive command which attempts to allocate streams resources shall be aborted with a status code of Namespace is Write Protected.

Commands not listed in Figure 623, and which meet the following conditions, shall be aborted with a status code of Namespace Is Write Protected (refer to Figure 102):

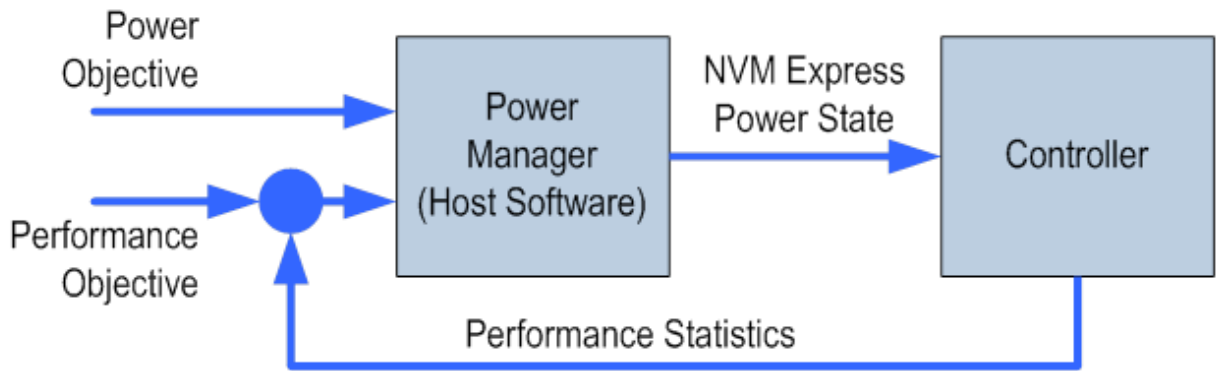
- a) Commands that specify an NSID for a namespace that is write protected;
  - b) Commands that specify an NSID for a namespace that is not write protected and the execution of which would modify another namespace that is write protected (e.g., a Format NVM command);
- and

- c) Commands that do not specify an NSID, and the execution of which would modify a namespace that is write protected (e.g., Sanitize command).

**8.1.17 Power Management**

The power management capability allows the host to manage NVM subsystem power statically or dynamically. Static power management consists of the host determining the maximum power that may be allocated to an NVM subsystem and setting the NVM Express power state to one that consumes this amount of power or less. Dynamic power management is illustrated in Figure 624 and consists of the host modifying the NVM Express power state to best satisfy changing power and performance objectives. This power management mechanism is meant to complement and not replace autonomous power management or thermal management performed by a controller.

**Figure 624: Dynamic Power Management**



The number of power states implemented by a controller is returned in the Number of Power States Supported (NPSS) field in the Identify Controller data structure. If the controller supports this feature, at least one power state shall be defined and optionally, up to a total of 32 power states may be supported. Power states shall be contiguously numbered starting with zero such that each subsequent power state consumes less than or equal to the maximum power consumed in the previous state. Thus, power state zero indicates the maximum power that the NVM subsystem is capable of consuming.

Associated with each power state is a Power State Descriptor in the Identify Controller data structure (refer to Figure 313). The descriptors for all implemented power states may be viewed as forming a table as shown in the example in Figure 625 for a controller with seven implemented power states. Note that Figure 625 is illustrative and does not include all fields in the power state descriptor. The Maximum Power (MP) field indicates the sustained maximum power that may be consumed in that state, where power measurement methods are outside the scope of this specification. The controller may employ autonomous power management techniques to reduce power consumption below this level, but under no circumstances is power allowed to exceed this level except for non-operational power states as described in section 8.1.17.1.

**Figure 625: Example Power State Descriptor Table**

Power State	Maximum Power (MP)	Entry Latency (ENLAT)	Exit Latency (EXLAT)	Relative Read Throughput (RRT)	Relative Read Latency (RRL)	Relative Write Throughput (RWT)	Relative Write Latency (RWL)
0	25 W	5 μs	5 μs	0	0	0	0
1	18 W	5 μs	7 μs	0	0	1	0
2	18 W	5 μs	8 μs	1	0	0	0
3	15 W	20 μs	15 μs	2	0	2	0
4	10 W	20 μs	30 μs	1	1	3	0
5	8 W	50 μs	50 μs	2	2	4	0



**Figure 625: Example Power State Descriptor Table**

Power State	Maximum Power (MP)	Entry Latency (ENLAT)	Exit Latency (EXLAT)	Relative Read Throughput (RRT)	Relative Read Latency (RRL)	Relative Write Throughput (RWT)	Relative Write Latency (RWL)
6	5 W	20 $\mu$ s	5,000 $\mu$ s	4	3	5	1

The Idle Power (IDLP) field indicates the typical power consumed by the NVM subsystem over 30 seconds in the power state when idle (e.g., there are no pending commands, property accesses, background processes, nor device self-test operations). The measurement starts after the NVM subsystem has been idle for 10 seconds.

The Active Power (ACTP) field indicates the largest average power of the NVM subsystem over a 10 second window on a particular workload (refer to section 8.1.17.3). Active Power measurement starts when the first command is submitted and ends when the last command is completed. The largest average power over a 10 second window, consumed by the NVM subsystem in that state is reported in the Active Power field. If the workload completes faster than 10 seconds, the average active power should be measured over the period of the workload. Non-operational states shall set Active Power Scale, Active Power Workload, and Active Power fields to 0h.

The host may dynamically modify the power state using the Set Features command and determine the current power state using the Get Features command. The host may directly transition between any two supported power states. The Entry Latency (ENLAT) field in the Power State Descriptor data structure indicates the maximum amount of time in microseconds to enter that power state and the Exit Latency (EXLAT) field indicates the maximum amount of time in microseconds to exit that state.

The maximum amount of time to transition between any two power states is equal to the sum of the old state's exit latency and the new state's entry latency. The host is not required to wait for a previously submitted power state transition to complete before initiating a new transition. The maximum amount of time for a sequence of power state transitions to complete is equal to the sum of transition times for each individual power state transition in the sequence.

Associated with each power state descriptor are Relative Read Throughput (RRT), Relative Write Throughput (RWT), Relative Read Latency (RRL) and Relative Write Latency (RWL) fields that provide the host with an indication of relative performance in that power state. Relative performance values provide an ordering of performance characteristics between power states. Relative performance values may repeat, may be skipped, and may be assigned in any order (i.e., increasing power states are not required to have increasing relative performance values).

A lower relative performance value indicates better performance (e.g., higher throughput or lower latency). For example, in Figure 625 power state 1 has higher read throughput than power state 2, and power states 0 through 3 all have the same read latency. Relative performance ordering is only with respect to a single performance characteristic. Thus, although the relative read throughput value of one power state may equal the relative write throughput value of another power state, this does not imply that the actual read and write performance of these two power states are equal.

The default NVM Express power state is implementation specific and shall correspond to a state that does not consume more power than the lowest value specified in the applicable form factor specification, if any. Refer to the Power Management section in the applicable NVM Express transport specification for transport specific power requirements impacting NVMe power states, if any.

#### 8.1.17.1 Non-Operational Power States

A power state may be a non-operational power state, as indicated by Non-Operational State (NOPS) field in Figure 313. Non-operational power states allow the following operations:

- property accesses;
- PMR accesses, if any;

- CMB accesses, if any;
- processing of Admin commands and processing background operations, if any, initiated by that command (e.g., Device Self-test command (refer to section 5.1.5), Sanitize command (refer to section 5.1.22)); and
- additional transport-specific accesses as defined in the applicable NVMe Transport binding specification.

For the operations listed in the preceding paragraph, the controller:

- may exceed the power advertised by the non-operational power state;
- shall logically remain in the current non-operational power state unless an I/O command is received or if an explicit transition is requested by a Set Features command with the Power Management Feature Identifier; and
- shall not exceed the maximum power advertised for the most recent operational power state.

HMB non-operational power state access restriction (refer to section 5.1.25.2.4) does not prohibit the controller from accessing the HMB while processing Admin commands and while performing host-initiated background operations initiated by Admin commands. HMB non-operational power state access restriction does prohibit the controller from accessing the HMB in order to perform controller-initiated activity (i.e., activity not directly associated with a command).

Execution of controller-initiated background operations may exceed the power advertised by the non-operational power state, if the Non-Operational Power State Permissive Mode is supported and enabled (refer to section 5.1.25.1.10).

No I/O commands are processed by the controller while in a non-operational power state. The host should wait until there are no pending I/O commands prior to issuing a Set Features command to change the current power state of the device to a non-operational power state and not submit new I/O commands until the Set Features command completes. Issuing an I/O command in parallel may result in the controller being in an unexpected power state.

When in a non-operational power state, regardless of whether autonomous power state transitions are enabled, the controller shall autonomously transition back to the most recent operational power state to process an I/O command.

### 8.1.17.2 Autonomous Power State Transitions

The controller may support autonomous power state transitions, as indicated in the Identify Controller data structure in Figure 312. Autonomous power state transitions provide a mechanism for the host to configure the controller to automatically transition between power states on certain conditions without software intervention.

The entry condition to transition to the Idle Transition Power State is that the controller has been in idle for a continuous period of time exceeding the Idle Time Prior to Transition time specified. The controller is idle when there are no commands outstanding to any I/O Submission Queue. If a controller has an operation in process (e.g., device self-test operation) that would cause controller power to exceed that advertised for the proposed non-operational power state, then the controller should not autonomously transition to that state.

The power state to transition to shall be a non-operational power state (a non-operational power state may autonomously transition to another non-operational power state). If an operational power state is specified by a Set Features command specifying the Autonomous Power State Transitions feature (i.e., the Feature Identifier field set to 0Ch (refer to section 5.1.25.1.6), then the controller should abort the command with a status code of Invalid Field in Command. Refer to section 8.1.17.1 for more details.

### 8.1.17.3 NVM Subsystem Workloads

The workload values described in this section may specify a workload hint in the Power Management Feature (refer to section 5.1.25.1.2) to inform the NVM subsystem or indicate the conditions for the active power level.

Active power values in the power state descriptors are specified for a particular workload since they may vary based on the workload of the NVM subsystem. The workload field indicates the conditions to observe the energy values. If Active Power is indicated for a power state, a corresponding workload shall also be indicated.

The workload values are described in Figure 626.

**Figure 626: Workload Hints**

Value	Definition
000b	<b>No Workload:</b> The workload is unknown or not provided.
001b	<b>Workload 1:</b> Extended Idle Period with a Burst of Random Writes. Workload #1 consists of five (5) minutes of idle followed by thirty-two (32) random write commands of size 1 MiB submitted to a single controller while all other controllers in the NVM subsystem are idle, and then thirty (30) seconds of idle.
010b	<b>Workload 2:</b> Heavy Sequential Writes. Workload #2 consists of 80,000 sequential write commands of size 128 KiB submitted to a single controller while all other controllers in the NVM subsystem are idle. The submission queue(s) should be sufficiently large allowing the host to ensure there are multiple commands pending at all times during the workload.
011b to 111b	Reserved

#### 8.1.17.4 Runtime D3 (RTD3) Transitions (PCIe only)

In Runtime D3, main power is removed from the controller. Auxiliary power may or may not be provided. RTD3 is used for additional power savings when the controller is expected to be idle for a period of time.

In this specification, RTD3 refers to the D3<sub>cold</sub> power state described in the PCI Express Base Specification. RTD3 does not include the PCI Express D3<sub>hot</sub> power state because main power is not removed from the controller in the D3<sub>hot</sub> power state. Refer to the PCI Express Base Specification for details on the D3<sub>hot</sub> power state and the D3<sub>cold</sub> power state.

To enable host software to determine when to use RTD3, the controller reports the RTD3 Entry Latency (RTD3E) field and the RTD3 Resume Latency (RTD3R) field in the Identify Controller data structure in Figure 312. The host may use the sum of these two values to evaluate whether the expected idle period is long enough to benefit from a transition to RTD3.

The RTD3 Resume Latency is the expected elapsed time from the time power is applied until the controller is able to:

- a) process and complete I/O commands; and
- b) access the resources (e.g., formatted storage) associated with attached namespace(s), if any, as part of I/O command processing.

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

If CSTS.ST is cleared to '0', then the RTD3 Entry Latency is the expected elapsed time from the time CC.SHN is set to 01b by host software until CSTS.SHST is set to 10b by the controller. When CSTS.SHST is set to 10b, the controller is ready for host software to remove power.

#### 8.1.17.5 Host Controlled Thermal Management

A controller may support host controlled thermal management (HCTM), as indicated in the Host Controlled Thermal Management Attributes of the Identify Controller data structure in Figure 312. Host controlled thermal management provides a mechanism for the host to configure a controller to automatically transition between active power states or perform vendor specific thermal management actions in order to attempt to meet thermal management requirements specified by the host. If active power states transitions are used to attempt to meet these thermal management requirements specified by the host, then those active power states transitions are vendor specific.

The host specifies and enables the thermal management requirements by setting the Thermal Management Temperature 1 field and/or Thermal Management Temperature 2 field (refer to section 5.1.25.1.9) in a Set Features command to a non-zero value. The supported range of values for the Thermal Management Temperature 1 field and Thermal Management Temperature 2 field are indicated in the Identify Controller data structure in Figure 312.

The Thermal Management Temperature 1 specifies that if the Composite Temperature (refer to Figure 207) is:

- a) greater than or equal to this value; and
- b) less than the Thermal Management Temperature 2, if non-zero,

then the controller should start transitioning to lower power active power states or perform vendor specific thermal management actions while minimizing the impact on performance in order to attempt to reduce the Composite Temperature (e.g., transition to an active power state that performs light throttling).

The Thermal Management Temperature 2 field specifies that if the Composite Temperature is greater than or equal to this value, then the controller shall start transitioning to lower power active power states or perform vendor specific thermal management actions regardless of the impact on performance in order to attempt to reduce the Composite Temperature (e.g., transition to an active power state that performs heavy throttling).

If the controller is currently in a lower power active power state or performing vendor specific thermal management actions because of this feature (e.g., throttling performance) because the Composite Temperature is:

- a) greater than or equal to the current value of the Thermal Management Temperature 1 field; and
- b) less than the current value of the Thermal Management Temperature 2 field,

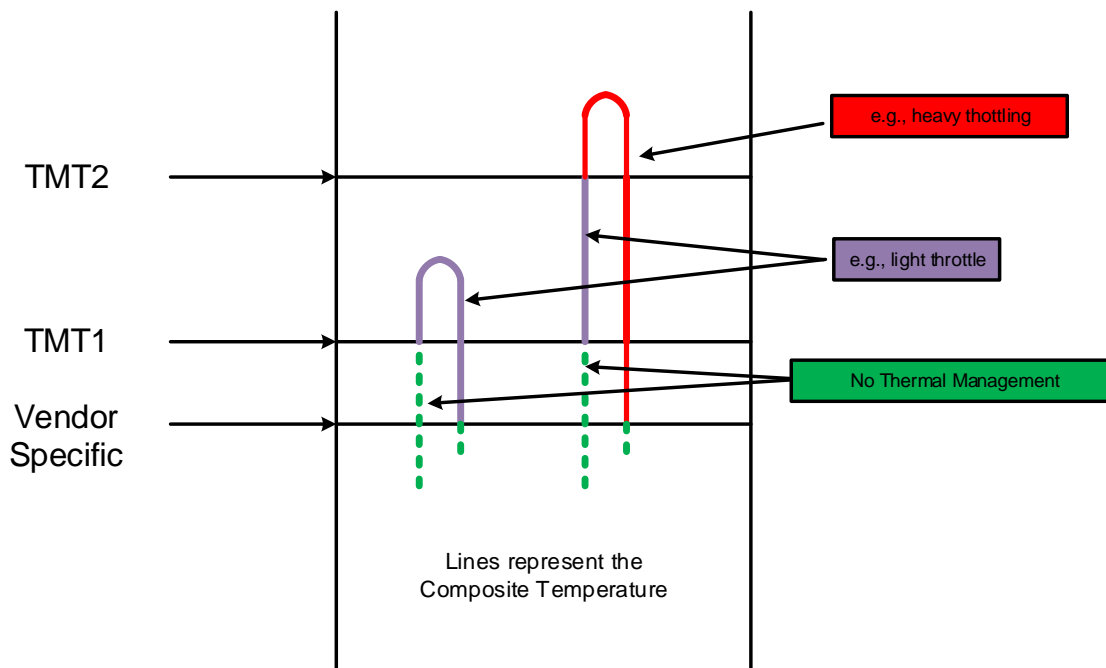
and the Composite Temperature decreases to a value below the current value of the Thermal Management Temperature 1 field, then the controller should return to the active power state that the controller was in prior to going to a lower power active power state or stop performing vendor specific thermal management actions because of this feature, the Composite Temperature and the current value of the Thermal Management Temperature 1 field.

If the controller is currently in a lower power active power state or performing vendor specific thermal management actions because the Composite Temperature is greater than or equal to the current value of the Thermal Management Temperature 2 field and the Composite Temperature decreases to a value less than the current value of the Thermal Management Temperature 1 field, then the controller should return to the active power state that the controller was in prior to going to a lower power active power state or stop performing vendor specific thermal management actions because of this feature, and the Composite Temperature.

The temperature at which the controller stops being in a lower power active power state or performing vendor specific thermal management actions because of this feature is vendor specific (i.e., hysteresis is vendor specific).

Figure 627 shows examples of how the Composite Temperature may be affected by this feature.

Figure 627: HCTM Example



Note: Since the host controlled thermal management (HCTM) feature uses the Composite Temperature, the actual interactions between a platform (e.g., tablet, or laptop) and two different device implementations may vary even with the same Thermal Management Temperature 1 and Thermal Management Temperature 2 temperature settings. The use of this feature requires validation between those devices' implementations and the platform in order to be used effectively.

The SMART / Health Information log page (refer to section 5.1.12.1.3) contains statistics related to Host Controller Thermal Management.

### 8.1.18 Predictable Latency Mode

Predictable Latency Mode is used to achieve predictable latency for read and write operations. When configured to operate in this mode using the Predictable Latency Mode Config Feature (refer to section 5.1.25.1.12), the namespaces in an NVM Set (refer to section 3.2.2) provide windows of operation for deterministic operation or non-deterministic operation.

When Predictable Latency Mode is enabled:

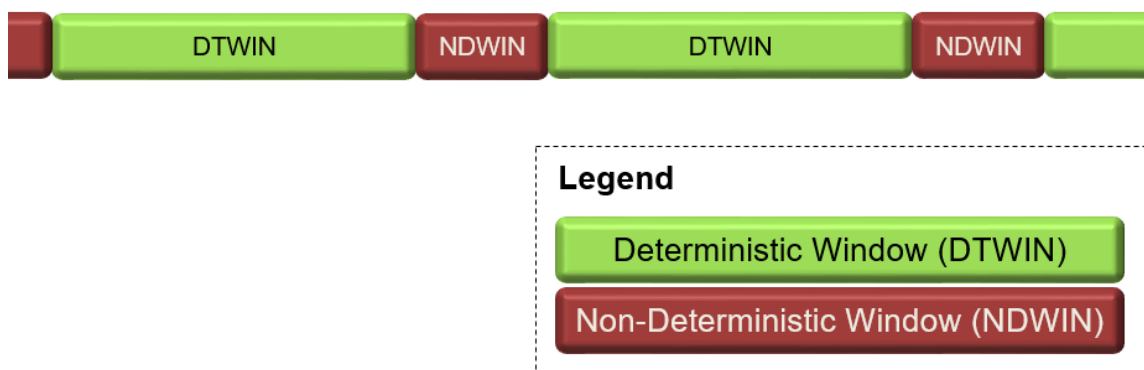
- NVM Sets and their associated namespaces have vendor specific quality of service attributes;
- I/O commands that access NVM in the same NVM Set have the same quality of service attributes; and
- I/O commands that access NVM in one NVM Set do not impact the quality of service of I/O commands that access NVM in a different NVM Set.

The quality of service attributes apply within the NVM subsystem and do not include the PCIe or fabric connection. To enhance isolation, the host should submit I/O commands for different NVM Sets to different I/O Submission Queues.

Read Recovery Levels (refer to section 8.1.20) shall be supported when Predictable Latency Mode is supported. The host configures the Read Recovery Level to specify the tradeoff between the quality of service versus the amount of error recovery to apply for a particular NVM Set.

The Deterministic Window (DTWIN) is the window of operation during which the NVM Set is able to provide deterministic latency for read and write operations. The Non-Deterministic Window (NDWIN) is the window of operation during which the NVM Set is not able to provide deterministic latency for read and write operations as a result of preparing for a subsequent Deterministic Window. Examples of actions that may be performed in the Non-Deterministic Window include background operations on the non-volatile storage media. The current window that an NVM Set is operating in is configured by the host using the Predictable Latency Mode Window Feature or by the controller as a result of an autonomous action.

**Figure 628: Deterministic and Non-Deterministic Windows**



To remain in the Deterministic Window, the host is required to follow operating rules (refer to section 8.1.18.1) ensuring that certain attributes do not exceed the typical or maximum values indicated in the Predictable Latency Per NVM Set log page. If the attributes exceed any of the typical or maximum values indicated in the Predictable Latency Per NVM Set log page or a Deterministic Excursion occurs, then the associated NVM Set may autonomously transition to the Non-Deterministic Window. A Deterministic Excursion is a rare occurrence in the NVM subsystem that requires immediate action by the controller.

The host configures Predictable Latency Events to report using the Predictable Latency Mode Config feature. The host may configure a Predictable Latency Event to be triggered when that value exceeds a specific value in order to manage window changes and avoid autonomous transitions by the controller. Refer to section 8.1.18.3.

If Predictable Latency Mode is supported, then all controllers in the NVM subsystem shall:

- Support one or more NVM Sets;
- Support Read Recovery Levels;
- Support the Predictable Latency Mode log page for each NVM Set;
- Support the Predictable Latency Event Aggregate log page;
- Support the Predictable Latency Mode Config Feature;
- Support the Predictable Latency Mode Window Feature;
- Support Predictable Latency Event Aggregate Log Change Notices; and
- Indicate support for Predictable Latency Mode in the Controller Attributes field in the Identify Controller data structure.

### 8.1.18.1 Host Operating Rules to Achieve Determinism

In order to achieve deterministic operation, the host is required to follow operating rules.

An NVM Set remains in the Deterministic Window while attributes do not exceed any of the typical or maximum values indicated in the Predictable Latency Per NVM Set log page, there is not a Deterministic Excursion, and the host does not request a transition to the Non-Deterministic Window. The attributes

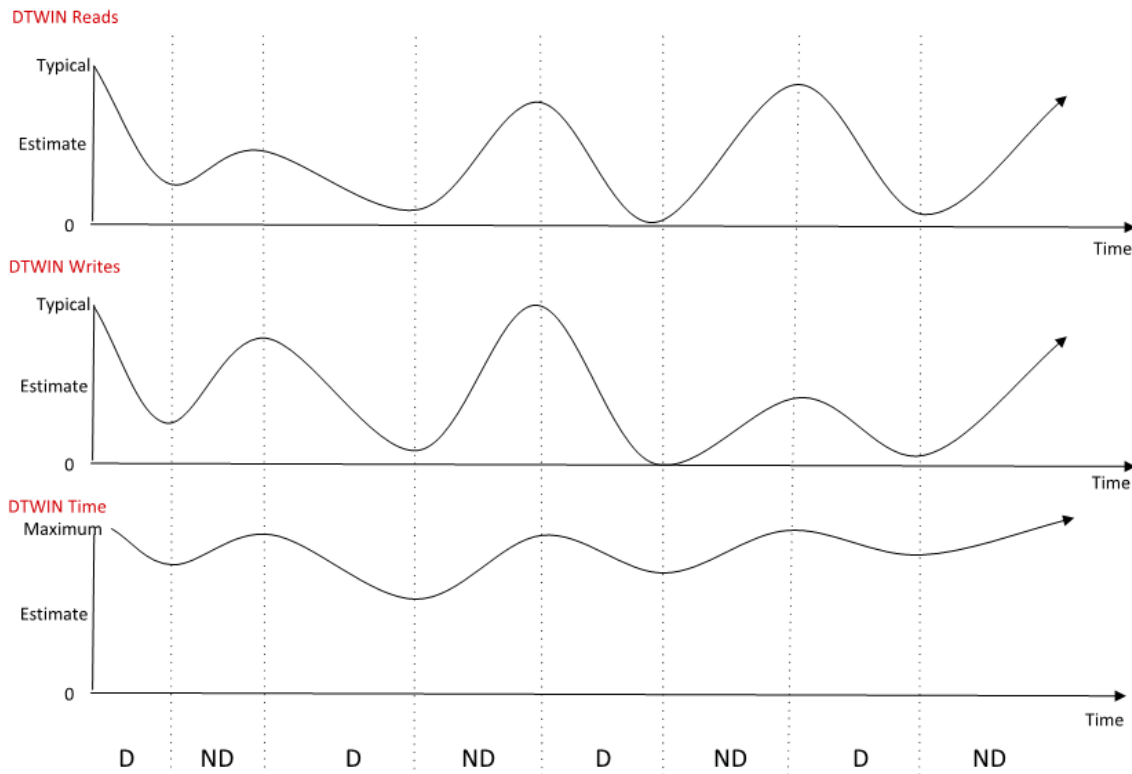
specified in this specification are the number of random 4 KiB reads, the number of writes in Optimal Write Size, and time in the Deterministic Window. Additional attributes are vendor specific.

For reads, writes, and time in the Deterministic Window, two values are provided in the Predictable Latency Per NVM Set log page (refer to section 5.1.12.1.11):

- A typical or maximum amount of that attribute that the host may consume during any given DTWIN; and
- A reliable estimate of the amount of that attribute that remains to be consumed during the current DTWIN.

Figure 629 shows how the Typical, Maximum, and Reliable Estimates for the DTWIN attributes increase or decrease when the associated NVM Set is in the Deterministic Window or Non-Deterministic Window.

**Figure 629: DTWIN Attributes and Estimates**

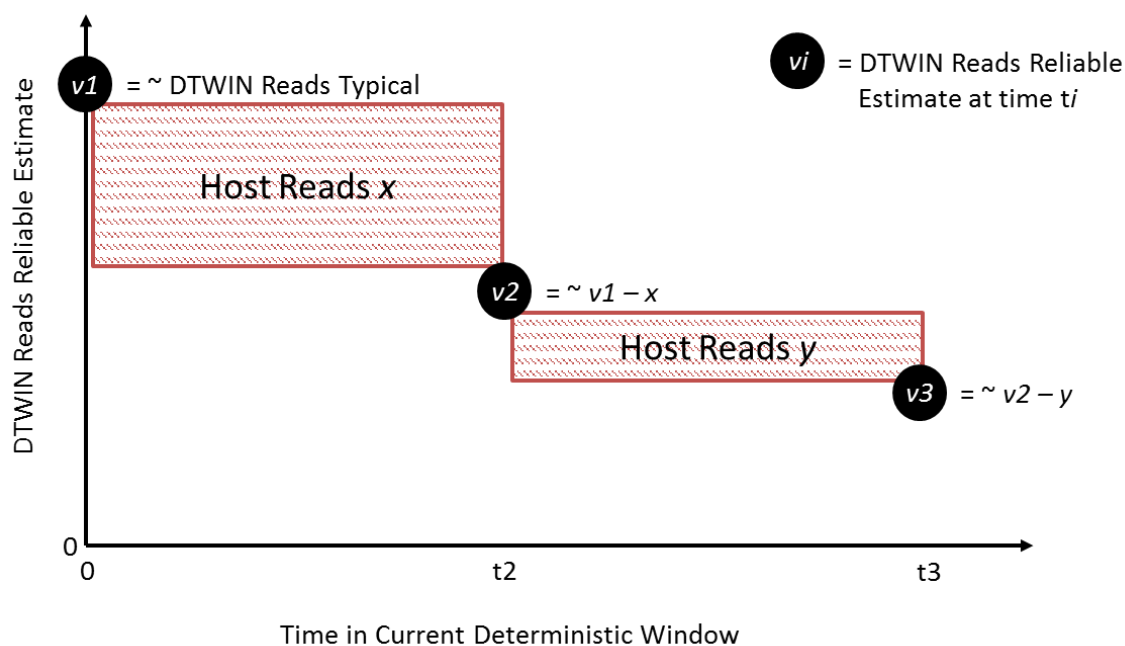


An NVM Set may transition autonomously to the NDWIN if, since entry to the current DTWIN:

- the number of reads is greater than the value indicated in the DTWIN Reads Typical field;
- the number of writes is greater than the value indicated in the DTWIN Writes Typical field;
- the amount of time indicated in the DTWIN Time Maximum field has passed; or
- a Deterministic Excursion occurs.

Figure 630 is an example that shows the relationship between the typical and reliable estimate values for DTWIN Reads. DTWIN Reads Reliable Estimate begins near the DTWIN Reads Typical value at the start of the current DTWIN at time 0. During the first time increment, the host reads  $x$  units, and the value of the reliable estimate at time  $t_2$  is decremented by approximately  $x$ . During the second time increment, the host reads a smaller amount consisting of  $y$  units and thus the reliable estimate at  $t_3$  is decremented by approximately  $y$ .

Figure 630: Typical and Reliable Estimate Example



The host configures the current window to be either DTWIN or NDWIN using a Set Features command with the Predictable Latency Mode Window Feature. The host may use the reliable estimates provided in the Predictable Latency Mode log page to ensure that the host transitions the NVM Set to the NDWIN prior to any reliable estimates exceeding one of the typical or maximum values (e.g., DTWIN Reads Estimate = 0).

The reliable estimates provided shall have the following properties when in the Deterministic Window:

- The estimates shall be monotonically decreasing towards 0h for the entirety of the DTWIN, depending on the attribute. For example, DTWIN Reads Reliable Estimate is monotonically decreasing and thus does not increase without transitioning from the DTWIN to the NDWIN; and
- The estimates shall not change abruptly unless operating conditions have changed abruptly. The estimate should be based on averaging or smoothing of data collected over some period of time.

### 8.1.18.2 Configuring Periodic Windows

When using the NVM Set in Predictable Latency Mode, the host should transition the controller to NDWIN for periodic maintenance. The maintenance is required in order for the NVM subsystem to reliably provide the amount of time indicated for Deterministic Windows.

There are three static time based parameters reported in the Predictable Latency Per NVM Set log page (refer to section 5.1.12.1.11) that may be used by the host to configure periodic windows. The values provided are worst-case for the life of the NVM subsystem:

- NDWIN Time Minimum Low is the minimum time that the NVM Set remains in the Non-Deterministic Window. The controller may delay completion of a Set Features command requesting a transition to the Deterministic Window until this time is completed. This time does not account for additional host activity in the Non-Deterministic Window;
- NDWIN Time Minimum High is the minimum time that the host should allow the NVM Set to remain in the Non-Deterministic Window after the NVM Set remained in the previous Deterministic Window for DTWIN Time Maximum. This time does not account for additional host activity in the Non-Deterministic Window; and
- DTWIN Time Maximum is the maximum time that the NVM Set is able to stay in a Deterministic Window.



The DTWIN Time Maximum and NDWIN Time Minimum High may provide a ratio of the amount of maintenance that needs to be performed based on the time that the NVM Set remains in the DTWIN, assuming no threshold is exceeded. Any scaling of the time in the Non-Deterministic Window based on the read, write, and time behavior in the previous Deterministic Window is implementation dependent.

The DTWIN Time Estimate may be used by the host when a Deterministic Excursion has occurred. This estimate allows the host to re-synchronize an NVM Set with other NVM Sets operating in Predictable Latency Mode, if applicable.

### 8.1.18.3 Configuring and Managing Events

The host may configure events to be triggered when thresholds do not exceed certain levels or when autonomous transitions occur using the Predictable Latency Mode Feature. The host submits a Set Feature command for the particular NVM Set and configures the specific event(s) and threshold(s) values that shall trigger a Predictable Latency Event Aggregate Log Change Notice event for that particular NVM Set to the host. Refer to Figure 404.

The host determines the NVM Sets that have outstanding events by reading the Predictable Latency Event Aggregate log page (refer to section 5.1.12.1.12). An entry is returned for each NVM Set that has an event outstanding. The host may use the NVM Set Identifier Maximum value reported in the Identify Controller data structure in order to determine the maximum size of this log page.

To determine the specific event(s) that have occurred for a reported NVM Sets, the host reads the Predictable Latency Per NVM Set log page (refer to section 5.1.12.1.11) for that NVM Set. The Event Type field indicates the event(s) that have occurred (e.g., an autonomous transition to the NDWIN). An event(s) for a particular NVM Set is cleared if the controller successfully processes a read for the Predictable Latency Per NVM Set log page for the affected NVM Set where the Get Log Page command has the Retain Asynchronous Event parameter cleared to '0'. If the Event Type field in the Predictable Latency Per NVM Set log page is cleared to 0h, then events for that particular NVM Set are not reported in the Predictable Latency Event Aggregate log page.

## 8.1.19 Reachability Reporting architecture

### 8.1.19.1 Reachability Reporting overview

For some operations that specify multiple namespaces it is necessary to indicate what namespaces are able to be used in those operations (e.g., Copy command that specifies multiple namespaces, Execute command in the Computational Programs command set). This ability is called reachability. Reachability is defined at the controller level (i.e., only namespaces attached to the same controller are reachable).

A Reachability Group becomes available to a controller if:

- a) a namespace becomes attached to that controller and that namespace is associated with a Reachability Group that did not currently have any namespaces attached to that controller; or
- b) a configuration change in an NVM subsystem changes the membership in a Reachability Group (e.g., a new Reachability Group is present in the Reachability Groups log page (refer to section 5.1.12.1.25)).

A Reachability Group becomes unavailable to a controller if a namespace is detached that is the only namespace attached to that controller for that Reachability Group (i.e., a Reachability Group that was present in the Reachability Groups log page is no longer present).

A Reachability Association change occurs if a Reachability Group that is associated with that Reachability Association becomes available or unavailable to that controller. (e.g., a new Reachability Association is present in the Reachability Associations log page (refer to section 5.1.12.1.26) or a Reachability Association that was present in the Reachability Associations log page is no longer present).

If an NVM subsystem supports Reachability Reporting, then all controllers in that NVM subsystem shall:

- set the RRSUP bit to '1' in the Controller Reachability Capabilities (CRCAP) field in the Identify Controller data structure (refer to Figure 312) to indicate support for Reachability Reporting;
- support Reachability Groups Change Notices (refer to section 5.1.25.1.5);

- support Reachability Association Change Notices (refer to section 5.1.25.1.5);
- support the Reachability Groups log page; and
- support the Reachability Associations log page.

Namespaces attached to a controller in an NVM subsystem that supports Reachability Reporting shall:

- be members of a Reachability Group; and
- supply a valid Reachability Group Identifier in the Reachability Group Identifier (RGRPID) field in the I/O Command Set Independent Identify Namespace Data Structure (refer to Figure 319).

Reachability and characteristics of reachability are indicated in the combination of the Reachability Groups log page (refer to section 5.1.12.1.25) and the Reachability Associations log page (refer to section 5.1.12.1.26). A Reachability Group defines a group of namespaces that are all associated with namespaces in other Reachability Groups defined by a Reachability Association. The method of assigning Reachability Group identifiers and Reachability Association identifiers is outside the scope of this specification. If members of a Reachability Group are able to reach members of that Reachability Group, then a Reachability Association that contains only that Reachability Group specifies the characteristics of that reachability. All Reachability Groups in a Reachability Association have identical access characteristics. All namespaces in a Reachability Group have identical access characteristics. If a Reachability Group is not in any Reachability Association then the members of that Reachability Group are not reachable by any other namespaces, including the members of that Reachability Group. A namespace is always able to reach itself (i.e., a command that has two or more NSIDs as part of the command where those NSIDs are all the same is not constrained by reachability).

If the Reachability Association Characteristics field (refer to Figure 278) is set to is 01h, then all namespaces in the associated Reachability Groups are able to reach other namespaces in the other Reachability Groups in that Reachability Association without any indication of a performance characteristic (e.g., for the Execute command in the Computational Programs Command Set). If the Reachability Association Characteristics field is set to 02h, then all namespaces in the associated Reachability Groups are reachable and support fast copy operations as defined by the applicable I/O Command Set specification (e.g., the Fast copy operations section in the NVM Command Set Specification). If the Reachability Association Characteristics field is set to 03h, then all namespaces in the associated Reachability Groups are reachable but do not support fast copy operations.

A namespace is only allowed to be in one Reachability Group. A Reachability Group is in zero or more Reachability Associations with different values of the Reachability Association Characteristics field for each Reachability Association.

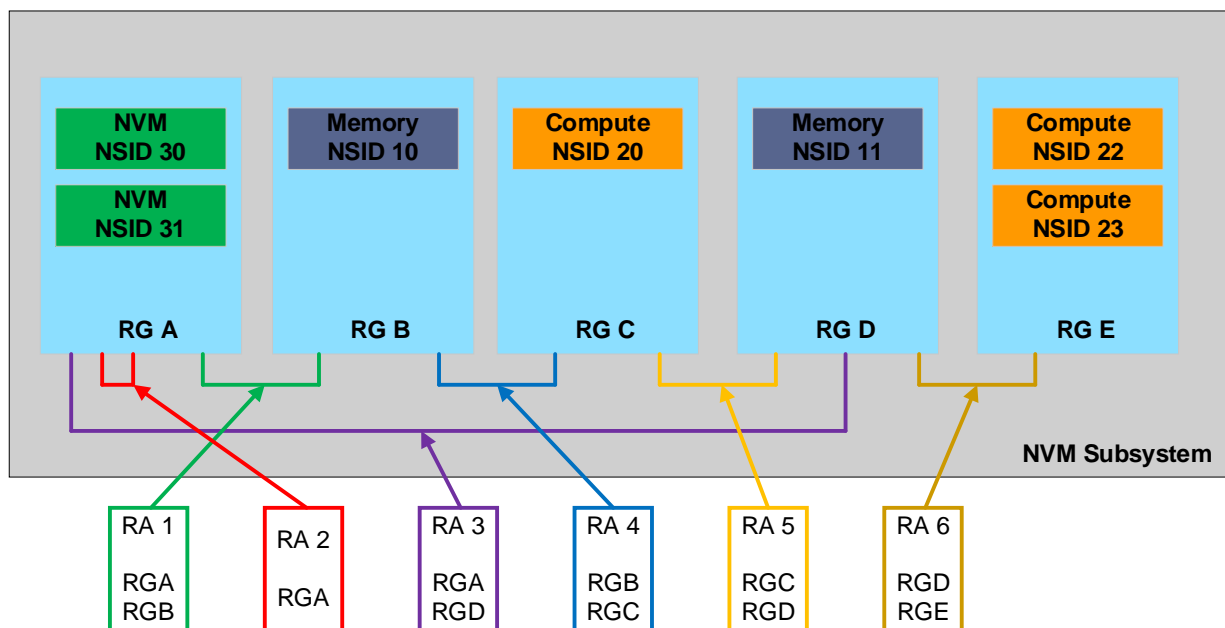
The Reachability Group Identifier (RGRPID) for each Reachability Group shall be unique within the NVM subsystem. If RGIDC bit in the CRCAP field in the Identify Controller data structure is set to '1', then the Reachability Group Identifier shall not change while the namespace is attached to any controller in the NVM subsystem. If RGIDC bit in the CRCAP field is cleared to '0', then the Reachability Group Identifier may change while the namespace is attached to any controller in the NVM subsystem. If the Reachability Group Identifier changes, the controller shall issue the Reachability Groups Change Notice as described in this section.

Figure 631 is an example configuration that shows that:

- Reachability Association 1 (i.e., RA 1):
  - specifies that:
    - NSID 10 and NSID 30 are able to reach each other; and
    - NSID 10 and NSID 31 are able to reach each other;
  - and
  - does not specify any reachability between NSID 30 and NSID 31;
- Reachability Association 2 (i.e., RA 2) is a self-reference for Reachability Group A (i.e., RG A) that specifies:

- that NSID 30 and NSID 31 are able to reach each other;
- Reachability Association 3 (i.e., RA 3):
  - specifies that:
    - NSID 11 and NSID 30 are able to reach each other; and
    - NSID 11 and NSID 31 are able to reach each other;
  - and
  - does not specify any reachability between NSID 30 and NSID 31;
- The lack of a Reachability Association that contains Reachability Group B (i.e., RG B) and Reachability Group D (i.e., RG D) specifies:
  - that NSID 10 and NSID 11 are not able to reach each other;
- and
- The lack of a self-referencing Reachability Association for Reachability Group E (i.e., RG E) specifies:
  - that NSID 22 and NSID 23 are not able to reach each other.

**Figure 631: Reachability Associations Example**



### 8.1.19.2 Reachability event notifications

Reachability may generate a Reachability Group Change notice (refer to section 5.1.2.1) or a Reachability Association Change notice (refer to section 5.1.2.1).

Receipt of a Reachability Group Change Notice from a controller may indicate:

- a) an NSID has been added to one or more of the Reachability Group Descriptors;
- b) an NSID has been removed from one or more of the Reachability Group Descriptors;
- c) a Reachability Group no longer has any NSIDs attached to this controller as members of that Reachability Group and therefore is no longer reported in the Reachability Groups log page for this controller; or
- d) the NSID of a namespace has moved from one Reachability Group Descriptor to a different Reachability Group Descriptor (i.e., the RGRPID field in the I/O Command Set Independent Identify

Namespace Data Structure for that namespace has changed), if RGIDC bit in the CRCAP field is cleared to '0' in the Identify Controller data structure (refer to Figure 312).

As a result of receiving a Reachability Group Change notice, the host should read the Reachability Groups log page (refer to section 5.1.12.1.25) to check for each of those possible changes.

Receipt of a Reachability Association Change Notice from a controller may indicate:

- a) an RGID has been added to one or more of the Reachability Association Descriptors;
- b) an RGID has been removed from one or more of the Reachability Association Descriptors; and/or
- c) a Reachability Association no longer has any RGIDs associated with this controller as members of that Reachability Association and therefore is no longer reported in the Reachability Associations log page for this controller.

As a result of receiving a Reachability Association Change notice, the host should read the Reachability Associations log page (refer to section 5.1.12.1.26) to check for each of those possible changes.

### **8.1.20 Read Recovery Level**

The Read Recovery Level (RRL) is an NVM Set configurable attribute that balances the completion time for read commands and the amount of error recovery applied to those read commands. The Read Recovery Level applies to an NVM Set with which the Read Recovery Level is associated. A namespace created within an NVM Set inherits the Read Recovery Level of that NVM Set. If NVM Sets are not supported, all namespaces in the NVM subsystem use an identical Read Recovery Level.

The controller indicates support for Read Recovery Levels in the Controller Attributes field in the Identify Controller data structure (refer to Figure 312). If Read Recovery Levels are supported, then the specific levels supported are indicated in the Read Recovery Levels Supported field in the Identify Controller data structure. There are 16 levels that may be supported. Level 0, if supported, provides the maximum amount of recovery. Level 4 is a mandatory level that provides a nominal amount of recovery and is the default level. Level 15 is a mandatory level that provides the minimum amount of recovery and is referred to as the 'Fast Fail' level. The levels are organized based on the amount of recovery supported, such that a higher numbered level provides less recovery than the preceding lower level.


Interactions between the Read Recovery Level and the Limited Retry (LR) field in I/O commands are implementation specific.

The Read Recovery Level may be configured using a Set Features command for the Read Recovery Level Config Feature. The Read Recovery Level may be determined using a Get Features command for the Read Recovery Level Config Feature.

**Figure 632: Read Recovery Level Overview**

Level	O/M	Definition
0	O	
1	O	
2	O	
3	O	
4	M	Default
5	O	
6	O	
7	O	
8	O	
9	O	
10	O	
11	O	
12	O	
13	O	
14	O	
15	M	Fast Fail

Maximum  
Recovery



Decreasing  
Amount of  
Recovery

Minimum  
Recovery

If Read Recovery Levels are supported, then the NVM subsystem and all controllers shall:

- Support at least Level 4 and Level 15;
- Indicate support for Read Recovery Levels in the Controller Attributes field in the Identify Controller data structure;
- Support the Read Recovery Levels Supported field in the Identify Controller data structure; and
- Support the Read Recovery Level Config Feature.

### 8.1.21 Replay Protected Memory Block

The Replay Protected Memory Block (RPMB) provides a means for the system to store data to a specific memory area in an authenticated and replay protected manner. This is provided by first programming authentication key information to the controller that is used as a shared secret. The system is not authenticated in this phase, therefore the authentication key programming should be done in a secure environment (e.g., as part of the manufacturing process). The authentication key is utilized to sign the read and write accesses made to the replay protected memory area with a Message Authentication Code (MAC). Use of random number (nonce) generation and a write count property provide additional protection against replay of messages where messages could be recorded and played back later by an attacker.

Any attempt to access the replay protected memory area prior to the Authentication Key being programmed results in an RPMB Operation Result Operation Status field set to 07h (i.e., Authentication Key not yet

programmed) (refer to Figure 635). Once the key is programmed, the RPMB Operation Result Operation Status field shall not be set to 07h.

An Authenticated Data Write to the RPMB Device Configuration Block data structure that attempts to set the Boot Partition Write Protection Enabled bit when RPMB Boot Partition Write Protection is not supported results in an RPMB Operation Result Operation Status of 05h (i.e., Write failure in the RPMB Operation Status field) (refer to Figure 635).

An Authenticated Data Write to the RPMB Device Configuration Block data structure that attempts to set the Boot Partition Write Protection Enabled bit when either Boot Partition is in the Write Locked Until Power Cycle state (refer to section 5.1.25.1.32 and section 8.1.3.3.1) results in an RPMB Operation Result Operation Status of 05h (i.e., Write failure in the RPMB Operation Status field) (refer to Figure 635).

An Authenticated Data Write to the RPMB Device Configuration Block data structure that attempts to change either the Boot Partition 0 Write Locked bit or the Boot Partition 1 Write Locked bit when the Boot Partition Write Protection Enabled bit is cleared to '0' results in an RPMB Operation Result Operation Status field set to 05h (i.e., Write failure) (refer to Figure 635).

If Set Features Boot Partition Write Protection is supported, then an Authenticated Data Write to the RPMB Device Configuration Block data structure that successfully enables RPMB Boot Partition Write Protection shall also result in the controller changing the Boot Partition 0 Write Protection State and Boot Partition 1 Write Protection State values in the Boot Partition Write Protection Config feature to a value of 100b to indicate that Boot Partition write protection is controlled by RPMB.

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

The message types defined in Figure 634 are used by the host to communicate with an RPMB target. Request Message Types are sent from the host to the controller using Security Send commands. Response Message Types are retrieved by the host from the controller using Security Receive commands.

Figure 633 defines the RPMB Device Configuration Block data structure – the non-volatile contents stored within the controller for RPMB target 0.

**Figure 633: RPMB Device Configuration Block Data Structure**

Bytes	Type	Description						
00	RW	<b>Boot Partition Protection Enable (BPPEE):</b> This field specifies whether Boot Partition Protection is enabled.						
		<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:1</td> <td>Reserved</td> </tr> <tr> <td>0</td> <td><b>Boot Partition Write Protection Enabled (BPPED):</b> If this bit is set to '1', then RPMB Boot Partition Write Protection is enabled. If this bit is cleared to '0', then RPMB Boot Partition Write Protection is disabled or not supported. Once enabled, the controller shall prevent disabling RPMB Boot Partition Write Protection.</td> </tr> </tbody> </table>	Bits	Description	7:1	Reserved	0	<b>Boot Partition Write Protection Enabled (BPPED):</b> If this bit is set to '1', then RPMB Boot Partition Write Protection is enabled. If this bit is cleared to '0', then RPMB Boot Partition Write Protection is disabled or not supported. Once enabled, the controller shall prevent disabling RPMB Boot Partition Write Protection.
		Bits	Description					
7:1	Reserved							
0	<b>Boot Partition Write Protection Enabled (BPPED):</b> If this bit is set to '1', then RPMB Boot Partition Write Protection is enabled. If this bit is cleared to '0', then RPMB Boot Partition Write Protection is disabled or not supported. Once enabled, the controller shall prevent disabling RPMB Boot Partition Write Protection.							
01	RW	<b>Boot Partition Protection State (BPLS):</b> This field specifies the current Boot Partition protection state when RPMB Boot Partition Write Protection is enabled. This field shall be cleared to 0h unless RPMB Boot Partition Write Protection is enabled. Refer to section 8.1.3.3.						
		<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:2</td> <td>Reserved</td> </tr> <tr> <td>1</td> <td><b>Boot Partition 1 Write Locked (BPP1L):</b> If this bit is set to '1', then Boot Partition 1 (i.e., the BPID bit in Figure 181 is set to 1) is write locked. If this bit is cleared to '0', then the Boot Partition 1 is write unlocked.</td> </tr> </tbody> </table>	Bits	Description	7:2	Reserved	1	<b>Boot Partition 1 Write Locked (BPP1L):</b> If this bit is set to '1', then Boot Partition 1 (i.e., the BPID bit in Figure 181 is set to 1) is write locked. If this bit is cleared to '0', then the Boot Partition 1 is write unlocked.
		Bits	Description					
7:2	Reserved							
1	<b>Boot Partition 1 Write Locked (BPP1L):</b> If this bit is set to '1', then Boot Partition 1 (i.e., the BPID bit in Figure 181 is set to 1) is write locked. If this bit is cleared to '0', then the Boot Partition 1 is write unlocked.							

**Figure 633: RPMB Device Configuration Block Data Structure**

Bytes	Type	Description								
		<table border="1"> <tr> <td>0</td> <td><b>Boot Partition 0 Write Locked (BPP0L):</b> If this bit is set to '1', then Boot Partition 0 (i.e., the BPID bit in Figure 181 is cleared to 0) is write locked. If this bit is cleared to '0', then the Boot Partition 0 is write unlocked.</td> </tr> </table>	0	<b>Boot Partition 0 Write Locked (BPP0L):</b> If this bit is set to '1', then Boot Partition 0 (i.e., the BPID bit in Figure 181 is cleared to 0) is write locked. If this bit is cleared to '0', then the Boot Partition 0 is write unlocked.						
0	<b>Boot Partition 0 Write Locked (BPP0L):</b> If this bit is set to '1', then Boot Partition 0 (i.e., the BPID bit in Figure 181 is cleared to 0) is write locked. If this bit is cleared to '0', then the Boot Partition 0 is write unlocked.									
02	RW	<p><b>Write Protection Control (WPC):</b> This field specifies whether the controller processes or aborts Set Features commands which enable certain namespace write protection states (refer to section 8.1.16 and section 5.1.25.1.31). If the controller does not support Namespace Write Protection, then this field shall be cleared to 0h.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>7:2</td> <td>Reserved</td> </tr> <tr> <td>1</td> <td> <p><b>Permanent Write Protect Control (PWPC):</b> This bit cleared to '0' specifies that the controller shall fail a Set Features command which attempts to set the namespace write protection state to Permanent Write Protect, as defined in section 8.1.16. This bit set to '1' specifies that the controller shall process a Set Features command which attempts to set the namespace write protection state to Permanent Write Protect.</p> <p>If the controller supports Namespace Write Protection, then this bit shall be cleared to '0' after a power cycle or a Controller Level Reset.</p> </td> </tr> <tr> <td>0</td> <td> <p><b>Write Protect Until Power Cycle Control (WPUPCC):</b> This bit cleared to '0' specifies that the controller shall fail a Set Features command which attempts to set the namespace write protection state to Write Protect Until Power Cycle, as defined in section 8.1.16. This bit set to '1' specifies that the controller shall process a Set Features command which sets the namespace write protection state to Write Protect Until Power Cycle.</p> <p>If the controller supports Namespace Write Protection, then this bit shall be cleared to '0' after a power cycle or a Controller Level Reset.</p> </td> </tr> </tbody> </table>	Bits	Description	7:2	Reserved	1	<p><b>Permanent Write Protect Control (PWPC):</b> This bit cleared to '0' specifies that the controller shall fail a Set Features command which attempts to set the namespace write protection state to Permanent Write Protect, as defined in section 8.1.16. This bit set to '1' specifies that the controller shall process a Set Features command which attempts to set the namespace write protection state to Permanent Write Protect.</p> <p>If the controller supports Namespace Write Protection, then this bit shall be cleared to '0' after a power cycle or a Controller Level Reset.</p>	0	<p><b>Write Protect Until Power Cycle Control (WPUPCC):</b> This bit cleared to '0' specifies that the controller shall fail a Set Features command which attempts to set the namespace write protection state to Write Protect Until Power Cycle, as defined in section 8.1.16. This bit set to '1' specifies that the controller shall process a Set Features command which sets the namespace write protection state to Write Protect Until Power Cycle.</p> <p>If the controller supports Namespace Write Protection, then this bit shall be cleared to '0' after a power cycle or a Controller Level Reset.</p>
Bits	Description									
7:2	Reserved									
1	<p><b>Permanent Write Protect Control (PWPC):</b> This bit cleared to '0' specifies that the controller shall fail a Set Features command which attempts to set the namespace write protection state to Permanent Write Protect, as defined in section 8.1.16. This bit set to '1' specifies that the controller shall process a Set Features command which attempts to set the namespace write protection state to Permanent Write Protect.</p> <p>If the controller supports Namespace Write Protection, then this bit shall be cleared to '0' after a power cycle or a Controller Level Reset.</p>									
0	<p><b>Write Protect Until Power Cycle Control (WPUPCC):</b> This bit cleared to '0' specifies that the controller shall fail a Set Features command which attempts to set the namespace write protection state to Write Protect Until Power Cycle, as defined in section 8.1.16. This bit set to '1' specifies that the controller shall process a Set Features command which sets the namespace write protection state to Write Protect Until Power Cycle.</p> <p>If the controller supports Namespace Write Protection, then this bit shall be cleared to '0' after a power cycle or a Controller Level Reset.</p>									
511:03		Reserved								

**Figure 634: RPMB Request and Response Message Types**

Request Message Types		Description	Requires Data	RPMB Frame Length (bytes)
0001h	Authentication key programming request	The host is attempting to program the Authentication Key for the selected RPMB target to the controller.	No	256
0002h	Reading of the Write Counter value request	The host is requesting to read the current Write Counter value from the selected RPMB target.	No	256
0003h	Authenticated data write request	The host is attempting to write data to the selected RPMB target.	Yes	M + 256
0004h	Authenticated data read request	The host is attempting to read data from the selected RPMB target.	No	256
0005h	Result read request	The host is attempting to read the result code for any of the other Message Types.	No	256
0006h	Authenticated Device Configuration Block write request	The host is attempting to write Device Configuration Block (DCB) to the selected RPMB target. This request message type is only valid for RPMB target 0.	Yes	512 + 256
0007h	Authenticated Device Configuration Block read request	The host is attempting to read Device Configuration Block (DCB) from the selected RPMB target. This request message type is only valid for RPMB target 0.	No	256
0100h	Authentication key programming response	Returned as a result of the host requesting a Result read request Message Type after programming the Authentication Key.	No	256

**Figure 634: RPMB Request and Response Message Types**

Request Message Types		Description	Requires Data	RPMB Frame Length (bytes)
0200h	Reading of the Write Counter value response	Returned as a result of the host requesting a Result read request Message Type after requesting the Write Counter value.	No	256
0300h	Authenticated data write response	Returned as a result of the host requesting a Result read request Message Type after attempting to write data to an RPMB target.	No	256
0400h	Authenticated data read response	Returned as a result of the host requesting a Result read request Message Type after attempting to read data from an RPMB target.	Yes	M + 256
0600h	Authenticated Device Configuration data write response	Returned as a result of the host requesting a Result read request Message Type after attempting to write a Device Configuration Block to an RPMB target.	No	256
0700h	Authenticated Device Configuration data read response	Returned as a result of the host requesting a Result read request Message Type after attempting to read DCB from an RPMB target.	Yes	512 + 256

The operation result defined in Figure 635 indicates whether an RPMB request was successful or not.

**Figure 635: RPMB Operation Result**

Bits	Description																						
15:08	Reserved																						
07	<b>Write Counter Status (WCS):</b> Indicates if the Write Counter has expired (i.e., reached its maximum value). A value of '1' indicates that the Write Counter has expired. A value of '0' indicates a valid Write Counter.																						
06:00	<b>Operation Status (OPSTAT):</b> Indicates the operation status. Valid operation status values are listed below.																						
	<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00h</td> <td>Operation successful</td> </tr> <tr> <td>01h</td> <td>General failure</td> </tr> <tr> <td>02h</td> <td>Authentication failure (MAC comparison not matching, MAC calculation failure)</td> </tr> <tr> <td>03h</td> <td>Counter failure (counters not matching in comparison, counter incrementing failure)</td> </tr> <tr> <td>04h</td> <td>Address failure (address out of range, wrong address alignment)</td> </tr> <tr> <td>05h</td> <td>Write failure (data/counter/result write failure)</td> </tr> <tr> <td>06h</td> <td>Read failure (data/counter/result read failure)</td> </tr> <tr> <td>07h</td> <td>Authentication Key not yet programmed</td> </tr> <tr> <td>08h</td> <td>Invalid RPMB Device Configuration Block – this may be used when the target is not 0h.</td> </tr> <tr> <td>09 to 3Fh</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00h	Operation successful	01h	General failure	02h	Authentication failure (MAC comparison not matching, MAC calculation failure)	03h	Counter failure (counters not matching in comparison, counter incrementing failure)	04h	Address failure (address out of range, wrong address alignment)	05h	Write failure (data/counter/result write failure)	06h	Read failure (data/counter/result read failure)	07h	Authentication Key not yet programmed	08h	Invalid RPMB Device Configuration Block – this may be used when the target is not 0h.	09 to 3Fh	Reserved
	Value	Definition																					
	00h	Operation successful																					
	01h	General failure																					
	02h	Authentication failure (MAC comparison not matching, MAC calculation failure)																					
	03h	Counter failure (counters not matching in comparison, counter incrementing failure)																					
	04h	Address failure (address out of range, wrong address alignment)																					
	05h	Write failure (data/counter/result write failure)																					
	06h	Read failure (data/counter/result read failure)																					
07h	Authentication Key not yet programmed																						
08h	Invalid RPMB Device Configuration Block – this may be used when the target is not 0h.																						
09 to 3Fh	Reserved																						

Figure 636 defines the non-volatile contents stored within the controller for each RPMB target.

**Figure 636: RPMB Contents**

Content	Type	Size	Description
Authentication Key	Write once, not erasable or readable	Size is dependent on authentication method reported in Identify Controller data structure (e.g., SHA-256 is 32 bytes (refer to RFC 6234))	Authentication key which is used to authenticate accesses when MAC is calculated.



**Figure 636: RPMB Contents**

Content	Type	Size	Description
Write Counter	Read only	4 bytes	Counter value for the total amount of successful authenticated data write requests made by the host. The initial value of this property after manufacture is 00000000h. The value is incremented by one automatically by the controller with each successful programming access. The value is not resettable. After the counter has reached the maximum value of FFFFFFFFh, the controller shall no longer increment to prevent overflow.
RPMB Data Area	Readable and writable, not erasable	Size is reported in Identify Controller data structure (128 KiB minimum, 32 MiB maximum)	Data that is able to be read and written only via successfully authenticated read/write access.

Each RPMB Data Frame is 256 bytes in size plus the size of the Data field, and is organized as shown in Figure 637. RPMB uses a sector size of 512 bytes. The RPMB sector size is independent and not related to the user data size used for the namespace(s).

**Figure 637: RPMB Data Frame**

Bytes	Description
222- $N$ :00	<b>Stuff Bytes (STBY):</b> Padding for the frame. Values in this field are not part of the MAC calculation. The size is 223 bytes minus the size of the Authentication Key ( $N$ ).
222:222- $(N-1)$	<b>Message Authentication Code / Authentication Key (MAC/Key):</b> Size is dependent on authentication method reported in the Identify Controller data structure (e.g., SHA-256 key is 32 bytes (refer to RFC 6234)).
223	<b>RPMB Target (RBT):</b> Indicates which RPMB this Request/Response is targeted for. Values 0-6 are supported. If the value in this field is not equal to the NVMe Security Specific Field (NSSF) in the Security Send or Security Receive command, then the controller shall return an error of Invalid Field in Command for the Security Send or Security Receive command.
239:224	<b>Nonce (NCE):</b> Random number generated by the host for the requests and copied to the response by the RPMB target.
243:240	<b>Write Counter (WCNTR):</b> Total amount of successfully authenticated data write requests.
247:244	<b>Address (ADDR):</b> Starting address of data to be programmed to or read from the RPMB.
251:248	<b>Sector Count (SCNT):</b> Number of sectors (512 bytes) requested to be read or written.
253:252	<b>Result (RSLT):</b> Defined in Figure 635. Note: The Result field is not needed for Requests.
255:254	<b>Request or Response Message (RRM):</b> Defined in Figure 634.
$(M-1)+256:256$	<b>Data (DATA):</b> Data to be written or read by signed access where $M = 512 * \text{Sector Count}$ . This field is optional.

Security Send and Security Receive commands are used to encapsulate and deliver data packets of any security protocol between the host and controller without interpreting, dis-assembling or re-assembling the data packets for delivery. Security Send and Security Receive commands used for RPMB access are populated with the RPMB Data Frame(s) defined in Figure 637. The controller shall not return successful completion of a Security Send or Security Receive command for RPMB access until the requested RPMB Request/Response Message Type indicated is completed. The Security Protocol used for RPMB is defined in section 5.1.23.3.

#### 8.1.21.1 Authentication Method

A controller supports one Authentication Method as indicated in the Identify Controller data structure.

If the Authentication Method supported is HMAC SHA-256 (refer to RFC 6234), then the message authentication code (MAC) is calculated using HMAC SHA-256 as defined in RFC 6234. The key used to

generate a MAC using HMAC SHA-256 is the 256-bit Authentication Key stored in the controller for the selected RPMB target. The HMAC SHA-256 calculation takes as input a key and a message. Input to the MAC calculation is the concatenation of the fields in the RPMB Data Frame (request or response) excluding stuff bytes and the MAC itself – i.e., bytes [223:255] and Data of the frame in that order.

### 8.1.21.2 RPMB Operations

The host sends a Request Message Type to the controller to request an operation by the controller or to deliver data to be written into the RPMB memory block. To deliver a Request Message Type, the host uses the Security Send command. If the data to be delivered to the controller is more than reported in Identify Controller data structure, the host sends multiple Security Send commands to transfer the entire data.

The host sends a Response Message Type to the controller to read the result of a previous operation request, to read the Write Counter, or to read data from the RPMB memory block. To deliver a Response Message Type, the host uses the Security Receive command. If the data to be read from the controller is more than reported in Identify Controller data structure, the host sends multiple Security Receive commands to transfer the entire data.

#### 8.1.21.2.1 Authentication Key Programming

Authentication Key programming is initiated by a Security Send command to program the Authentication Key to the specified RPMB target, followed by a subsequent Security Send command to request the result, and lastly, the host issues a Security Receive command to retrieve the result.

**Figure 638: RPMB – Authentication Key Data Flow**

Command	Bytes in Command	Field Name	Value	Objective
Security Send 1	<b>Data populated by the host and sent to the controller</b>			Send Authentication Key to be Programmed to the controller
	222-N:00	Stuff Bytes	0...00h	
	222:222-(N-1)	MAC/Key	<i>Key to be programmed</i>	
	223	RPMB Target	<i>RPMB target to access</i>	
	239:224	Nonce	0...00h	
	243:240	Write Counter	00000000h	
	247:244	Address	00000000h	
	251:248	Sector Count	00000000h	
	253:252	Result	0000h	
255:254	Request/Response	0001h ( <i>Request</i> )		
Security Send 2	<b>Data populated by the host and sent to the controller</b>			Request Result of Key Programming
	222-N:00	Stuff Bytes	0...00h	
	222:222-(N-1)	MAC/Key	0...00h	
	223	RPMB Target	<i>RPMB target to access</i>	
	239:224	Nonce	0...00h	
	243:240	Write Counter	00000000h	
	247:244	Address	00000000h	
	251:248	Sector Count	00000000h	
	253:252	Result	0000h	
255:254	Request/Response	0005h ( <i>Request</i> )		
Security Receive 1	<b>Data populated by the controller and returned to the host</b>			Retrieve the Key Programming Result
	222-N:00	Stuff Bytes	0...00h	
	222:222-(N-1)	MAC/Key	0...00h	
	223	RPMB Target	<i>RPMB target response was sent from</i>	
	239:224	Nonce	0...00h	
	243:240	Write Counter	00000000h	
247:244	Address	00000000h		

**Figure 638: RPMB – Authentication Key Data Flow**

Command	Bytes in Command	Field Name	Value	Objective
	251:248	Sector Count	00000000h	
	253:252	Result	<i>Result Code</i>	
	255:254	Request/Response	0100h ( <i>Response</i> )	

### 8.1.21.2.2 Read Write Counter Value

The Read Write Counter Value sequence is initiated by a Security Send command to request the Write Counter value, followed by a Security Receive command to retrieve the Write Counter result.

**Figure 639: RPMB – Read Write Counter Value Flow**

Command	Bytes in Command	Field Name	Value	Objective
Security Send 1	Data populated by the host and sent to the controller			Request Write Counter Read
	222-N:00	Stuff Bytes	0...00h	
	222:222-(N-1)	MAC/Key	0...00h	
	223	RPMB Target	<i>RPMB target to access</i>	
	239:224	Nonce	<i>Nonce generated by the host</i>	
	243:240	Write Counter	00000000h	
	247:244	Address	00000000h	
	251:248	Sector Count	00000000h	
	253:252	Result	0000h	
255:254	Request/Response	0002h ( <i>Request</i> )		
Security Receive 1	Data populated by the controller and returned to the host			Retrieve Write Counter Result
	222-N:00	Stuff Bytes	0...00h	
	222:222-(N-1)	MAC/Key	<i>MAC generated by the controller</i>	
	223	RPMB Target	<i>RPMB target response was sent from</i>	
	239:224	Nonce	<i>Copy of the Nonce generated by the host</i>	
	243:240	Write Counter	<i>Current Write Counter value</i>	
	247:244	Address	00000000h	
	251:248	Sector Count	00000000h	
	253:252	Result	<i>Result Code</i>	
255:254	Request/Response	0200h ( <i>Response</i> )		

### 8.1.21.2.3 Authenticated Data Write

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

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

After checking the Write Counter is not expired, the Address is checked. If there is an error in the Address (e.g., out of range), then the result is set to 0004h (address failure) and no data is written to the RPMB data area.

After checking the Address is valid, the controller calculates the MAC (refer to section 8.1.21.1) and compares this with the MAC in the request. If the MAC in the request and the calculated MAC are different, then the controller sets the result to 0002h (authentication failure) and no data is written to the RPMB data area.

If the MAC in the request and the calculated MAC are equal, then the controller compares the Write Counter in the request with the Write Counter stored in the controller. If the counters are different, then the controller sets the result to 0003h (counter failure) and no data is written to the RPMB data area.

If the MAC and Write Counter comparisons are successful, then the write request is authenticated. The Data from the request is written to the Address indicated in the request and the Write Counter is incremented by one.

If the write fails, then the returned result is 0005h (write failure). If another error occurs during the write procedure, then the returned result is 0001h (general failure).

The controller returns a successful completion for the Security Send command when the Authenticated Data Write operation is completed regardless of whether the Authenticated Data Write was successful or not.

The success of programming the data should be checked by the host by reading the result property of the RPMB:

- 1) The host initiates the Authenticated Data Write verification process by issuing a Security Send command with delivery of a RPMB data frame containing the Request Message Type = 0005h;
- 2) The controller returns a successful completion of the Security Send command when the verification result is ready for retrieval;
- 3) The host should then retrieve the verification result by issuing a Security Receive command; and
- 4) The controller returns a successful completion of the Security Receive command and returns the RPMB data frame containing the Response Message Type = 0300h, the incremented counter value, the data address, the MAC and result of the data programming operation.

**Figure 640: RPMB – Authenticated Data Write Flow**

Command	Bytes in Command	Field Name	Value	Objective
Security Send 1	<b>Data populated by the host and sent to the controller</b>			Program request data
	222-N:00	Stuff Bytes	0...00h	
	222:222-(N-1)	MAC/Key	MAC generated by the host	
	223	RPMB Target	RPMB target to access	
	239:224	Nonce	0...00h	
	243:240	Write Counter	Current Write Counter value	
	247:244	Address	Address in the RPMB	
	251:248	Sector Count	Number of 512B blocks	
	253:252	Result	0000h	
	255:254	Request/Response	0003h (Request)	
(M-1)+256:256	Data	Data to be written		
Security Send 2	<b>Data populated by the host and sent to the controller</b>			Request Result of data programming
	222-N:00	Stuff Bytes	0...00h	
	222:222-(N-1)	MAC/Key	0...00h	
	223	RPMB Target	<i>RPMB target to access</i>	
	239:224	Nonce	0...00h	
	243:240	Write Counter	00000000h	
	247:244	Address	00000000h	
	251:248	Sector Count	00000000h	
	253:252	Result	0000h	
255:254	Request/Response	0005h ( <i>Request</i> )		
Security Receive 1	<b>Data populated by the controller and returned to the host</b>			Retrieve Result from data programming
	222-N:00	Stuff Bytes	0...00h	
	222:222-(N-1)	MAC/Key	<i>MAC generated by the controller</i>	
	223	RPMB Target	<i>RPMB target response was sent from</i>	
	239:224	Nonce	0...00h	
	243:240	Write Counter	<i>Incremented Write Counter value</i>	
	247:244	Address	<i>Address in RPMB</i>	
	251:248	Sector Count	00000000h	
	253:252	Result	<i>Result Code</i>	
255:254	Request/Response	0300h ( <i>Response</i> )		

**8.1.21.2.4 Authenticated Data Read**

The Authenticated Data Read sequence is initiated by a Security Send command. The RPMB data frame delivered from the host to the controller includes the Request Message Type = 0004h, Nonce, Address, and the Sector Count.

When the controller receives this RPMB Data Frame, that controller first checks the Address. If there is an error in the Address, then the result is set to 0004h (address failure) and the data read is not valid.

When the host receives a successful completion of the Security Send command from the controller, that host should send a Security Receive command to the controller to retrieve the data. The controller returns an RPMB Data Frame with Response Message Type (0400h), the Sector Count, a copy of the Nonce received in the request, the Address, the Data, the controller calculated MAC, and the Result. Note: It is the responsibility of the host to verify the MAC returned on an Authenticated Data Read Request.

If the data transfer from the addressed location in the controller fails, the returned Result is 0006h (read failure). If the Address provided in the Security Send command is not valid, then the returned Result is 0004h (address failure). If another error occurs during the read procedure, then the returned Result is 0001h (general failure).

Figure 641: RPMB – Authenticated Data Read Flow

Command	Bytes in Command	Field Name	Value	Objective
Security Send 1	<b>Data populated by the host and sent to the controller</b>			Read request Data
	222-N:00	Stuff Bytes	0...00h	
	222:222-(N-1)	MAC/Key	0...00h	
	223	RPMB Target	<i>RPMB target to access</i>	
	239:224	Nonce	<i>Nonce generated by the host</i>	
	243:240	Write Counter	00000000h	
	247:244	Address	<i>Address in RPMB</i>	
	251:248	Sector Count	<i>Number of 512B blocks</i>	
	253:252	Result	0000h	
255:254	Request/Response	0004h ( <i>Request</i> )		
Security Receive 1	<b>Data populated by the controller and returned to the host</b>			Retrieve result and data from read request
	222-N:00	Stuff Bytes	0...00h	
	222:222-(N-1)	MAC/Key	<i>MAC generated by the controller</i>	
	223	RPMB Target	<i>RPMB target response was sent from</i>	
	239:224	Nonce	<i>Copy of the Nonce generated by the host</i>	
	243:240	Write Counter	0000h	
	247:244	Address	<i>Address in RPMB</i>	
	251:248	Sector Count	<i>Number of 512B blocks</i>	
	253:252	Result	<i>Result Code</i>	
255:254	Request/Response	0400h ( <i>Response</i> )		
(M-1)+256:256	Data	<i>Data read from RPMB target</i>		

### 8.1.21.3 Authenticated Device Configuration Block Write

The Authenticated Device Configuration Block Write is initiated by a Security Send command. The RPMB Data Frame delivered from the host to the controller includes the Request Message Type = 0006h, Sector Count = 01h, MAC, Write Counter set to the current Write Counter value, and the RPMB Device Configuration Block data structure (refer to Figure 642). All other fields are cleared to 0h.

If the Write Counter has expired, then that controller sets the result to 0005h (write failure, write counter expired) and no data is written to the Device Configuration Block.

The controller calculates the MAC of Request Type, Block Count, Write Counter, Address and Data, and compares this with the MAC in the request. If the MAC in the request and the calculated MAC are different, then the controller sets the result to 0002h (authentication failure) and no data is written to the RPMB Device Configuration Block.

If the Data from the RPMB Device Configuration Block attempts to disable Boot Partition Protection, then the controller sets the result to 0008h (Invalid RPMB Device Configuration Block) and no data is written to the RPMB Device Configuration Block.

If the MAC in the request and the calculated MAC are equal, then the write request is authenticated. The Data from the request is written to the RPMB Device Configuration Block.

If any other error occurs during the write procedure, then the returned result is 0001h (general failure).

The controller returns a successful completion for the Security Send command when the Authenticated Data Write operation is completed regardless of whether the Authenticated Device Configuration Block Write was successful or not.

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

The Write Counter for the Device Configuration Block is independent of the Write Counter for RPMB target 0. Authenticated Device Configuration Block Writes do not affect the Write Counter for RPMB target 0 since the data is not part of the RPMB data area. The current value of the Write Counter for the Device Configuration Block may be read using an Authenticated Device Configuration Block Read (refer to section 8.1.21.4).

**Figure 642: RPMB – Authenticated Device Configuration Block Write Flow**

Command	Bytes in Command	Field Name	Value	Objective
Security Send 1	<b>Data populated by the host and sent to the controller</b>			Request Device Configuration Block Write
	222-N:00	Stuff Bytes	0...00h	
	222:222-(N-1)	MAC/Key	MAC generated by the host	
	223	RPMB Target	00h	
	239:224	Nonce	0...00h	
	243:240	Write Counter	Current Write Counter value	
	247:244	Address	00000000h	
	251:248	Sector Count	00000001h	
	253:252	Result	0000h	
	255:254	Request/Response	0006h (Request)	
Security Send 2	Data populated by the host and sent to the controller			Request Result of data programming
	222-N:00	Stuff Bytes	0...00h	
	222:222-(N-1)	MAC/Key	0...00h	
	223	RPMB Target	RPMB target to access	
	239:224	Nonce	0...00h	
	243:240	Write Counter	00000000h	
	247:244	Address	00000000h	
	251:248	Sector Count	00000000h	
	253:252	Result	0000h	
255:254	Request/Response	0005h (Request)		
Security Receive 1	<b>Data populated by the controller and returned to the host</b>			Retrieve Device Configuration Block Write Result
	222-N:00	Stuff Bytes	0...00h	
	222:222-(N-1)	MAC/Key	MAC generated by the controller	
	223	RPMB Target	00h	
	239:224	Nonce	0...00h	
	243:240	Write Counter	Incremented Write Counter value	
	247:244	Address	00000000h	
	251:248	Sector Count	00000000h	
	253:252	Result	Result Code	
255:254	Request/Response	0600h (Response)		

### 8.1.21.4 Authenticated Device Configuration Block Read

The Authenticated Device Configuration Block Read sequence is initiated by a Security Send command. The RPMB data frame delivered from the host to the controller includes the Nonce, Request Message Type = 0007h and the Sector Count = 01h. All other fields are cleared to 0h.

When the host receives a successful completion of the Security Send command from the controller, that host should send a Security Receive command to the controller to retrieve the data. The controller returns an RPMB Data Frame with Response Message Type (0700h), the Sector Count = 01h, a copy of the Nonce

received in the request, the RPMB Device Configuration Block Data Structure (refer to Figure 633), the MAC, the Write Counter set to the current Write Counter value, and the Result. All other fields are cleared to 0h.

The Write Counter for the Device Configuration Block is independent of the Write Counter for RPMB target 0. The controller returns the Device Configuration Block Write Counter as shown in Figure 643.

The MAC is calculated from Response Type, Nonce, Address, Data and Result fields. If the MAC calculation fails, then the returned result is 0002h (authentication failure). If another error occurs during the read procedure, then the returned Result is 0001h (general failure).

**Figure 643: RPMB – Authenticated Device Configuration Block Read Flow**

Command	Bytes in Command	Field Name	Value	Objective
Security Send 1	<b>Data populated by the host and sent to the controller</b>			Request Device Configuration Block Read
	222-N:00	Stuff Bytes	0...00h	
	222:222-(N-1)	MAC/Key	0...00h	
	223	RPMB Target	00h	
	239:224	Nonce	<i>Nonce generated by the host</i>	
	243:240	Write Counter	00000000h	
	247:244	Address	00000000h	
	251:248	Sector Count	00000001h	
	253:252	Result	0000h	
255:254	Request/Response	0007h ( <i>Request</i> )		
Security Receive 1	<b>Data populated by the controller and returned to the host</b>			Retrieve Device Configuration Block Read Result
	222-N:00	Stuff Bytes	0...00h	
	222:222-(N-1)	MAC/Key	<i>MAC generated by the controller</i>	
	223	RPMB Target	00h	
	239:224	Nonce	<i>Copy of the Nonce generated by the host</i>	
	243:240	Write Counter	<i>Current Write Counter value</i>	
	247:244	Address	00000000h	
	251:248	Sector Count	00000001h	
	253:252	Result	<i>Result Code</i>	
	255:254	Request/Response	0700h ( <i>Response</i> )	
767:256	Data	<i>RPMB Device Configuration Block data structure</i>		

### 8.1.22 Reservations

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

Reservation operation after a division event (refer to section 3.2.5.1) is described in section 3.2.5.2.

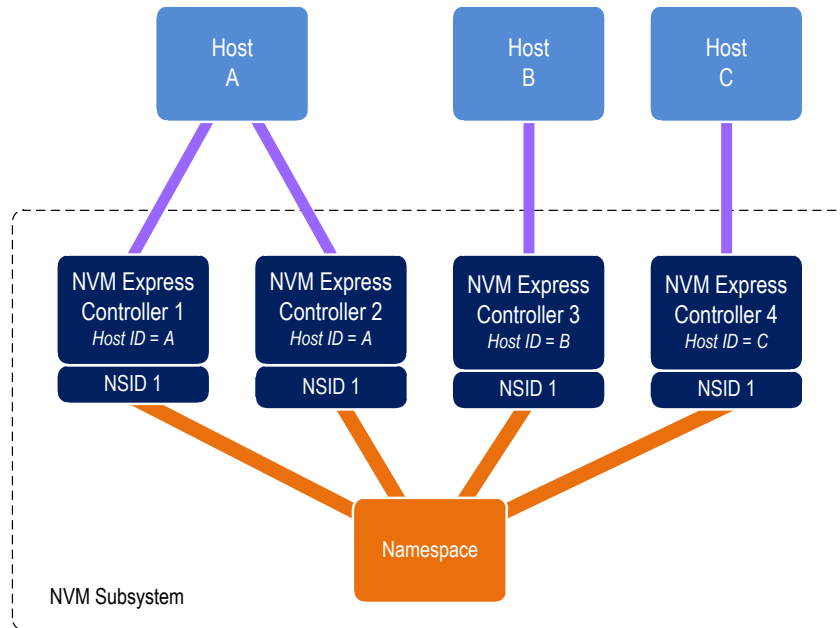
A reservation on a namespace restricts hosts access to that namespace. If a host submits a command to a namespace in the presence of a reservation and lacks sufficient rights, then the command is aborted by the controller with a status code of Reservation Conflict. If a host submits a command with the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

A command is checked for reservation conflict at the time that the controller begins processing that command. If that reservation conflict check allows the command to be performed (i.e., the host has sufficient



rights for that command with respect to existing reservations, if any), then that command shall not be subsequently aborted by the controller with a status code of Reservation Conflict (e.g., due to a subsequent reservation).

**Figure 644: Example Multi-Host System**



A reservation requires an association between a host and a namespace. As shown in Figure 644, each controller in a multi-path I/O and namespace sharing environment is associated with exactly one host. While it is possible to construct systems where two or more hosts share a single controller, such usage is outside the scope of this specification.

A host may be associated with multiple controllers. In Figure 644 host A is associated with two controllers while hosts B and C are each associated with a single controller. A host should register a non-zero Host Identifier (refer to section 5.1.25.1.28) with each controller with which that host is associated using a Set Features command (refer to section 5.1.25) prior to performing any operations associated with reservations. The Host Identifier allows the NVM subsystem to identify controllers associated with the same host and preserve reservation properties across these controllers (i.e., a host issued command has the same reservation rights no matter which controller associated with the host processes the command).

An NVM subsystem may require that the host register a non-zero Host Identifier for a host to use reservations (i.e., the RHII bit is set to '1' in the CTRATT field of the Identify Controller data structure (refer to Figure 312)). If the controller does not support reservations with a Host Identifier value of 0h and a reservation command is received from a host with a Host Identifier value of 0h, then the controller shall abort that reservation command with a status of Host Identifier Not Initialized.

If an NVM subsystem:

1. supports reservations with a Host Identifier value of 0h;
2. registrations or reservations are established by a host with a Host Identifier value of 0h; and
3. the Host Identifier is changed to a non-zero value,

then those registrations or reservations remain associated with the Host with a Host Identifier value of 0h and are not associated with the host with the non-zero Host Identifier.

If the controller does not support reservations with a Host Identifier value of 0h, reservations may have been established by hosts with non-zero Host Identifiers connected to other controllers, and commands from a host with a Host Identifier value of 0h that conflict with a reservation (refer to Figure 645 and Figure 646) are aborted by the controller with a status code of Reservation Conflict.

Support for reservations by a namespace or controller is optional. A namespace indicates support for reservations by reporting a non-zero value in the Reservation Capabilities (RESCAP) field in the Identify Namespace data structure. A controller indicates support for reservations through the Optional NVM Command Support (ONCS) field in the Identify Controller data structure (refer to Figure 312). If a host submits a command associated with reservations (i.e., Reservation Report, Reservation Register, Reservation Acquire, and Reservation Release) to a controller or a namespace that do not both support reservations, then the command is aborted by the controller with a status code of Invalid Command Opcode.

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

- Indicate support for reservations by returning a '1' in the Reservations Support (RESERVS) bit of the Optional NVM Command Support (ONCS) field in the Identify Controller data structure;
- Support the Reservation Report command (refer to section 7.8), Reservation Register command (refer to section 7.6), Reservation Acquire command (refer to section 7.5), and Reservation Release command (refer to section 7.7);
- Support the Reservation Notification log page;
- Support the Reservation Log Page Available asynchronous events;
- Support the Reservation Notification Mask Feature;
- Support the Host Identifier Feature; and
- Support the Reservation Persistence Feature.

If a controller supports dispersed namespaces and supports reservations, then the controller supports the Dispersed Namespace Reservation Support (DISNSRS) bit being set to '1' in the Reservation Report command, Reservation Register command, Reservation Acquire command, and Reservation Release command, as described in section 8.1.9.6.

If a namespace supports reservations, then the namespace shall:

- Report a non-zero value in the Reservation Capabilities (RESCAP) field in the Identify Namespace data structure;
- Support Persist Through Power Loss (PTPL) state; and
- Support sufficient resources to allow a host to successfully register a reservation key on every controller in the NVM subsystem with access to the shared namespace (i.e., a Reservation Register command shall never fail due to lack of resources).

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

### 8.1.22.1 Reservation Types

The NVM Express interface supports six types of reservations:

- Write Exclusive;
- Exclusive Access;
- Write Exclusive - Registrants Only;
- Exclusive Access - Registrants Only;
- Write Exclusive - All Registrants; and
- Exclusive Access - All Registrants.

**Figure 645: Command Behavior in the Presence of a Reservation**

Reservation Type	Reservation Holder		Registrant		Non-Registrant		Reservation Holder Definition
	Read	Write	Read	Write	Read	Write	
Write Exclusive	Y	Y	Y	N	Y	N	One Reservation Holder
Exclusive Access	Y	Y	N	N	N	N	One Reservation Holder
Write Exclusive - Registrants Only	Y	Y	Y	Y	Y	N	One Reservation Holder
Exclusive Access - Registrants Only	Y	Y	Y	Y	N	N	One Reservation Holder
Write Exclusive - All Registrants	Y	Y	Y	Y	Y	N	All Registrants are Reservation Holders
Exclusive Access - All Registrants	Y	Y	Y	Y	N	N	All Registrants are Reservation Holders

The differences between these reservation types are: the type of access that is excluded (i.e., writes or all accesses), whether registrants have the same access rights as the reservation holder, and whether registrants are also considered to be reservation holders. These differences are summarized in Figure 645 and the specific behavior for each NVM Express command is shown in Figure 646.

Reservations and registrations persist across all Controller Level Resets and all NVM Subsystem Resets except reset due to power loss. A reservation may be optionally configured to be retained across a reset due to power loss using the Persist Through Power Loss State (PTPLS). A Persist Through Power Loss State (PTPLS) is associated with each namespace that supports reservations and may be modified as a side effect of a Reservation Register command (refer to section 7.6) or a Set Features command (refer to section 5.1.25).

**Figure 646: Command Behavior in the Presence of a Reservation**

NVMe Command	Write Exclusive Reservation		Exclusive Access Reservation		Write Exclusive Registrants Only or Write Exclusive All Registrants Reservation		Exclusive Access Registrants Only or Exclusive Access All Registrants Reservation	
	Non-Registrant	Registrant	Non-Registrant	Registrant	Non-Registrant	Registrant	Non-Registrant	Registrant
<b>Read Command Group</b>								
Security Receive (Admin) I/O Command Set specific Copy Commands (source) <sup>2,3</sup> I/O Command Set specific Read Commands <sup>2</sup>	A	A	C	C	A	A	C	A

**Figure 646: Command Behavior in the Presence of a Reservation**

NVMe Command	Write Exclusive Reservation		Exclusive Access Reservation		Write Exclusive Registrants Only or Write Exclusive All Registrants Reservation		Exclusive Access Registrants Only or Exclusive Access All Registrants Reservation	
	Non-Registrant	Registrant	Non-Registrant	Registrant	Non-Registrant	Registrant	Non-Registrant	Registrant
<b>Write Command Group</b>								
Capacity Management (Admin) Flush Format NVM (Admin) Namespace Attachment (Admin) Namespace Management (Admin) Sanitize (Admin) Security Send (Admin) I/O Command Set specific Copy Commands (destination) <sup>2,3</sup> I/O Command Set specific Write Commands <sup>2</sup>	C	C	C	C	C	A	C	A
<b>Reservation Command Groups</b>								
Reservation Acquire - Acquire	C	C	C	C	C	C	C	C
Reservation Acquire - Preempt Reservation Acquire - Preempt and Abort Reservation Release	C	A	C	A	C	A	C	A
<b>All Other Commands Group</b>								
All other commands <sup>1</sup>	A	A	A	A	A	A	A	A
Key: A definition: A=Allowed, command processed normally by the controller C definition: C=Conflict, command aborted by the controller with a status code of Reservation Conflict Notes: 1. The behavior of a vendor specific command is vendor specific. 2. Refer to the applicable I/O Command Set specification 3. For an I/O Command Set specific Copy command, each source namespace is checked for reservation conflict as if accessed by a read command and the destination namespace is checked for reservation conflict as if accessed by a write command, as described in the applicable I/O command Set specification.								

### 8.1.22.2 Reservation Notifications

There are three types of reservation notifications: registration preempted, reservation released, and reservation preempted. Conditions that cause a reservation notification to occur are described in the following sections. A Reservation Notification log page is created whenever an unmasked reservation notification occurs on a namespace associated with the controller (refer to section 5.1.12.1.32). Reservation notifications may be masked from generating a Reservation Notification log page on a per reservation notification type and per namespace ID basis through the Reservation Notification Mask feature (refer to section 5.1.25.1.29). A host may use the Asynchronous Event Request command (refer to section 5.1.2) to be notified of the presence of one or more available Reservation Notification log pages (refer to section 5.1.12.1.32).

### 8.1.22.3 Registering

Prior to establishing a reservation on a namespace, a host shall become a registrant of that namespace by registering a reservation key. This reservation key may be used by the host as a means of identifying the registrant (host), authenticating the registrant, and preempting a failed or uncooperative registrant. The value of the reservation key used by a host and the method used to select its value is outside the scope of this specification.

Registering a reservation key with a namespace creates an association between a host and a namespace. A host that is a registrant of a namespace may use any controller with which that host is associated (i.e., that has the same Host Identifier, refer to section 5.1.25.1.28) to access that namespace as a registrant. Thus, a host is only required to register on a single controller to become a registrant of the namespace on all controllers in the NVM subsystem that have access to the namespace and are associated with the host. If a host attempts to access a dispersed namespace that is able to be accessed by controllers in multiple participating NVM subsystems, then that host is only required to register on a single controller in a single participating NVM subsystem to become a registrant of the dispersed namespace on all controllers in all participating NVM subsystems that have access to the dispersed namespace and are associated with the host.

A host registers a reservation key by executing a Reservation Register command (refer to section 7.6) on the namespace with the Reservation Register Action (RREGA) field cleared to 000b (i.e., Register Reservation Key) and supplying a reservation key in the New Reservation Key (NRKEY) field.

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

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

There are no restrictions on the reservation key value used by hosts with different Host Identifiers. For example, multiple hosts may all register with the same reservation key value.

A host that is a registrant of a namespace may replace the existing reservation key value for that namespace by executing a Reservation Register command on the namespace with the:

- a) RREGA field set to 010b (i.e., Replace Reservation Key);
- b) current reservation key in the Current Reservation Key (CRKEY) field; and
- c) new reservation key in the NRKEY field.

The current reservation key value shall be replaced by the new reservation key value in all controllers to which the namespace is attached that have the same Host Identifier as the Host Identifier of the controller processing the command. For dispersed namespaces, this requirement includes all participating NVM subsystems. If the contents of the CRKEY field do not match the key currently associated with the host, then the command shall be aborted with a status code of Reservation Conflict. A host may replace its reservation key without regard to its registration status or current reservation key value by setting the Ignore Existing Key (IEKEY) bit to '1' in the Reservation Register command. Replacing a reservation key has no effect on any reservation that may be held on the namespace.

### 8.1.22.4 Unregistering

A host that is a registrant of a namespace may unregister with the namespace by executing a Reservation Register command (refer to section 7.6) on the namespace with the RREGA field set to 001b (i.e., Unregister Reservation Key) and supplying its current reservation key in the CRKEY field. If the contents of the CRKEY field do not match the key currently associated with the host, then the command is aborted with a status code of Reservation Conflict. If the host is not a registrant, then the command is aborted with a status code of Reservation Conflict.

Successful completion of an unregister operation causes the host to no longer be a registrant of that namespace. A host may unregister without regard to its current reservation key value by setting the IEKEY bit to '1' in the Reservation Register command.

Unregistering by a host may cause a reservation held by the host to be released. If a host is the last remaining reservation holder (i.e., the reservation type is Write Exclusive - All Registrants or Exclusive Access - All Registrants) or is the only reservation holder, then the reservation is released when the host unregisters.

If a reservation is released and the type of the released reservation was Write Exclusive - Registrants Only or Exclusive Access - Registrants Only, then a reservation released notification occurs on all controllers associated with a registered host other than the host that issued the Reservation Register command.

#### **8.1.22.5 Acquiring a Reservation**

In order for a host to obtain a reservation on a namespace, that host shall be a registrant of that namespace. A registrant obtains a reservation by executing a Reservation Acquire command (refer to section 7.5), clearing the Reservation Acquire Action (RACQA) field to 000b (Acquire), and supplying the current reservation key associated with the host in the Current Reservation Key (CRKEY) field. The CRKEY value shall match that used by the registrant to register with the namespace. If the CRKEY value does not match, then the command is aborted with a status code of Reservation Conflict. If the host is not a registrant, then the command is aborted with a status code of Reservation Conflict.

Only one reservation is allowed at a time on a namespace. If a registrant attempts to obtain a reservation on a namespace that already has a reservation holder, then the command is aborted with a status code of Reservation Conflict. If a reservation holder attempts to obtain a reservation of a different type on a namespace for which that host already is the reservation holder, then the command is aborted with a status code of Reservation Conflict. If a reservation holder attempts to obtain a reservation of the same type on a namespace for which that host already is the reservation holder, then it is not a Reservation Conflict and the command is processed. A reservation holder may preempt a reservation to change the reservation type.

#### **8.1.22.6 Releasing a Reservation**

Only a reservation holder is able to release a reservation held on a namespace. A host should release a reservation using the following sequence:

- a) executing a Reservation Release command (refer to section 7.7);
- b) clearing the Reservation Release Action (RRELA) field to 000b (i.e., Release);
- c) setting the Reservation Type (RTYPE) field to the type of reservation being released; and
- d) supplying the current reservation key associated with the host in the Current Reservation Key (CRKEY) field. The CRKEY value shall match that used by the host to register with the namespace.

If the key value does not match, then the command is aborted with a status code of Reservation Conflict. If the RTYPE field does not match the type of the current reservation, then the command completes with a status code of Invalid Field in Command.

An attempt by a registrant to release a reservation using the Reservation Release command in the absence of a reservation held on the namespace or when the host is not the reservation holder shall cause the command to complete successfully, but shall have no effect on the controller or namespace.

When a reservation is released as a result of actions described in this section and the reservation type is not Write Exclusive or Exclusive Access, a reservation released notification occurs on all controllers in the NVM subsystem that are associated with hosts that are registrants except for controllers that are associated with the host that issued the Reservation Release command.

#### **8.1.22.7 Preempting a Reservation or Registration**

A host that is a registrant may preempt a reservation and/or registration by executing a Reservation Acquire command (refer to section 7.5), specifying:

- the Reservation Acquire Action (RACQA) field set to 001b (i.e., Preempt) or set to 010b (i.e., Preempt and Abort); and
- the current reservation key associated with the host in the Current Reservation Key (CRKEY) field.

The CRKEY value shall match that used by the registrant to register with the namespace. If the CRKEY value does not match, then the command is aborted with a status code of Reservation Conflict. The preempt actions that occur are dependent on the type of reservation held on the namespace, if any, and the value of the Preempt Reservation Key (PRKEY) field in the command. If the host is not a registrant, then the command is aborted with a status code of Reservation Conflict. The remainder of this section assumes that the host is a registrant.

If the existing reservation type is not Write Exclusive - All Registrants and not Exclusive Access - All Registrants, then the actions performed by the command depend on the value of the PRKEY field as follows:

- a) If the PRKEY field value matches the reservation key of the current reservation holder, then the following occur as an atomic operation:
    - all registrants with a matching reservation key other than the host that issued the command are unregistered;
    - the reservation is released; and
    - a new reservation is created of the type specified by the Reservation Type (RTYPE) field in the command for the host that issued the command as the reservation key holder;
- or
- b) If the PRKEY field value does not match that of the current reservation holder and is not equal to 0h, then registrants whose reservation key matches the value of the PRKEY field are unregistered. If the PRKEY field value does not match that of the current reservation holder and is equal to 0h, then the command is aborted with a status code of Invalid Field in Command.

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

- a) If the PRKEY field value is 0h, then the following occurs as an atomic operation:
    - all registrants other than the host that issued the command are unregistered;
    - the reservation is released; and
    - a new reservation is created of the type specified by the Reservation Type (RTYPE) field in the command for the host that issued the command as the reservation key holder;
- or
- b) If the PRKEY value is non-zero, then registrants whose reservation key matches the value of the PRKEY field are unregistered. If the PRKEY value is non-zero and there are no registrants whose reservation key matches the value of the PRKEY field, the controller shall abort the command with a status code of Reservation Conflict.

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

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

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

A host may abort commands as a side effect of preempting a reservation by executing a Reservation Acquire command (refer to section 7.5) and setting the RACQA field to 010b (Preempt and Abort). The

behavior of such a command is exactly the same as that described above with the RACQA field set to 001b (Preempt), with two exceptions:

- After the atomic operation changes namespace reservation and registration state, all controllers associated with any host whose reservation or registration is preempted by that atomic operation are requested to abort all commands being processed that were addressed to the specified namespace in the NSID field in the Reservation Acquire command (refer to section 3.4.4 for the definition of “being processed”); and
- Completion of the Reservation Acquire command shall not occur until all commands that are requested to be aborted are completed, regardless of whether or not each command is actually aborted.

As with the Abort command (refer to section 5.1), aborting a command as a side effect of preempting a reservation is best effort; as a command that is requested to be aborted may currently be at a point in execution where that command is no longer able to be aborted or may have already completed, when a Reservation Acquire or Abort Admin command is submitted. Although prompt execution of abort requests reduces delay in completing the Reservation Acquire command, a command which is requested to be aborted shall either be aborted or otherwise completed before the completion of the Reservation Acquire command.

When a registrant is unregistered as a result of actions described in this section, then a registration preempted notification occurs on all controllers associated with a host that was unregistered other than the host that issued the Reservation Acquire command.

When the type of reservation held on a namespace changes as a result of actions described in this section, then a reservation released notification occurs on all controllers associated with hosts that remain registrants of the namespace except the host that issued the Reservation Acquire command.

#### **8.1.22.8 Clearing a Reservation**

A host that is a registrant may clear a reservation (i.e., force the release of a reservation held on the namespace and unregister all registrants) by:

- a) executing a Reservation Release command (refer to section 7.7);
- b) setting the Reservation Release Action (RRELA) field to 001b (i.e., Clear); and
- c) supplying the current reservation key associated with the host in the Current Reservation Key (CRKEY) field.

If the value in the CRKEY field does not match the value used by the host to register with the namespace, then the command shall be aborted with a status code of Reservation Conflict. If the host is not a registrant, then the command is aborted with a status code of Reservation Conflict. When a command to clear a reservation is executed the following occur as an atomic operation: the reservation held on the namespace, if any, is released, and all registrants are unregistered from the namespace.

A reservation preempted notification occurs on all controllers in the NVM subsystem that are associated with hosts that have their registrations removed as a result of actions taken in this section except those associated with the host that issued the Reservation Release command.

#### **8.1.22.9 Reporting Reservation Status**

A host may determine the current reservation status associated with a namespace by executing a Reservation Report command (refer to section 7.8).

### **8.1.23 Rotational Media**

Rotational media has different operational, endurance and performance characteristics than non-rotational media (e.g., NAND). Rotational media utilizes electromechanical methods for accessing data.

Rotational media contains one or more spinning platters containing the media, and one or more actuators that provide physical access to the data on that media (e.g., a hard disk drive or a CD-ROM).

A controller that supports namespaces that store user data on rotational media shall:



- a) set the Rotational Media bit to '1' in the NSFEAT field of the I/O Command Set Independent Identify Namespace data structure (refer to the NVM Command Set Specification) for any namespace that stores data on rotational media;
- b) support the Rotational Media Information log page (refer to section 5.1.12.1.22);
- c) support the Spinup Control feature (refer to section 5.1.25.1.18);
- d) support Endurance Groups (refer to section 3.2.3); and
- e) set the EG Rotational Media bit to '1' in the EGFEAT field in the Endurance Group Information log page for each Endurance Group that stores data on rotational media.

If a namespace that stores data on rotational media is attached to a controller, and the spindle used by that namespace is not spinning, then that controller shall be in a non-operational power state (i.e., NOPS is set to '1', refer to Figure 313).

If:

- a) a domain contains an Endurance Group that stores data on rotational media;
- b) that domain processes an NVM Subsystem Reset; and
- c) the Spinup Control feature (refer to section 5.1.25.1.18) is:
  - a. disabled, then initial spinup for all such Endurance Groups in that domain shall be initiated; and
  - b. enabled, then initial spinup for all such Endurance Groups in that domain shall be inhibited during processing of the NVM Subsystem Reset until any controller within that domain processes a Set Features (Power Management) command that specifies an operational power state.

If the PCIe transport is used for a controller, then the PCIe Slot Power Control feature may affect the power states supported (refer to the PCI Express Base Specification).

#### **8.1.24 Sanitize Operations**

A sanitize operation alters all user data in the sanitization target such that recovery of any previous user data from any cache, the non-volatile storage media, or any Controller Memory Buffer is not possible. It is implementation specific whether Submission Queues and Completion Queues within a Controller Memory Buffer are altered by a sanitize operation; all other data stored in all Controller Memory Buffers is altered by a sanitize operation. If a portion of the user data was not altered and the sanitize operation completed successfully, then the NVM subsystem shall ensure permanent inaccessibility of that portion of the media allocated for user data for any future use within the NVM subsystem (e.g., retrieval from NVM media, caches, or any Controller Memory Buffer) and permanent inaccessibility of that portion of the media allocated for user data via any interface to the NVM subsystem, including management interfaces as defined by the NVM Express Management Interface Specification.

##### **8.1.24.1 Elements of Sanitize Operations**

A sanitize operation consists of:

- sanitize processing, which may include:
  - deallocation of all media allocated for user data; and
  - additional media modification;
- optional verification of media allocated for user data; and
- post-verification deallocation of all media allocated for user data following media verification, if any.

Sanitize processing is performed in either restricted completion mode (i.e., in the Restricted Processing state) or in unrestricted completion mode (i.e., in the Unrestricted Processing state), as specified by the Allow Unrestricted Sanitize Exit (AUSE) bit in the Sanitize command (refer to Figure 372).

Additional media modification may be performed as part of sanitize processing (i.e., in the Restricted Processing state or the Unrestricted Processing state) to prevent commands that access media after completion of sanitize processing from encountering data integrity errors caused by that sanitize processing.

Additional media modification shall be performed if the NODMMAS field is set to 10b (refer to Figure 312) and the Sanitize command that started the sanitize operation specifies:

- the Enter Media Verification State (EMVS) bit cleared to '0'; and
- the No-Deallocate After Sanitize (NDAS) bit set to '1'.

Verification of media allocated for user data is able to be performed by the host if the Sanitize Operation State Machine is in the Media Verification state. A Block Erase sanitize operation or Crypto Erase sanitize operation may invalidate error correction codes on the media, causing subsequent reads to fail because of media errors. In the Media Verification state, reads are successful regardless of such invalid error correction codes which enables the host to perform an audit (refer to section 1.5.11) to verify that the media was sanitized. Refer to the applicable I/O Command Set specification for details. Media verification is performed if the Sanitize command that starts a sanitize operation specifies the EMVS bit set to '1' and sanitize processing completes successfully.

If a sanitize operation includes deallocation of all media allocated for user data, then that deallocation shall be performed in exactly one of the following states:

- Restricted Processing state;
- Unrestricted Processing state; or
- Post-Verification Deallocation state.

If the Sanitize command (refer to Figure 372) that starts a sanitize operation specifies:

- the Enter Media Verification State (EMVS) bit cleared to '0';
- the AUSE bit cleared to '0'; and
- the No-Deallocate After Sanitize (NDAS) bit is:
  - cleared to '0'; or
  - set to '1' and the controller encounters a condition that results in unexpected deallocation of all media allocated for user data (refer to section 5.1.25.1.15),

then deallocation of all media allocated for user data shall be performed in the Restricted Processing state. If that deallocation fails, then sanitize processing fails.

If Media Verification state is canceled (i.e., the MVCNCLD bit is set to '1') during the Restricted Processing state, then deallocation of all media allocated for user data shall be performed in the Restricted Processing state.

If the Sanitize command that starts a sanitize operation specifies:

- the Enter Media Verification State (EMVS) bit cleared to '0';
- the AUSE bit set to '1'; and
- the No-Deallocate After Sanitize (NDAS) bit is:
  - cleared to '0'; or
  - set to '1' and the controller encounters a condition that results in unexpected deallocation of all media allocated for user data (refer to section 5.1.25.1.15),

then deallocation of all media allocated for user data shall be performed in the Unrestricted Processing state. If that deallocation fails, then sanitize processing fails.

If the Media Verification state is canceled (i.e., the MVCNCLD bit is set to '1') during the Unrestricted Processing state, then deallocation of all media allocated for user data shall be performed in the Unrestricted Processing state.

In the Post-Verification Deallocation state the controller deallocates all user data.

In the Post-Verification Deallocation state, if the controller:

- successfully completes deallocating all media allocated for user data, then the sanitization target enters the Idle state; or

- fails to deallocate all media allocated for user data, then the sanitization target enters the Restricted Failure state or the Unrestricted Failure state, as described in section 8.1.24.3.7.

A Controller Level Reset may cause the sanitize operation not to include the Media Verification state and the Post-Verification Deallocation state, as described in section 8.1.24.3.

#### 8.1.24.2 Sanitize Operation Types and Support

The scope of a sanitize operation is all locations in the NVM subsystem that are able to contain user data, including caches, Persistent Memory Regions, and unallocated or deallocated areas of the media.

If the composition of the NVM subsystem (refer to section 3.2.5) changes (e.g., a new domain is added, or a division event occurs) and that change prevents the successful completion of a sanitize operation, then the sanitize operation shall fail.

If the composition of the NVM subsystem changes (e.g., a new domain is added, or a division event occurs) and that change prevents verification of media allocated for user data, then the Media Verification state is canceled and the MVCNCLD bit is set to '1'.

Sanitize operations do not affect the Replay Protected Memory Block, boot partitions, or other media and caches that do not contain user data. A sanitize operation also may alter log pages and features as necessary (e.g., to prevent derivation of user data from log page information or feature information). A sanitize operation is only able to be started if the NVM subsystem is not divided (refer to section 3.2.5). A sanitize operation in the Restricted Processing state, the Unrestricted Processing state, the Media Verification state, or the Post-Verification Deallocation state is not able to be aborted and continues after a Controller Level Reset, including across power cycles. Refer to Annex A for further information about sanitize operations.

The Sanitize command (refer to section 5.1.22) is used to start a sanitize operation, to recover from a previously failed sanitize operation, or to exit the Media Verification state. All sanitize operations are performed in the background (i.e., completion of the Sanitize command that starts a sanitize operation does not indicate completion of that sanitize operation). The completion of a sanitize operation and the optional transition into the Media Verification state are indicated in the Sanitize Status log page, and with:

- the Sanitize Operation Completed asynchronous event,
- the Sanitize Operation Completed With Unexpected Deallocation asynchronous event, or
- the Sanitize Operation Entered Media Verification State asynchronous event.

If the Sanitize command that started a sanitize operation was submitted to a controller's Admin Submission Queue, then the asynchronous event shall be reported only by that controller. If the Sanitize command that started a sanitize operation was submitted to a Management Endpoint (refer to the NVM Express Management Interface Specification), then the asynchronous event shall not be reported by any controller in the NVM subsystem.

The Sanitize Capabilities (SANICAP) field of the Identify Controller data structure (refer to Figure 312) indicates the sanitize operation types supported and controller attributes specific to sanitize operations.

The sanitize operation types are:

- the Block Erase sanitize operation, which alters user data with a low-level block erase method that is specific to the media for all locations on the media within the sanitization target in which user data may be stored;
- the Crypto Erase sanitize operation, which alters user data by changing the media encryption keys for all locations on the media within the sanitization target in which user data may be stored; and
- the Overwrite sanitize operation, which alters user data by writing a fixed data pattern or related patterns one or more times to all locations on the media within the sanitization target in which user data may be stored. Figure 647 defines the data pattern or patterns that are written.

Controller attributes specific to sanitize operations include:

- the No-Deallocate Modifies Media After Sanitize (NODMMAS) field, which indicates whether media is modified by the controller as part of sanitize processing that had been requested with the No-

Deallocate After Sanitize (NDAS) bit set to ‘1’ in the Sanitize command that started the sanitize operation;

- the No-Deallocate Inhibited (NDI) bit, which indicates if the controller supports the No-Deallocate After Sanitize bit in the Sanitize command; and
- the Verification Support (VERS) bit, which indicates if the controller supports the Media Verification state and the Post-Verification Deallocation state for sanitization operations that perform block erase or crypto erase.

If the NODMMAS field indicates a value of 10b in the Identify Controller data structure (refer to Figure 312) and a Sanitize command that starts a sanitize operation specifies the No-Deallocate After Sanitize (NDAS) bit set to ‘1’, then sanitize processing includes additional media modification. Refer to Annex A.3 for further information about sanitize operations and interactions with integrity circuits.

The Overwrite sanitize operation is media specific and may not be appropriate for all media types. For example, if the media is NAND, multiple pass overwrite operations may have an adverse effect on media endurance.

If the NVM subsystem supports the Key Per I/O capability (refer to section 8.1.11), then a sanitize operation shall alter all user data such that recovery of any previous user data using the KEYTAG values specified when that previous user data was written (i.e., original KEYTAG values) is infeasible for a given level of effort (refer to ISO/IEC 27040).

**Figure 647: Sanitize Operations – Overwrite Mechanism**

OIPBP <sup>1</sup>	Overwrite Pass Count <sup>1</sup>	Overwrite Pass Number	User Data except PI Metadata	Protection Information <sup>2</sup>
‘0’	All	All	Overwrite Pattern <sup>1</sup>	Each byte set to FFh
‘1’	Even	First	Inversion of Overwrite Pattern <sup>1</sup>	Each byte cleared to 00h
		Subsequent	Inversion of Overwrite Pattern <sup>1</sup> from previous pass (i.e., each bit XORed with ‘1’)	
‘1’	Odd	First	Overwrite Pattern <sup>1</sup>	Each byte set to FFh
		Subsequent	Inversion of Overwrite Pattern <sup>1</sup> from previous pass (i.e., each bit XORed with ‘1’)	
Notes:				
1. Parameters are specified in Command Dword 10 and Command Dword 11 of the corresponding Sanitize command that started the Overwrite operation. The Overwrite Invert Pattern Between Passes (OIPBP) field is defined in Command Dword 10. The Overwrite Pass Count field is defined in Command Dword 10. The Overwrite Pattern field is defined in Command Dword 11. Refer to section 5.1.22.				
2. If Protection Information is present within the metadata.				

To start a sanitize operation, the host submits a Sanitize command specifying the SANACT field set to:

- 010b (i.e., start a Block Erase type sanitize operation);
- 011b (i.e., start a Overwrite type sanitize operation); or
- 100b (i.e., start a Crypto Erase type sanitize operation).

The Sanitize command specifies other parameters, including:

- the Allow Unrestricted Sanitize Exit (AUSE) bit;
- the No-Deallocate After Sanitize (NDAS) bit; and
- the Enter Media Verification State (EMVS) 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 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 shall ignore Critical Warning(s) in the SMART / Health Information log page (e.g., read only mode) and shall attempt to complete the sanitize operation requested. Refer to section 5 for further information about restrictions on Admin commands during the processing of a Sanitize command.

Following a successful sanitize operation, the values of user data (including protection information (PI), if any, and non-PI metadata, if any) that result from an audit (refer to section 1.5.11) of the sanitization target are defined in the I/O command set specifications.

The Sanitize Status log page (refer to section 5.1.12.1.33) indicates estimated times for sanitize operations and a consistent snapshot of information about the most recently started sanitize operation, including whether a sanitize operation is in progress, the sanitize operation parameters and the status of the most recent sanitize operation. The controller shall report that a sanitize operation is in progress if:

- sanitize processing is in progress (including additional media modification, if required);
- the sanitization target is in the Media Verification state; or
- the sanitization target is in the Post-Verification Deallocation state.

If a sanitize operation is not in progress, then the Global Data Erased (GDE) bit in the log page indicates whether the sanitization target may contain any user data (i.e., whether the sanitization target has been written to since the most recent successful sanitize operation).

The Sanitize Status log page shall be:

- initialized before any controller in the NVM subsystem is ready as described in sections 3.5.3 and 3.5.4; and
- updated when any state transition occurs (refer to section 8.1.24.3).

The Sanitize Status log page is updated periodically during a sanitize operation to make progress information available to hosts.

During a sanitize operation, the host may periodically examine the Sanitize Status log page to check for progress, however, the host should limit this polling (e.g., to at most once every several minutes) to avoid interfering with the progress of the sanitize operation itself.

The Sanitize Progress (SPROG) field in the Sanitize Status log page indicates progress during states that may take long times to complete (i.e., the Restricted Processing state, the Unrestricted Processing state, and the Post-Verification Deallocation state). The SPROG field is cleared to 0h upon entry to any of those states, and while in any of those states is updated as described in Figure 291. The SPROG field shall not be modified under any conditions not explicitly permitted by this specification.

A sanitize operation completes when the sanitization target enters any of the following states (refer to section 8.1.24.3):

- the Idle state;
- the Restricted Failure state; or
- the Unrestricted Failure state.

Upon completion of a sanitize operation or upon entry to the Media Verification state, the host should read the Sanitize Status log page with the Retain Asynchronous Event bit cleared to '0' (which clears the asynchronous event, if one was generated).

If a sanitize operation fails (i.e., the sanitization target enters the Restricted Failure state or the Unrestricted Failure state), all controllers in the NVM subsystem shall abort any command not allowed during a sanitize operation with a status code of Sanitize Failed (refer to section 8.1.24.3) until a subsequent sanitize operation is started or successful recovery from the failed sanitize operation occurs. A subsequent successful sanitize operation or the Exit Failure Mode action may be used to recover from a failed sanitize operation. Refer to section 5.1.22 for recovery details.

If the Sanitize command is supported, then all controllers in the NVM subsystem shall:

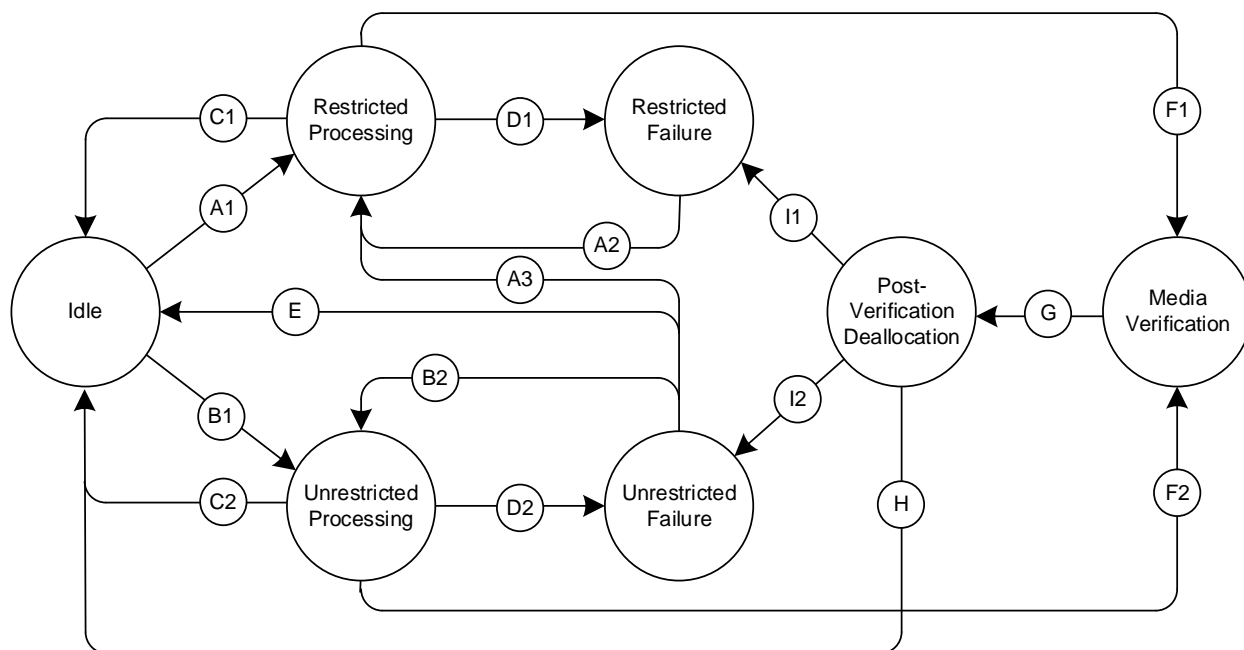
- support the Sanitize Status log page;
- support the Sanitize Operation Completed asynchronous event;
- support the Sanitize Operation Completed With Unexpected Deallocation asynchronous event, if the Sanitize Config feature is supported;
- support the Exit Failure Mode action for a Sanitize command;
- support at least one of the following sanitize operation types: Block Erase, Overwrite, or Crypto Erase;
- support the same set of sanitize operation types;
- indicate the supported sanitize operation types in the Sanitize Capabilities field in the Identify Controller data structure; and
- if the Verification Support (VERS) bit is set to '1' in the Identify Controller data structure (refer to Figure 312), support:
  - the FAILS field in the Sanitize Status log page (refer to section 5.1.12.1.33);
  - the SANS field in the Sanitize Status log page;
  - the Media Verification state;
  - the Post-Verification Deallocation state; and
  - the Sanitize Operation Entered Media Verification State asynchronous event.

The Sanitize Config Feature Identifier (refer to section 5.1.25.1.15) contains the No-Deallocate Response Mode (NODRM) bit that specifies the response of the controller to a Sanitize command specifying the No-Deallocate After Sanitize (NDAS) bit (refer to Figure 372) set to '1' if the No-Deallocate Inhibited bit is set to '1' in the Sanitize Capabilities field of the Identify Controller data structure (refer to Figure 312). In the No-Deallocate Error Response Mode, the controller aborts such Sanitize commands with a status code of Invalid Field in Command. In the No-Deallocate Warning Response Mode, the controller processes such Sanitize commands, and if a resulting sanitize operation is completed successfully, then the SOS field is set to 100b (i.e., Sanitized Unexpected Deallocate) in the Sanitize Status log page (refer to Figure 291).

#### **8.1.24.3 Sanitize Operation State Machine**

The Sanitize Operation State Machine (refer to Figure 648) defines the state of sanitization of a sanitization target. The label on each transition begins with a letter and may include a number. The letter indicates the condition causing that transition, as described in the section for each state. The number differentiates between different transitions that have the same transition condition.

**Figure 648: Sanitize Operation State Machine**



In the state transitions described in this section, asynchronous events are reported as described in section 8.1.24.2. In Completion Queue Entry Dword 0 (refer to Figure 147) for the Asynchronous Event Request command:

- a) the Log Page Identifier field shall be set to 81h (i.e., Sanitize Status log page);
- b) the Asynchronous Event Information field (refer to section 5.1.2.1) shall be set to:
  - 01h (i.e., Sanitize Operation Completed);
  - 02h (i.e., Sanitize Operation Completed With Unexpected Deallocation); or
  - 03h (i.e., Sanitize Operation Entered Media Verification State);

and

- c) the Asynchronous Event Type field shall be set to 110b (i.e., I/O Command specific status).

In each state transition described in this section, the controller shall set the Sanitize State (SANS) field to the value corresponding to the state being entered.

### 8.1.24.3.1 Idle State

In this state, no sanitize operation is in process. This state applies in the following cases:

- a) no sanitize operation has ever been performed on the NVM subsystem since the NVM subsystem was manufactured;
- b) the most recent sanitize operation on the NVM subsystem was successful; and
- c) the most recent sanitize operation failed in unrestricted completion mode (i.e., the Sanitize command specified the AUSE bit set to '1') and then the Sanitize Operation State Machine transitioned from the Unrestricted Failure state to the Idle state when any controller in the NVM subsystem performed an Exit Failure Mode action.

In this state, any controller in the NVM subsystem processing a Sanitize command specifying the Sanitize Action field set to 001b (i.e., Exit Failure Mode) shall not be considered an error.

In this state, all controllers in the NVM subsystem are permitted to resume any power management that was suspended by any prior sanitize operation.

**Figure 649: Idle State Transition Conditions**

State Transition			Transition Condition
Starting	Ending	Label <sup>1</sup>	
Idle	Restricted Processing	A1	The controller starts a sanitize operation in restricted completion mode (i.e., the Sanitize command specified the AUSE bit cleared to '0').
	Unrestricted Processing	B1	The controller starts a sanitize operation in unrestricted completion mode (i.e., the Sanitize command specified the AUSE bit set to '1').
Notes:			
1. Refer to Figure 648.			

**Transition Idle:Restricted Processing:**

The controller shall clear the:

- a) Sanitize Progress (SPROG) field to 0h; and
- b) Media Verification Canceled (MVCNCLD) bit to '0'.

**Transition Idle:Unrestricted Processing:**

The controller shall clear the:

- a) Sanitize Progress (SPROG) field to 0h; and
- b) Media Verification Canceled (MVCNCLD) bit to '0'.

**8.1.24.3.2 Restricted Processing State**

In this state, if the sanitize operation fails, then the sanitization target transitions to the Restricted Failure state.

In this state:

- a) the controller shall set the Sanitize Progress (SPROG) field as described in Figure 291;
- b) the controller shall perform additional media modification, if required, as described in section 8.1.24.1;
- c) the controller should deallocate all media allocated for user data, if permitted, as described in section 8.1.24.1;
- d) if a change in the composition of the NVM subsystem occurs then the MVCNCLD bit shall be set to '1';
- e) if a Controller Level Reset of any controller in the NVM subsystem occurs that is caused by:
  - a transport-specific reset type (refer to the applicable NVMe Transport specification); or
  - an NVM Subsystem Reset,
then the MVCNCLD bit shall be set to '1'; and
- f) all controllers and Management Endpoints in the NVM subsystem shall process commands as described in section 8.1.24.4.



**Figure 650: Restricted Processing State Transition Conditions**

State Transition			Transition Condition
Starting	Ending	Label <sup>1</sup>	
Restricted Processing	Idle	C1	Sanitize processing completes successfully and the EMVS bit was: a) cleared to '0' in the Sanitize command that started the sanitize operation; or b) set to '1' in the Sanitize command that started the sanitize operation, the MVCNCLD bit is set to '1', and deallocation of all media allocated for user data completes successfully.
	Restricted Failure	D1	Sanitize processing fails.
	Media Verification	F1	The EMVS bit was set to '1' in the Sanitize command that started the sanitize operation, the sanitize processing completes successfully, and the MVCNCLD bit is cleared to '0'.
Notes:			
1. Refer to Figure 648.			

**Transition Restricted Processing:Idle:**

The controller shall:

- a) report the Sanitize Operation Completed asynchronous event or the Sanitize Operation Completed With Unexpected Deallocation asynchronous event as described in section 8.1.24.2.

**Transition Restricted Processing:Restricted Failure:**

The controller shall:

- a) set the FAILS field to 1h (i.e., Restricted Processing state); and
- b) report the Sanitize Operation Completed asynchronous event or the Sanitize Operation Completed With Unexpected Deallocation asynchronous event as described in section 8.1.24.2.

**Transition Restricted Processing:Media Verification:**

The controller shall:

- a) report the Sanitize Operation Entered Media Verification State asynchronous event as described in section 8.1.24.2.

**8.1.24.3.3 Restricted Failure State**

This state is entered if sanitize processing was performed in the Restricted Processing state and:

- a) sanitize processing failed; or
- b) a failure occurred during deallocation of media allocated for user data in the Post-Verification Deallocation state.

In this state:

- a) all controllers and Management Endpoints in the NVM subsystem shall process commands as described in section 8.1.24.4;
- b) failure recovery requires the host to issue a subsequent Sanitize command specifying the AUSE bit cleared to '0' (i.e., restricted completion mode);
- c) all controllers in the NVM subsystem shall abort a Sanitize command specifying:
  - the SANACT field set to 001b (i.e., Exit Failure Mode), with a status code of Sanitize Failed;
  - the SANACT field set to 101b (i.e., Exit Media Verification State), with a status code of Invalid Field in Command; or
  - the USE bit set to '1' (i.e., unrestricted completion mode), with a status code of Sanitize Failed;

and

- d) the Persistent Memory Region shall behave as described in section 8.1.24.4.

**Figure 651: Restricted Failure State Transition Conditions**

State Transition			Transition Condition
Starting	Ending	Label 1	
Restricted Failure	Restricted Processing	A2	The controller starts a sanitize operation in restricted completion mode (i.e., the Sanitize command specified the AUSE bit cleared to '0').
Notes:			
1. Refer to Figure 648.			

**Transition Restricted Failure:Restricted Processing:**

The controller shall start a sanitize operation in the Restricted Processing state. The controller shall:

- a) clear the Sanitize Progress (SPROG) field to 0h; and
- b) clear the Media Verification Canceled (MVCNCLD) bit to '0'.

**8.1.24.3.4 Unrestricted Processing State**

In this state, if the sanitize operation fails, then the sanitization target transitions to the Unrestricted Failure state, from which it is able to transition to the Idle state without successful sanitization.

In this state:

- a) the controller shall set the Sanitize Progress (SPROG) field as described in Figure 291;
- b) the controller shall perform additional media modification, if required, as described in section 8.1.24.1;
- c) the controller should deallocate all media allocated for user data, if permitted, as described in section 8.1.24.1;
- d) if a change in the composition of the NVM subsystem occurs, then the MVCNCLD bit shall be set to '1';
- e) if a Controller Level Reset of any controller in the NVM subsystem occurs that is caused by:
  - a transport-specific reset type (refer to the applicable NVMe Transport specification); or
  - an NVM Subsystem Reset,
 then the MVCNCLD bit shall be set to '1'; and
- f) all controllers and Management Endpoints in the NVM subsystem shall process commands as described in section 8.1.24.4.

**Figure 652: Unrestricted Processing State Transition Conditions**

State Transition			Transition Condition
Starting	Ending	Label <sup>1</sup>	
Unrestricted Processing	Idle	C2	The sanitize processing completes successfully and the EMVS bit was: <ul style="list-style-type: none"> <li>a) cleared to '0' in the Sanitize command that started the sanitize operation; or</li> <li>b) set to '1' in the Sanitize command that started the sanitize operation, the MVCNCLD bit is set to '1', and deallocation of all media allocated for user data completes successfully.</li> </ul>
	Unrestricted Failure	D2	The sanitize processing fails.
	Media Verification	F2	The EMVS bit was set to '1' in the Sanitize command that started the sanitize operation, the sanitize processing completes successfully, and the MVCNCLD bit is cleared to '0'.
Notes:			
1. Refer to Figure 648.			

**Transition Unrestricted Processing:Idle:**

The controller shall:

- a) report the Sanitize Operation Completed asynchronous event or the Sanitize Operation Completed With Unexpected Deallocation asynchronous event as described in section 8.1.24.2.

**Transition Unrestricted Processing:Unrestricted Failure:**

The controller shall:

- a) set the FAILS field to 3h (i.e., Unrestricted Processing state); and
- b) report the Sanitize Operation Completed asynchronous event or the Sanitize Operation Completed With Unexpected Deallocation asynchronous event as described in section 8.1.24.2.

**Transition Unrestricted Processing:Media Verification:**

The controller shall:

- a) report the Sanitize Operation Entered Media Verification State asynchronous event as described in section 8.1.24.2.

**8.1.24.3.5 Unrestricted Failure State**

This state is entered if sanitize processing was performed in the Unrestricted Processing state and:

- a) sanitize processing failed; or
- b) a failure occurred during deallocation of media allocated for user data in the Post-Verification Deallocation state.

In this state:

- a) all controllers and Management Endpoints in the NVM subsystem shall process commands as described in section 8.1.24.4;
- b) all controllers in the NVM subsystem shall abort a Sanitize command specifying the SANACT field set to 101b (i.e., Exit Media Verification State) with a status code of Invalid Field in Command;
- c) all controllers in the NVM subsystem shall abort a Sanitize command specifying the SANACT field set to a value other than:
  - 001b (i.e., Exit Failure Mode);
  - 010b (i.e., Start a Block Erase sanitize operation);
  - 011b (i.e., Start an Overwrite sanitize operation); or
  - 100b (i.e., Start a Crypto Erase sanitize operation),

with a status code of Sanitize Failed; and

- d) the Persistent Memory Region shall behave as described in section 8.1.24.4.

**Figure 653: Unrestricted Failure State Transition Conditions**

State Transition			Transition Condition
Starting	Ending	Label <sup>1</sup>	
Unrestricted Failure	Restricted Processing	A3	The controller starts a sanitize operation in restricted completion mode (i.e., the Sanitize command specified the AUSE bit cleared to '0').
	Unrestricted Processing	B2	The controller starts a sanitize operation in unrestricted completion mode (i.e., the Sanitize command specified the AUSE bit set to '1').
	Idle	E	Any controller in the NVM subsystem performs an Exit Failure Mode action.
Notes:			
1. Refer to Figure 648.			

**Transition Unrestricted Failure:Restricted Processing:**

The controller shall start a sanitize operation in the Restricted Processing state. The controller shall:

- a) clear the Sanitize Progress (SPROG) field to 0h; and
- b) clear the Media Verification Canceled (MVCNCLD) bit to '0'.

**Transition Unrestricted Failure:Unrestricted Processing:**

The controller shall start a sanitize operation in the Unrestricted Processing state. The controller shall:

- a) clear the Sanitize Progress (SPROG) field to 0h; and
- b) clear the Media Verification Canceled (MVCNCLD) bit to '0'.

**Transition Unrestricted Failure:Idle:**

If any controller in the NVM subsystem performs an Exit Failure Mode action, then the controller shall recover from the sanitization failure by transitioning the Sanitize Operation State Machine to the Idle state and shall complete the Sanitize command that specified the Exit Failure Mode action with a status code of Successful Completion.

**8.1.24.3.6 Media Verification State**

In this state, the sanitize processing completed successfully, and all media allocated for user data in the sanitization target is readable by the host for purposes of verifying sanitization.

In this state:

- a) the Sanitize Operation State Machine shall transition to the Post-Verification Deallocation state if any controller in the NVM subsystem performs an Exit Media Verification State action;
- b) all controllers in the NVM subsystem shall abort a Sanitize command specifying the SANACT field not set to 101b (i.e., Exit Media Verification State) with a status code of Invalid Field in Command; and
- c) all controllers and Management Endpoints in the NVM subsystem shall process commands as described in section 8.1.24.4, with exceptions as described in the appropriate I/O command set specification (e.g., the Read command in the NVM Command Set is processed as described in the Media Verification section of the NVM Command Set Specification).

**Figure 654: Media Verification State Transition Conditions**

State Transition			Transition Condition
Starting	Ending	Label <sup>1</sup>	
Media Verification	Post-Verification Deallocation	G	Either: <ul style="list-style-type: none"> <li>a) any controller in the NVM subsystem performs an Exit Media Verification State action;</li> <li>b) an NVM Subsystem Reset occurs in any domain in the NVM subsystem;</li> <li>c) a Controller Level Reset caused by a transport-specific reset type (refer to the applicable NVMe Transport specification) of any controller in the NVM subsystem occurs; or</li> <li>d) a change in the composition of the NVM subsystem prevents media verification.</li> </ul>
Notes:			
1. Refer to Figure 648.			

**Transition Media Verification:Post-Verification Deallocation:**

The controller shall:

- a) clear the Sanitize Progress (SPROG) field to 0h;
- b) set the Media Verification Canceled (MVCNCLD) bit to '1', if any controller in the NVM subsystem processes a Controller Level Reset caused by:
  - an NVM Subsystem Reset; or
  - a transport-specific reset type (refer to the applicable NVMe Transport specification), if any;
- c) set the Media Verification Canceled (MVCNCLD) bit to '1', if a change in the composition of the NVM subsystem prevents media verification; and
- d) complete the Sanitize command with a status code of Successful Completion, if any controller in the NVM subsystem performs an Exit Media Verification State action.

**8.1.24.3.7 Post-Verification Deallocation State**

In this state:

- a) the controller shall deallocate all media allocated for user data in the sanitization target;
- b) the Sanitize Progress (SPROG) field shall be set as described in Figure 291; and
- c) all controllers and Management Endpoints in the NVM subsystem shall process commands as described in section 8.1.24.4.

**Figure 655: Post-Verification Deallocation state Transition Conditions**

State Transition			Transition Condition
Starting	Ending	Label <sup>1</sup>	
Post-Verification Deallocation	Idle	H	The controller completes deallocation of all media allocated for user data.
	Restricted Failure	I1	The sanitize operation was started by a Sanitize command specifying the AUSE bit cleared to '0' (i.e., restricted completion mode), and a failure occurs during deallocation of all media allocated for user data.
	Unrestricted Failure	I2	The sanitize operation was started by a Sanitize command specifying the AUSE bit set to '1' (i.e., unrestricted completion mode), and a failure occurs during deallocation of all media allocated for user data.
Notes:			
1. Refer to Figure 648.			

**Transition Post-Verification Deallocation:Idle:**

The controller shall:

- a) report the Sanitize Operation Completed asynchronous event as described in section 8.1.24.2.

**Transition Post-Verification Deallocation:Restricted Failure:**

The controller shall:

- a) report the Sanitize Operation Completed asynchronous event as described in section 8.1.24.2; and
- b) set the FAILS field to 6h (i.e., Post-Verification Deallocation state).

**Transition Post-Verification Deallocation:Unrestricted Failure:**

The controller shall:

- a) report the Sanitize Operation Completed asynchronous event as described in section 8.1.24.2; and
- b) set the FAILS field to 6h (i.e., Post-Verification Deallocation state).

**8.1.24.4 Sanitize Operation Restrictions**

In the following states:

- Restricted Processing;
- Restricted Failure;
- Unrestricted Processing;
- Unrestricted Failure;
- Media Verification; and
- Post-Verification Deallocation,

all enabled controllers in the NVM subsystem are restricted to performing only a limited set of actions.

When a sanitize operation starts on any controller (i.e., a transition into the Restricted Processing state occurs or a transition into the Unrestricted Processing state occurs), all controllers in the NVM subsystem shall:

- clear all of the following outstanding asynchronous events:
  - Sanitize Operation Completed asynchronous event, if any;
  - Sanitize Operation Completed With Unexpected Deallocation asynchronous event, if any; and
  - Sanitize Operation Entered Media Verification State asynchronous event, if any;
- update the Sanitize Status log page (refer to section 5.1.12.1.33);
- abort any command (submitted or in progress) not allowed during a sanitize operation (refer to Figure 142) with a status code of Sanitize In Progress, unless otherwise specified;
- abort device self-test operations in progress;
- suspend autonomous power state management activities as described in section 8.1.17.2; and
- release stream identifiers for any open streams.

If a sanitize operation is in any of the following states (i.e., is in progress):

- Restricted Processing;
- Unrestricted Processing;
- Media Verification; or
- Post-Verification Deallocation,

then for each controller in the NVM subsystem:

- all I/O commands other than a Flush command shall be aborted with a status code of Sanitize In Progress, unless otherwise specified;
- processing of a Flush command is specified in section 7.2;

- any command or command option that is not explicitly permitted in Figure 142 shall be aborted with a status code of Sanitize In Progress if processed by the controller;
- 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'); and
- activation of new firmware is prohibited.

While the sanitization target is in the Restricted Failure state or the Unrestricted Failure state, then for each controller in the NVM subsystem:

- any command or command option that is not explicitly permitted in Figure 142 shall be aborted with a status code of Sanitize In Progress if processed by the controller;
- all I/O commands other than a Flush command (refer to section 7.2), shall be aborted with a status code of Sanitize Failed;
- the Sanitize command is permitted with action restrictions (refer to section 5.1.22); 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').

When a sanitize operation starts on any controller in an NVM subsystem (i.e., a transition into the Restricted Processing state occurs or a transition into the Unrestricted Processing state occurs), all Management Endpoints in the NVM subsystem shall perform the sanitize operation as described in the NVM Express Management Interface Specification.

#### **8.1.24.5 Sanitize Operation Effects on Exported NVM Subsystems**

Performing a sanitize operation on an Underlying NVM Subsystem (refer to section 8.3.3) sanitizes user data in all Underlying Namespaces contained in that NVM subsystem, including any Underlying Namespace that is associated with an Exported Namespace in any Exported NVM Subsystem (refer to section 5.3.7).

If an Exported NVM Subsystem contains an Exported Namespace that is associated with an Underlying Namespace in an Underlying NVM Subsystem for which one of the following conditions exists:

- a sanitize operation is in progress; or
- the most recent sanitize operation has failed and successful recovery from the failed sanitize operation has not occurred,

then, while that condition exists, that Exported NVM Subsystem shall enforce the I/O command sanitize operation restrictions described in section 8.1.24.4 on I/O commands that specify that Exported Namespace and may enforce additional sanitize operation restrictions described in that section.

#### **8.1.25 Submission Queue (SQ) Associations**

When Predictable Latency Mode is enabled, all I/O commands for namespaces in a given NVM Set have the same quality of service attributes and shall exhibit predictable latencies as described in section 8.1.18.

The SQ Associations capability provides hints to the controller as to which specific I/O Queues are associated with a given NVM Set. The controller uses this information to further enhance performance when Predictable Latency Mode is enabled.

The SQ Associations capability is an optional capability. Predictable Latency Mode (refer to section 8.1.18) is not dependent on the use of the SQ Associations capability.

If a controller supports SQ Associations, then the controller shall:

- indicate support for the SQ Associations capability in the Controller Attributes (CTRATT) field in the Identify Controller data structure;
- indicate support for NVM Sets in the Controller Attributes (CTRATT) field in the Identify Controller data structure; and
- indicate support for Predictable Latency Mode in the Controller Attributes (CTRATT) field in the Identify Controller data structure (refer to Figure 312).

The host enables the SQ Associations capability by creating an association between an NVM Set and a Submission Queue at the time the Submission Queue is created (e.g., with a Create I/O Submission Queue command (refer to section 5.2.2) or a Connect command (refer to section 6.3)).

For the SQ Associations capability to yield benefits, the host is required to:

- a) create an association between each Submission Queue and some NVM Set; and
- b) only issue I/O commands to Submission Queues that have an association with the NVM Set that contains the namespace associated with the Namespace Identifier specified in that I/O command.

While this capability is enabled, failure to follow the specified operating rules may impact Predictable Latency (refer to section 8.1.18).

### 8.1.26 Standard Vendor Specific Command Format

Controllers may support the standard Vendor Specific command format defined in Figure 82. 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 312; refer to the Admin Vendor Specific Command Configuration field and the I/O Command Set Vendor Specific Command Configuration field.

### 8.1.27 Telemetry

Telemetry enables manufacturers to collect internal data logs to improve the functionality and reliability of products. The telemetry data collection may be initiated by the host or by the controller. The data is returned in the Telemetry Host-Initiated log page or the Telemetry Controller-Initiated log page (refer to section 5.1.12.1.8 and 5.1.12.1.9). The data captured is vendor specific. The telemetry feature defines the mechanism to collect the vendor specific data. The controller indicates support for the telemetry log pages and for the Data Area 4 size in the Log Page Attributes (LPA) field in the Identify Controller data structure (refer to Figure 312).

An important aspect to discovering issues by collecting telemetry data is the ability to qualify distinct issues that are being collected. The ability to create a one to one mapping of issues to data collections is essential. If a one to one mapping is not established, there is the risk that several payload collections appear distinct but are actually all caused by the same issue. Conversely, a single payload collection may have payloads caused by several issues mixed together creating additional complexity in determining the root cause. As a result, flexibility in size is provided in the collection of telemetry payloads and a three phase process is typically used.

The first phase establishes that an issue exists and is best accomplished by collecting a minimum set of data to identify the issue as being distinct from other issues. Once the number of instances of an issue establish an investigation, another phase may be necessary to collect actionable information. In the second phase, a targeted collection of more in depth medium size payloads are gathered and analyzed to identify the source of the problem.

If the small or medium sized telemetry data collection provides insufficient information, a third phase may be employed to collect additional details. If the Data Area 4 Support (DA4S) bit is cleared to '0' in the Log Page Attributes field, then the third phase provides the largest and most complete payload to diagnose the issue. If the DA4S bit is set to '1' and the Extended Telemetry Data Area 4 Supported (ETDAS) field is set to 1h in the Host Behavior Support feature (refer to section 5.1.25.1.14) then a fourth phase may be employed to collect the largest and most complete payload to diagnose the issue. If Data Area 4 is created, then Data Area 3 of non-zero length shall also be created and populated as part of data collection.

There are two telemetry data logs (i.e., Telemetry Host-Initiated log page and Telemetry Controller-Initiated log page) defined. Each telemetry data log is made up of a single set of Telemetry Data Blocks. Each Telemetry Data Block is 512 bytes in size. Telemetry data is returned (refer to section 5.1.12.1.8 and section 5.1.12.1.9) in units of Telemetry Data Blocks. Each telemetry data log is segmented into:

- a) Three Telemetry Data Areas (i.e., small, medium, and large), if the DA4S bit is cleared to '0'; or



- b) Four Telemetry Data Areas (i.e., small, medium, large and extra-large) If the DA4S bit is set to '1' and the Extended Telemetry Data Area 4 Supported (ETDAS) field is set to 1h in the Host Behavior Support feature (refer to section 5.1.25.1.14).

All telemetry data areas start at Telemetry Data Block 1.

Each Telemetry Data Area shall represent the controller's internal state at the time the telemetry data was captured.

Each Telemetry Data Area is intended to capture a richer set of data to aid in resolution of issues. Telemetry Data Area 1 is intended to have a small size payload (i.e., the first phase), Telemetry Data Area 2 is intended to have a medium size payload (i.e., the second phase), and Telemetry Data Area 3 is intended to have a large size payload (i.e., the third phase). Telemetry Data Area 4 is intended to have an extra-large size payload (i.e., the fourth phase). The size of each Telemetry Data Area is vendor specific and may change on each data collection. When possible, the host should retrieve the payload for all supported Telemetry Data Areas to enable the best diagnosis of the issue(s).

The preparation, collection, and submission of telemetry data is similar for host-initiated and controller-initiated data; the primary difference is the trigger for the collection. The operational model for telemetry is:

1. The host identifies controller support for Telemetry log pages in the Identify Controller data structure;
2. The host may indicate the support for the Telemetry Host-Initiated Data Area 4 and Telemetry Controller-Initiated Data Area 4 by setting the Extended Telemetry Data Area 4 Supported (ETDAS) field to 1h in the Host Behavior Support feature (refer to section 5.1.25.1.14);
3. The host prepares an area to store telemetry data if needed;
4. To receive notification that controller-initiated telemetry data is available, the host enables Telemetry Log Notices using the Asynchronous Event Configuration feature (refer to section 5.1.25.1.5); and
5. If the host decides to collect host-initiated telemetry data or the controller signals that controller-initiated telemetry data is available:
  - a. The host reads the appropriate blocks of the Telemetry Data Area from the Telemetry Host-Initiated log page (refer to section 5.1.12.1.8) or the Telemetry Controller-Initiated log page (refer to section 5.1.12.1.9). If possible, the host should collect Telemetry Data Area 1, 2, 3, and 4. The host reads the log in 512 byte Telemetry Data Block units (i.e., a starting offset that is a multiple of 512, and a length that is a multiple of 512). The host should set the Retain Asynchronous Event bit to '1';
  - b. The host re-reads the header of the log page and ensures that the Telemetry Host-Initiated Data Generation Number field from the Telemetry Host-Initiated log page or the Telemetry Controller-Initiated Data Generation Number field in the Telemetry Controller-Initiated log page matches the original value read. If these values do not match, then the data captured is not consistent and should be re-read from the log page with the Retain Asynchronous Event bit set to '1';
  - c. If the host is reading the controller-initiated log, then the host ensures that the Telemetry Controller-Initiated Data Available field is still set to 1h after reading the appropriate blocks of the Telemetry Data Area because the Telemetry Controller-Initiated Data Available field may have been cleared to 0h by another entity while the log page was being read;
  - d. If the host is reading the Telemetry Controller-Initiated log page, then the host reads any portion of that log page with the Retain Asynchronous Event bit cleared to '0' to indicate to the controller that the host has completed reading the Telemetry Controller-Initiated log page; and
  - e. When all telemetry data has been saved, the data should be forwarded to the manufacturer of the controller.

The trigger for the collection for host-initiated data is typically a system crash, but may also be initiated during normal operation. The host proceeds with a host-initiated data collection by submitting the Get Log Page command for the Telemetry Host-Initiated log page with the Create Telemetry Host-Initiated Data bit

set to '1' in the Log Specific Parameter field. The controller should complete the command quickly (e.g., in less than one second) to avoid a user rebooting the system prior to completion of the data collection.

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

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

The controller notifies the host to collect controller-initiated data through the completion of an Asynchronous Event Request command with an Asynchronous Event Type of Notice that indicates a Telemetry Log Changed event. The host may also determine controller-initiated data is available via the Telemetry Controller-Initiated Data Available field in the Telemetry Host-Initiated or the Telemetry Controller-Initiated log pages. The host proceeds with a controller-initiated data collection by submitting the Get Log Page command for the Telemetry Controller-Initiated log page. Once the host has started reading the Telemetry Controller-Initiated log page, the controller should avoid modifying the controller-initiated data until the host has finished reading all controller-initiated data. The amount of time for the host to read the controller-initiated data is vendor specific.

Since there is only one set of controller-initiated data, the controller is responsible for prioritizing the version of the controller-initiated data that is available for the host to collect. When the controller replaces the controller-initiated data with new controller-initiated data, the controller shall increment the Telemetry Controller-Initiated Data Generation Number field. The host needs to ensure that the Telemetry Controller-Initiated Data Generation Number field has not changed between the start and completion of the controller-initiated data collection to ensure the data captured is consistent.

#### 8.1.27.1 Telemetry Data Collection Examples (Informative)

This section includes several examples of Telemetry Host-Initiated Data Areas for illustration. The same concepts apply to the Telemetry Controller-Initiated Data Areas.

If a Telemetry Host-Initiated log page has no data for collection, then the following fields are all cleared to 0h:

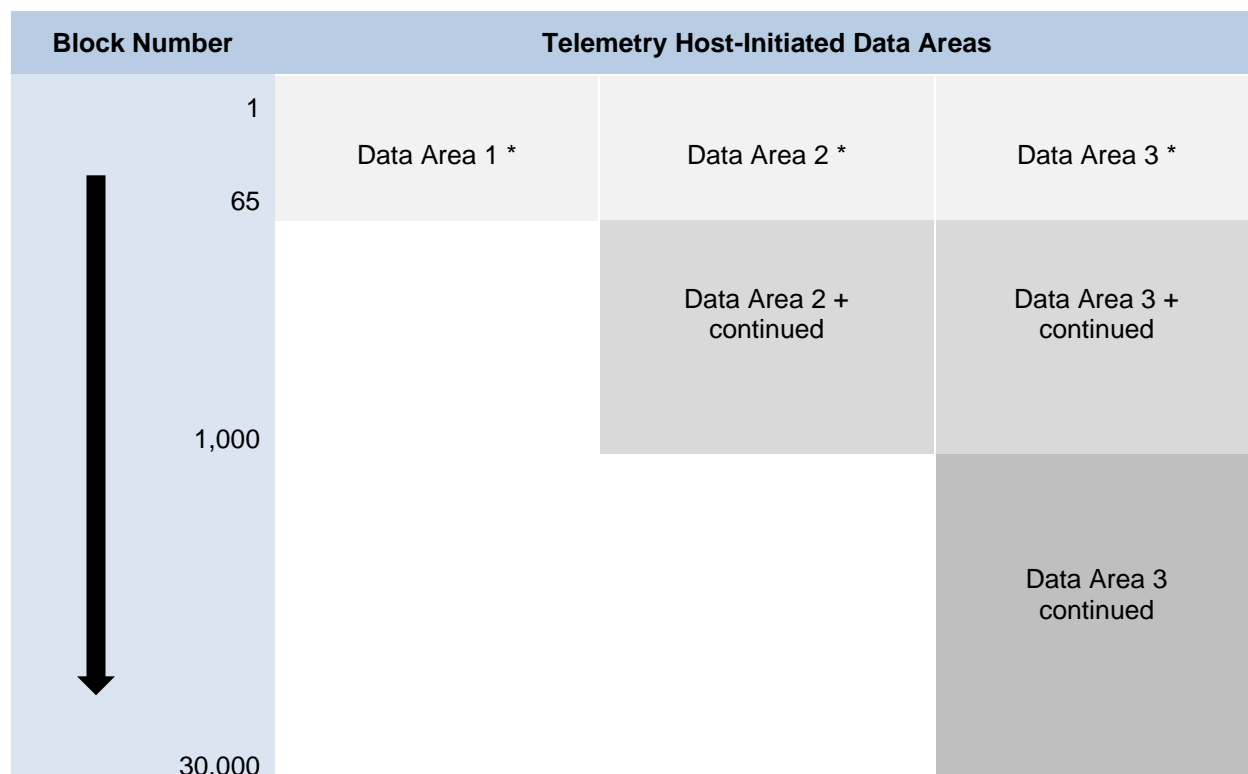
- Telemetry Host-Initiated Data Area 1 Last Block = 0;
- Telemetry Host-Initiated Data Area 2 Last Block = 0; and
- Telemetry Host-Initiated Data Area 3 Last Block = 0.

When all three telemetry data areas are populated, then the Telemetry Host-Initiated log page has different values in each of the Telemetry Host-Initiated Data Area n Last Block fields. For example, the following values correspond to the layout shown in Figure 656:

- Telemetry Host-Initiated Data Area 1 Last Block = 65;
- Telemetry Host-Initiated Data Area 2 Last Block = 1,000; and
- Telemetry Host-Initiated Data Area 3 Last Block = 30,000.

As a result of telemetry data areas being made up of a single set of Telemetry Data Blocks starting at Telemetry Data Block 1, the telemetry data contained in Telemetry Data Block 1 through Telemetry Data Block 65 of data area 1, data area 2, and data area 3 is the same. In addition, the telemetry data contained in Telemetry Data Block 66 through Telemetry Data Block 1,000 of data area 2 and data area 3 is the same.

**Figure 656: Telemetry Log Example – All Data Areas Populated**



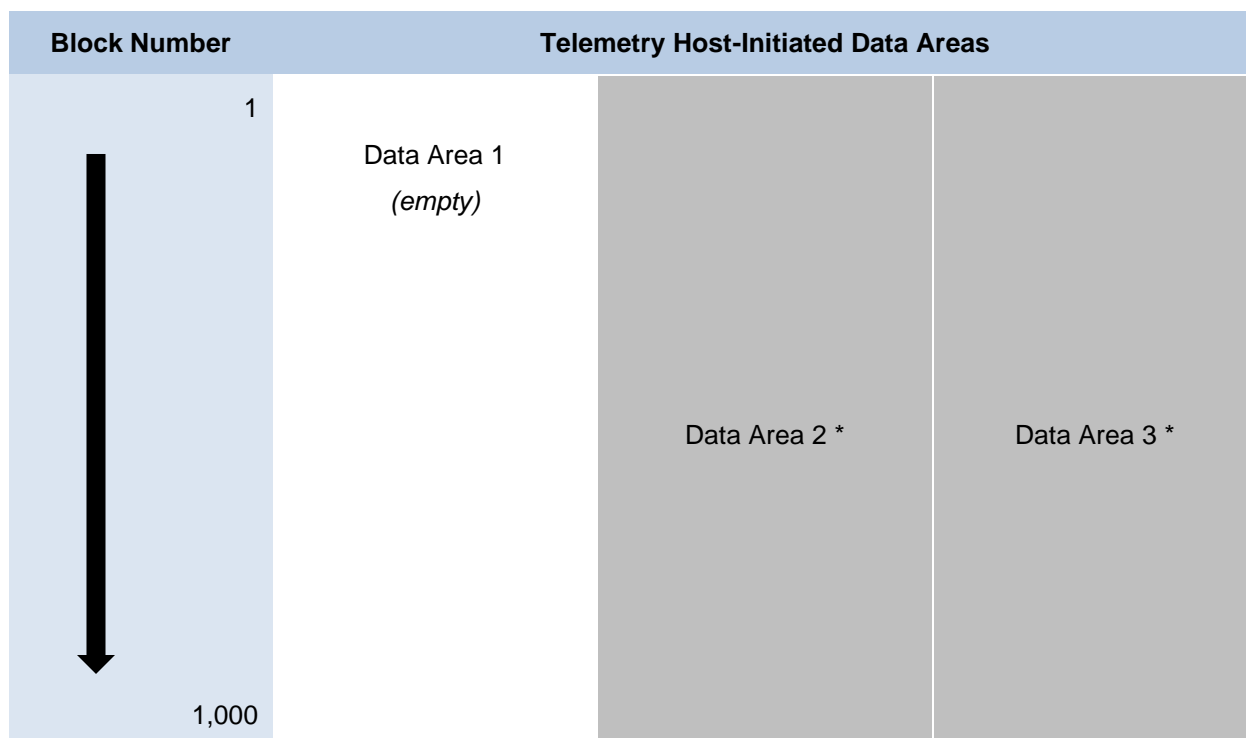
\* Data Area 1, Data Area 2, and Data Area 3 contain the same telemetry data in blocks 1 through 65.  
 + Data Area 2 and Data Area 3 contain the same telemetry data in blocks 66 through 1,000.

When only the second data area is populated, then the Telemetry Host-Initiated log page has no data in Telemetry Data Area 1 shown by having its corresponding last block value cleared to 0h, and no additional data in Telemetry Data Area 3 shown by having its corresponding last block value set to the same value as the last block value for Telemetry Data Area 2. For example, the following values correspond to the layout shown in Figure 657:

- Telemetry Host-Initiated Data Area 1 Last Block = 0;
- Telemetry Host-Initiated Data Area 2 Last Block = 1,000; and
- Telemetry Host-Initiated Data Area 3 Last Block = 1,000.

As a result of telemetry data areas being made up of a single set of Telemetry Data Blocks starting at Telemetry Data Block 1, the telemetry data contained in Telemetry Data Block 1 through Telemetry Data Block 1,000 of data area of data area 2 and data area 3 is the same.

Figure 657: Telemetry Log Example – Data Area 2 Populated



\* Data Area 2, and Data Area 3 contain the same telemetry data in blocks 1 through 1,000.

## 8.1.28 Universally Unique Identifiers (UUIDs) for Vendor Specific Information

### 8.1.28.1 UUIDs for Vendor Specific Information Introduction

Several commands send or receive information that contains fields described as Vendor Specific or that is specified by a command field containing a value in a vendor specific range. Examples include the Set Features command, which may specify a vendor specific feature identifier, and the Identify command, which may retrieve a data structure having a vendor specific area.

The vendor specific information may have different definitions (e.g., a vendor specific log page identifier with the contents of the page defined differently by different entities, such as an NVM subsystem vendor and an NVM subsystem customer). By associating each definition of the information with a UUID specified by the defining entity, a command is able to specify the particular definition of the information.

A command specifies a particular definition of the information by specifying an index into a list of UUIDs supported by the controller (refer to section 5.1.13.2.14). The NVMe Invalid UUID (refer to section 8.1.28.2) is used to replace a previously valid UUID in the UUID List (refer to Figure 320). This is done to keep the values in the list at a static index, as that index is used by the Host to access the UUID List contents.

NVM subsystem vendors and customers communicate (by means outside the scope of this specification) the UUID used for each definition of the information.

### 8.1.28.2 UUIDs for Vendor Specific Information Requirements

A UUID list is a list of non-zero UUID values, terminated by a 0h UUID value. Each non-zero UUID value may be either a valid UUID or the NVMe Invalid UUID. The NVMe Invalid UUID is the hexadecimal value FFFFFFFF\_FFFFFFFF\_7FFFFFFF\_FFFFFFFFh. A valid UUID is any non-zero value other than the NVMe Invalid UUID.

If a command supports selection of a UUID, then the UUID Selection Supported bit in the Commands Supported and Effects data structure for that command (refer to Figure 210) shall be set to '1'. If a command does not support selection of a UUID, then the UUID Selection Supported bit shall be cleared to '0'.

If the UUID Selection Supported bit is set to '1' for one or more commands, then the UUID List bit in the Controller Attributes field shall be set to '1' (refer to Figure 312), and the controller shall support reporting of a UUID List (refer to Figure 320).

If a command supports selection of a UUID, then that command contains a UUID Index field (refer to Figure 658).

**Figure 658: UUID Index Field**

Bits	Description
6:0	<b>UUID Index (UIDX):</b> If this field is set to a non-zero value, then the value of this field is the index of a UUID in the UUID List (refer to Figure 320) that is used by the command. If this field is cleared to 0h, then no UUID index is specified.

If the UUID Index field specifies a valid UUID (i.e., the UUID Index field is set to a non-zero value and the UUID at that index indicates a valid UUID) (refer to section 5.1.13.2.14), then the controller shall process the command using the vendor specific information specified by that UUID. If the UUID Index field is cleared to 0h, then the command does not specify a UUID.

If no UUID is specified by the command, then the controller shall process the command, returning vendor specific information.

The controller shall abort the command with a status code of Invalid Field in Command if:

- a) The controller does not support the UUID specified by the UUID Index for the specified information;
- b) The UUID specified by the UUID Index is cleared to 0h; or
- c) The UUID specified by the UUID Index is the NVMe Invalid UUID.

If a firmware image is activated that has a UUID List in which an entry is different from that of the previously-active firmware image, then a host that is unaware of the change may issue a command with the UUID index value for that entry. Such a command may produce unexpected results because the UUID specified by that UUID Index has changed. To avoid this, vendors should follow the following revision guidelines for UUID lists when constructing firmware images that support UUID selection:

- a) Add UUIDs that are not supported in prior firmware image revisions to the end of the UUID List in subsequent firmware image revisions;
- b) Remove UUIDs that are supported in prior firmware image revisions by replacing the UUID with the NVMe Invalid UUID in the same entry in the UUID list in subsequent firmware image revisions;
- c) Do not replace the NVMe Invalid UUID with a valid UUID in the same UUID list entry in subsequent firmware image revisions; and
- d) Do not shorten or remove the UUID list in subsequent firmware image revisions.

In these guidelines, the terms "prior" and "subsequent" refer to a linear sequence of firmware versions (e.g., based on the date and time of the construction of the downloadable firmware image).

Following these guidelines prevents the host from inadvertently specifying the wrong UUID because there is at most one valid UUID for each entry in the UUID list. Hence a command that specifies a UUID Index either specifies the intended UUID or is aborted because that entry in the UUID list is empty or contains the NVMe Invalid UUID.

The controller shall require a reset to activate a downloaded firmware image (refer to section 5.1.8) if the downloaded image reports a UUID list with at least one slot in which a valid UUID replaces the NVMe Invalid UUID or a different valid UUID in the existing image. All controllers that are affected by the UUID list change caused by activation of a downloaded firmware image shall be reset as part of activating that downloaded firmware image.

The above requirements for a reset to activate a downloaded firmware image do not require the controller to directly compare the UUID lists in the current and downloaded firmware images. For example, a vendor

could use a vendor-specific major.minor firmware image revision numbering system (e.g., 3.5, 4.1) where all downloadable firmware images with the same major revision number follow the above guidelines. In that scenario, the controller is able to meet these reset requirements by requiring a reset if the downloaded firmware image and the currently executing firmware have different major revision numbers.

### 8.1.28.3 UUIDs for Vendor Specific Information Examples

This section includes examples of the use of UUIDs to select vendor specific information.

#### 8.1.28.3.1 Vendor Specific Log Page Example

If entity C and entity V create different definitions for a vendor specific log page having the same log page identifier (e.g., D0h), then each assigns a UUID to distinguish their definition (e.g., entity V assigns UUID V and entity C assigns UUID C).

A controller supporting both definitions of the log page:

- a) Sets the UUID List bit to '1' in the CTRATT field of the Identify Controller data structure (refer to Figure 312);
- b) Sets the UUID Selection Supported bit to '1' in the Commands Supported and Effects data structure (refer to Figure 210) corresponding to the Get Log Page command; and
- c) Reports both UUID V and UUID C in the UUID list (refer to Figure 320).

A host requesting the log page defined by entity C:

- 1) Determines the index of UUID C in the UUID list;
- 2) Sets the Log Page Identifier field of the Get Log Page command to D0h; and
- 3) Sets the UUID Index field of the Get Log Page command to the index of UUID C.

A host requesting the log page defined by entity V:

- 1) Determines the index of UUID V in the UUID list;
- 2) Sets the Log Page Identifier field of the Get Log Page command to D0h; and
- 3) Sets the UUID Index field of the Get Log Page command to the index of UUID V.

A host not specifying the definition of the log page clears the UUID Index field to 0h. The selection of the log page definition returned by the controller is vendor specific (e.g., the controller may select any definition for the returned data).

#### 8.1.28.3.2 Vendor Specific Feature Example

If entity C and entity V create different definitions for a vendor specific feature having the same Feature Identifier (e.g., F1h), then each assigns a UUID to distinguish their definitions (e.g., entity V assigns UUID V and entity C assigns UUID C).

A controller supporting both definitions of the feature for the Get Features command:

- a) Sets the UUID List bit to '1' in the CTRATT field of the Identify Controller data structure (refer to Figure 312);
- b) Sets the UUID Selection Supported bit to '1' in the Commands Supported and Effects data structure (refer to Figure 210) corresponding to the Get Features command;
- c) Sets the UUID Selection Supported bit to '1' in the FID Supported and Effects log page (refer to Figure 262); and
- d) Reports both UUID V and UUID C in the UUID list (refer to Figure 320).

A host retrieving the attributes of the feature defined by entity C:

- 1) Determines the index of UUID C in the UUID list;
- 2) Sets the Feature Identifier field of the Get Features command to F1h; and
- 3) Sets the UUID Index field of the Get Features command to the index of UUID C.

A host retrieving the attributes of the feature defined by entity V:

- 1) Determines the index of UUID V in the UUID list;

- 2) Sets the Feature Identifier field of the Get Features command to F1h; and
- 3) Sets the UUID Index field of the Get Features command to the index of UUID V.

## 8.2 Memory-Based Transport Extended Capabilities (PCIe)

This section describes extended capabilities that are specific to the Memory-based transport model.

### 8.2.1 Controller Memory Buffer

The Controller Memory Buffer (CMB) is a region of general purpose read/write memory on the controller. The controller indicates support for the CMB by setting CAP.CMBS to '1'. The host indicates intent to use the CMB by setting CMBMSC.CRE to '1'. If this bit is set to '1', the controller indicates the properties of the CMB via the CMBLOC and CMBSZ properties (refer to section 3.1.4).

The CMB may be used for a variety of purposes. The controller indicates which purposes the memory may be used for by setting support flags in the CMBSZ property.

The CMB's PCI Express address range is used for external memory read and write requests to the CMB. The PCI Express base address of the CMB is defined by the PCI Base Address Register (BAR) indicated by CMBLOC.BIR, and the offset indicated by CMBLOC.OFST. The size of the CMB is indicated by CMBSZ.SZ.

The controller uses the CMB's controller address range to reference CMB with addresses supplied by the host. The PCI Express address range and the controller address range of the CMB may differ, but both ranges have the same size, and equivalent offsets within each range have a one-to-one correspondence. The host configures the controller address range via the CMBMSC property.

The host enables the CMB's controller memory space via the CMBMSC.CMSE bit. When controller memory space is enabled, if the host supplies an address referencing the CMB's controller address range, then the controller directs memory read or write requests for this address to the CMB.

When the CMB's controller memory space is disabled, the controller does not consider any host-supplied address to reference the CMB's controller address range, and memory read and write requests are directed elsewhere (e.g., to memory other than the CMB).

To prevent possible misdirection of the controller's memory requests, before host software enables the CMB's controller memory space, the host should configure the CMB's controller address range so that the addresses do not overlap any address that host software intends to use for DMA.

In versions prior to NVM Express Base Specification, Revision 1.4, for a controller that supports the CMB, the CMB's controller address range is fixed to be equal to its PCI Express address range, and the CMB's controller memory space is always enabled whenever the controller is enabled. To prevent misdirection of controller memory requests when such a controller is assigned to a virtual machine, host software (on the hypervisor or host OS) should not enable translation of the CMB's PCI Express address range and should ensure that this address range does not overlap any range of pre-translated addresses that the virtual machine may use for DMA.

A host 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).

Submission Queues in host memory require the controller to perform a PCI Express read from host memory in order to fetch the submission queue entries. Submission Queues in controller memory enable host software to directly write the entire submission queue entry to the controller's internal memory space, avoiding one read from the controller to the host. This approach reduces latency in command execution and improves efficiency in a PCI Express fabric topology that may include multiple switches. Similarly, PRP Lists or SGLs require separate fetches across the PCI Express fabric, which may be avoided by writing the PRP or SGL to the Controller Memory Buffer. Completion Queues in the Controller Memory Buffer may be used for peer to peer or other applications. For writes of small amounts of data, it may be advantageous to

have the host write the data and/or metadata to the Controller Memory Buffer rather than have the controller fetch it from host memory.

The contents of the Controller Memory Buffer are undefined as the result of:

- the CMBMSC.CMSE bit transitioning from '0' to '1';
- a Controller Reset; or
- a Function Level Reset.

Host software should initialize any memory in the Controller Memory Buffer before being referenced (e.g., a Completion Queue shall be initialized by host software in order for the Phase Tag to be used correctly (refer to section 4.2.4)).

A CMB implementation has a maximum sustained write throughput. The CMB implementation may also have an optional write elasticity buffer used to buffer writes from CMB PCIe write requests. When the CMB sustained write throughput is less than the PCI Express link throughput, then such a write elasticity buffer allows PCIe write request burst throughput to exceed the CMB sustained write throughput without backpressuring into the PCI Express fabric.

The time required to transfer data from the write elasticity buffer to the CMB is the amount of data written to the elasticity buffer divided by the Controller Memory Buffer Sustained Write Throughput (refer to section 3.1.4.19). The time to transfer the entire contents of the write elasticity buffer is the Controller Memory Buffer Elasticity Buffer Size (refer to section 3.1.4.18) divided by the Controller Memory Buffer Sustained Write Throughput.

A controller memory-based queue is used in the same manner as a host memory-based queue – the difference is the memory address used is located within the controller's own memory rather than in the host memory. The Admin or I/O Queues may be placed in the Controller Memory Buffer. If the CMBLOC.CQMMS bit (refer to Figure 47) is cleared to '0', then for a particular queue, all memory associated with it shall reside in either the Controller Memory Buffer or outside the Controller Memory Buffer.

If the CMBLOC.CQPDS bit (refer to Figure 47) is cleared to '0', then for all queues in the Controller Memory Buffer, the queue shall be physically contiguous.

The controller may support PRP Lists and SGLs in the Controller Memory Buffer. If the CMBLOC.CDPMLS bit (refer to Figure 47) is cleared to '0', then for a particular PRP List or SGL associated with a single command, all memory containing the PRP List or SGL shall be either entirely located in the Controller Memory Buffer or entirely located outside the Controller Memory Buffer.

PRP Lists and SGLs associated with a command may be placed in the Controller Memory Buffer if that command is present in a Submission Queue in the Controller Memory Buffer. If:

- a) CMBLOC.CDPCILS bit (refer to Figure 47) is cleared to '0'; and
- b) a command is not present in a Submission Queue in the Controller Memory Buffer,

then the PRP Lists and SGLs associated with that command shall not be placed in the Controller Memory Buffer.

The controller may support data and metadata in the Controller Memory Buffer. If the CMBLOC.CDMMMS bit (refer to Figure 47) is cleared to '0', then all data and metadata, if any, associated with a particular command shall be either entirely located in the Controller Memory Buffer or entirely located outside the Controller Memory Buffer.

If the requirements for the Controller Memory Buffer use are violated by the host, the controller shall abort the associated command with a status code of Invalid Use of Controller Memory Buffer.

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

### 8.2.2 Doorbell Stride for Software Emulation

The doorbell stride, specified in CAP.DSTRD (refer to Figure 36), may be used to separate doorbells by a number of bytes in memory space. The doorbell stride is a number of bytes equal to  $(2^{(2 + \text{CAP.DSTRD})})$ . This is useful in software emulation of an NVM Express controller. In this case, a software thread is monitoring doorbell notifications. The software thread may be made more efficient by monitoring one doorbell per discrete cacheline or utilize the monitor/mwait CPU instructions. For hardware implementations of the NVM Express interface, the expected doorbell stride value is 0h.

### 8.2.3 Host Memory Buffer

The Host Memory Buffer (HMB) feature allows the controller to utilize an assigned portion of host memory exclusively. The use of the host memory resources is vendor specific. Host software may not be able to provide any or a limited amount of the host memory resources requested by the controller. The controller shall function properly without host memory resources. Refer to section 5.1.25.2.4.

The controller may indicate limitations for the minimum usable descriptor entry size and the maximum number of descriptor entries (refer to the HMMINDS and HMMAXD fields in the Identify Controller data structure, Figure 312). If the host does not create the Host Memory Buffer within the indicated limits, then the host memory allocated for use by the controller may not be fully utilized (e.g., descriptor entries beyond the maximum number of entries indicated may be ignored by the controller).

During initialization, host software may provide a descriptor list that describes a set of host memory address ranges for exclusive use by the controller. The host memory resources assigned are for the exclusive use of the controller (host software should not modify the ranges) until host software requests that the controller release the ranges and the controller completes the Set Features command. The controller is responsible for initializing the host memory resources. Host software should request that the controller release the assigned ranges prior to a shutdown event, a Runtime D3 event, or any other event that requires host software to reclaim the assigned ranges. After the controller acknowledges that the ranges are no longer in use, host software may reclaim the host memory resources. In the case of Runtime D3, host software should provide the host memory resources to the controller again and inform the controller that the ranges were in use prior to the RTD3 event and have not been modified.

The host memory resources are not persistent in the controller across a Controller Level Reset. Host software should provide the previously allocated host memory resources to the controller after that reset completes. If host software is providing previously allocated host memory resources (with the same contents) to the controller, the Memory Return bit (refer to Figure 447) is set to '1' in the Set Features command.

The controller shall ensure that there is no data loss or data corruption in the event of a surprise removal while the Host Memory Buffer feature is being utilized.

### 8.2.4 Persistent Memory Region

The Persistent Memory Region (PMR) is an optional region of general purpose PCI Express read/write persistent memory that may be used for a variety of purposes. The controller indicates support for the PMR by setting CAP.PMRS (refer to section 3.1.4.1) to '1' and indicates whether the controller supports command data and metadata transfers to or from the PMR by setting support flags in the PMRCAP property. When command data and metadata transfers to or from PMR are supported, all data and metadata associated with a particular command shall be either entirely located in the Persistent Memory Region or outside the Persistent Memory Region.

The PMR's PCI Express address range is used for external memory read and write requests to the PMR. The PCI Express address range and size of the PMR is defined by the PCI Base Address Register (BAR) indicated by PMRCAP.BIR. The PMR consumes the entire address region exposed by the BAR and supports all the required features of the PCI Express programming model (i.e., it in no way restricts what is otherwise permitted by PCI Express).

The controller uses the PMR's controller address range to reference PMR with addresses supplied by the host. The PCI Express address range and the controller address range of the PMR may differ, but both ranges have the same size, and equivalent offsets within each range have a one-to-one correspondence. The host configures the controller address range via the PMRMSCU and PMRMSCL properties.

The host enables the PMR's controller memory space via the PMRMSCL.CMSE bit. When controller memory space is enabled, if host supplies an address referencing the PMR's controller address range, then the controller directs memory read or write requests for this address to the PMR.

When the PMR's controller memory space is disabled, the controller does not consider any host-supplied address to reference the PMR's controller address range, and memory read and write requests are directed elsewhere (e.g., to memory other than the PMR).

The contents of data written to the PMR while the PMR is ready persists across power cycles, Controller Level Resets, and disabling of the PMR. The mechanism used to make a write to the PMR persistent is implementation specific. For example, in one implementation this may mean that a write to non-volatile memory has completed while in another implementation this may mean that the write has been stored in a non-volatile write buffer and is written to non-volatile memory at some later point.

A PMR implementation has a maximum sustained write throughput. The PMR implementation may also have an optional write elasticity buffer used to buffer writes from PMR PCIe write requests. When the PMR sustained write throughput is less than the PCI Express link throughput, then such a write elasticity buffer allows PCIe write request burst throughput to exceed the PMR sustained write throughput without backpressuring into the PCI Express fabric.

The time required to transfer data from the write elasticity buffer to nonvolatile media is the amount of data written to the elasticity buffer divided by the Persistent Memory Region Sustained Write Throughput (refer to section 3.1.4.26). The time to transfer the entire contents of the write elasticity buffer is the Persistent Memory Region Elasticity Buffer Size (refer to section 3.1.4.25) divided by the Persistent Memory Region Sustained Write Throughput.

The host enables the PMR by setting PMRCTL.EN to '1'. Once enabled, the controller indicates that the PMR is ready by clearing PMRSTS.NRDY to '0'. It is not necessary to enable the controller to enable the PMR. Restoring and saving the contents of the PMR may take time to complete. When the host modifies the value of PMRCTL.EN, the host should wait for at least the time interval specified in PMRCAP.PMRT0 for PMRSTS.NRDY to reflect the change.

When the PMR is not ready, PMR reads complete successfully and return an undefined value while PMR writes complete normally, but do not update memory (i.e., the contents of the PMR address written remains unchanged). The undefined value returned by a PMR read following a sanitize operation is such that recovery of any previous user data from any cache or the non-volatile storage media is not possible.

When the PMR becomes read-only or unreliable, then a critical warning is reported in the SMART/Health Information Log which may be used to trigger an NVMe interface asynchronous event. Since reporting of an asynchronous event may occur an unspecified amount of time after the PMR health status has changed, the host should assume that all operations to the PMR have been affected since the last time normal operation was reported in PMRSTS.HSTS.

PMRCAP.PMRWBM enumerates supported PMR write barrier mechanisms. At least one mechanism shall be supported. An implementation may optionally support a mechanism where a PCI Express read of any size to the PMR, including a "zero-length read," ensures that all previous memory writes (i.e., Posted PCI Express requests) to the PMR have completed and are persistent. An implementation may optionally support a write barrier mechanism that utilizes a read of the PMRSTS property. When supported, a read of the PMRSTS property allows a host to:

- ensure that previously issued memory writes to the PMR have completed; and
- determine whether the PMR updates associated with those writes have completed without error and are persistent.

A PMR memory write error may be the result of a poisoned PCI Express TLP, an NVM subsystem internal error, or a PMR health status issue.

Regardless of the supported PMR write barrier mechanisms, a host may periodically read the PMRSTS property to ensure that reads to the PMR have returned valid data. For example, if a read to the PMRSTS property indicates that the PMR is operating normally is then followed by a series of reads, and finally a second read to the PMRSTS property that indicates the PMR is unreliable, then one or more of the reads between the two PMRSTS property reads may have returned invalid data. Such polling of the PMRSTS property may be unnecessary if the host handles poisoned TLPs and/or poisoned TLP error reporting is enabled.

The PMR write elasticity buffer size along with the PMR sustained write throughput allows a host to determine the amount of time for a read associated with a Persistent Memory Region write barrier mechanism to complete.

Support for PRPs, SGL Lists, Completion Queues, and Submission Queues in the Persistent Memory Region is outside the scope of this specification. If the host attempts to use the Persistent Memory Region for a PRP, SGL List, Completion Queue, or Submission Queue, the controller may abort the command with a status code of Invalid Field in Command.

### 8.2.5 Power Loss Signaling

Power Loss Signaling (PLS) is a capability that the host uses to inform all controllers in a domain of impending power loss, and that each controller uses to inform the host that the controller is preparing for that power loss.

There are two modes of Power Loss Signaling processing, Forced Quiescence Processing (refer to section 8.2.5.2) and Emergency Power Fail Processing (refer to section 8.2.5.3). All controllers in a domain support the same modes (i.e., all controllers report the same value in the PLSFQ bit and all controllers report the same value in the PLSEPF bit; refer to Figure 312).

Not more than one Power Loss Signaling mode is active at any time. The host uses the Power Loss Signaling Config feature (refer to section 5.1.25.1.19) to select the mode of operation or to disable Power Loss Signaling. All controllers in a domain use the same mode (i.e., the scope of the Power Loss Signaling Config feature is domain). The selection persists across power cycles as defined in Figure 385.

Each controller contains two variables which are used by Power Loss Signaling to perform communication between the host and controller:

- The Power Loss Notification (PLN) variable is set by the NVMe Transport and has two values, Asserted and Deasserted.
- The Power Loss Acknowledge (PLA) variable is set by the controller and has four values, Asserted-FQ, Asserted-EPF-Enabled, Asserted-EPF-Disabled, and Deasserted.

The values of the variables are described in Figure 659.

**Figure 659: Power Loss Signaling Variables**

Variable Name	Reset <sup>1</sup>	Description						
PLN	Deasserted	<b>PLN Value:</b> The PLN variable values are defined as follows:						
		<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>Asserted</td> <td>A power loss is impending.</td> </tr> <tr> <td>Deasserted</td> <td>A power loss is not impending.</td> </tr> </tbody> </table>	Value	Definition	Asserted	A power loss is impending.	Deasserted	A power loss is not impending.
		Value	Definition					
Asserted	A power loss is impending.							
Deasserted	A power loss is not impending.							

Figure 659: Power Loss Signaling Variables

Variable Name	Reset <sup>1</sup>	Description										
PLA	Deasserted	<b>PLA Value:</b> The PLA variable values are defined as follows:										
		<table border="1"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>Asserted-FQ</td> <td>The controller is performing Forced Quiescence Processing (refer to section 8.2.5.2).</td> </tr> <tr> <td>Asserted-EPF-Enabled</td> <td>The controller is performing Emergency Power Fail processing and the port is enabled (refer to section 8.2.5.3).</td> </tr> <tr> <td>Asserted-EPF-Disabled</td> <td>The controller is performing Emergency Power Fail Processing and the port is disabled (refer to section 8.2.5.3).</td> </tr> <tr> <td>Deasserted</td> <td>The controller is not performing Power Loss Signaling processing.</td> </tr> </tbody> </table>	Value	Definition	Asserted-FQ	The controller is performing Forced Quiescence Processing (refer to section 8.2.5.2).	Asserted-EPF-Enabled	The controller is performing Emergency Power Fail processing and the port is enabled (refer to section 8.2.5.3).	Asserted-EPF-Disabled	The controller is performing Emergency Power Fail Processing and the port is disabled (refer to section 8.2.5.3).	Deasserted	The controller is not performing Power Loss Signaling processing.
		Value	Definition									
		Asserted-FQ	The controller is performing Forced Quiescence Processing (refer to section 8.2.5.2).									
		Asserted-EPF-Enabled	The controller is performing Emergency Power Fail processing and the port is enabled (refer to section 8.2.5.3).									
Asserted-EPF-Disabled	The controller is performing Emergency Power Fail Processing and the port is disabled (refer to section 8.2.5.3).											
Deasserted	The controller is not performing Power Loss Signaling processing.											
Note:												
1. Following a Controller Level Reset, the variable shall be set to this value.												

Transport-specific details of the PLN variable and the PLA variable, including effects on communication connectivity between host and controller, are described in the Power Loss Signaling Support section of the appropriate NVMe Transport specification and in the Power Loss Signaling Interactions section of the NVM Express Management Interface Specification.

If the controller supports Power Loss Signaling, then the controller:

- shall support the PLN variable as specified in this section;
- may support the PLA variable as specified in this section;
- shall support Forced Quiescence Processing (refer to section 8.2.5.2), Emergency Power Fail Processing (refer to section 8.2.5.3), or both;
- if Forced Quiescence Processing is supported, shall report a non-zero value in the Forced Quiescence Vault Time field in the Power State Descriptor of one or more of the supported power states (refer to Figure 313);
- if Emergency Power Fail Processing is supported, shall report non-zero values in the Emergency Power Fail Vault Time field and the Emergency Power Fail Recovery Time field in the Power State Descriptor of one or more of the supported power states (refer to Figure 313);
- shall support reporting of whether I/O performance is degraded in the I/O Impacted (IOI) field (i.e., reports values 10b and 11b) in the I/O Command Set Independent Identify Namespace data structure (refer to Figure 319); and
- shall support the Power Loss Signaling Config feature (refer to section 5.1.25.1.19).

If the PLN variable is set to Asserted, then the controller performs either Forced Quiescence or Emergency Power Fail Processing, as determined by the setting of the Power Loss Signaling Config feature.

The controller shall ignore transitions in the PLN variable if:

- a Controller Level Reset (refer to section 3.7.2) is in process; or
- if the CSTS.SHST field is not cleared to 00b (i.e., the controller is in the process of shutting down or has completed shutdown).

If the PLN variable is set to Asserted and the controller is in a power state for which:

- the Emergency Power Fail Vault Time field is cleared to 0h;
- the Forced Quiescence Vault Time field is cleared to 0h; or
- the Emergency Power Fail Recovery Time field is cleared to 0h,

then the time to perform an action for which the corresponding value is cleared to 0h is vendor specific.

If the controller is in the EPF Complete Port Enabled state, the EPF Complete Port Disabled state, or the FQ Complete state and power is lost, then during the first restoration of power following the power loss,

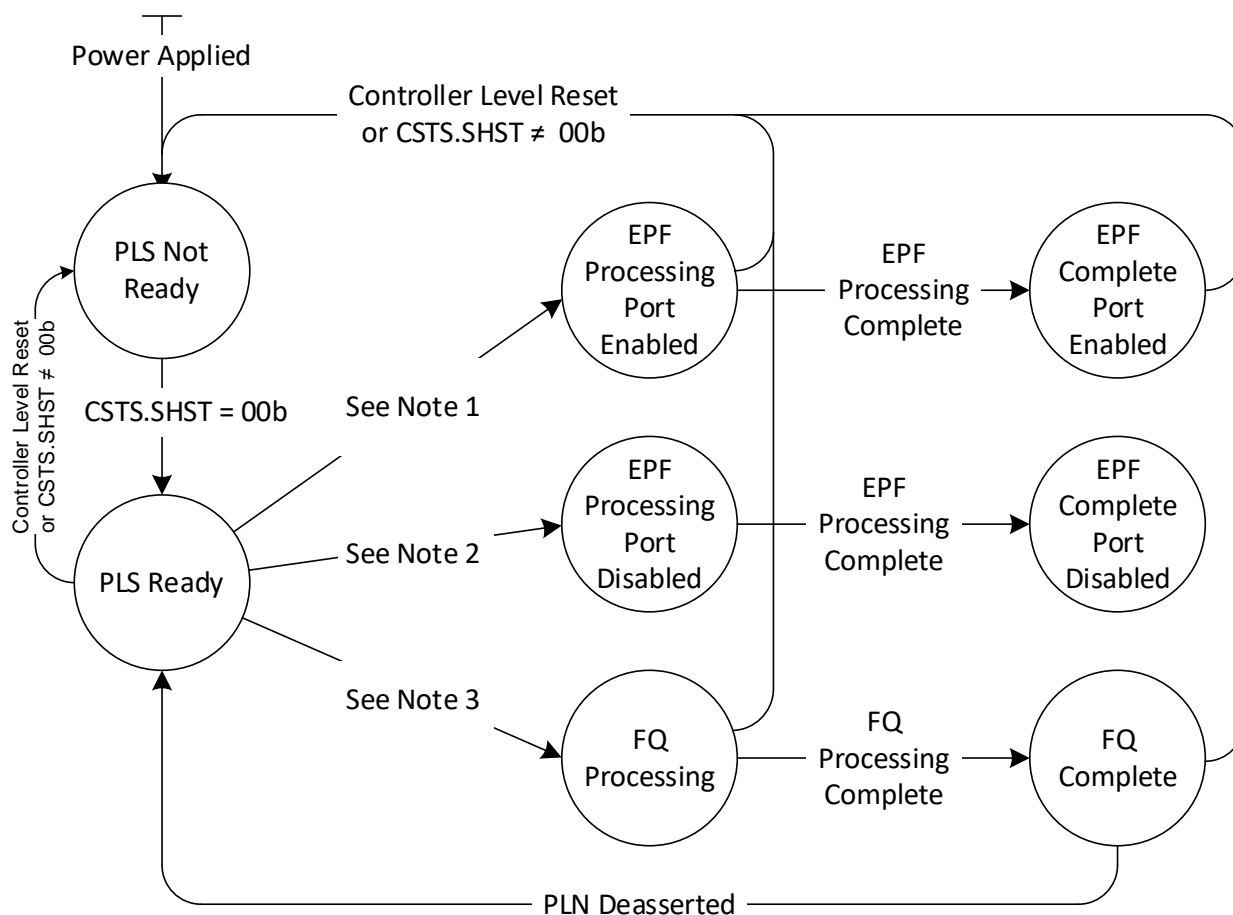
processing of commands may be affected while the controller performs internal recovery operations. Examples of these effects include:

- a) a namespace not being ready (i.e., the NRDY bit is cleared to '0' in the NSTAT field; refer to Figure 319); and
- b) commands to a namespace being processed at reduced performance, as indicated by the IOI field in the NSTAT field (refer to Figure 319).

### 8.2.5.1 Power Loss Signaling Processing State Machine

Figure 660 illustrates how transitions in the PLN variable initiate Power Loss Signaling processing by the controller. Each circle represents a processing state.

**Figure 660: Power Loss Signaling Processing State Machine**



Note 1: EPF is Enabled & PLN is Asserted & EPF Processing Port Enabled state is supported.

Note 2: EPF is Enabled & PLN is Asserted & EPF Processing Port Disabled state is supported.

Note 3: FQ is Enabled & PLN is Asserted.

For all states, a power cycle causes a transition to the PLS Not Ready state.

The Power Loss Signaling processing state of the controller determines the value of the PLA variable and whether communications on the port are processed (refer to Figure 661).

**Figure 661: PLS States**

State Name	PLA Variable Value (if supported)	Port Communication Processed
PLS Not Ready	Deasserted	Yes
PLS Ready	Deasserted	Yes
FQ Processing	Asserted-FQ	Yes
FQ Complete	Deasserted	Yes
EPF Processing Port Disabled <sup>1</sup>	Asserted-EPF-Disabled	No
EPF Complete Port Disabled	Deasserted	No
EPF Processing Port Enabled <sup>1</sup>	Asserted-EPF-Enabled	Yes
EPF Complete Port Enabled	Deasserted	Yes
Notes:		
1. The controller shall not implement both the EPF Processing Port Disabled state and the EPF Processing Port Enabled state.		

Note: I/O command processing in all of the PLS states, other than the PLS Not Ready state, complies with atomic operation requirements for power fail, if any, as specified in the appropriate I/O Command Set specification.

The conditions which trigger state transitions are described in the following sections. If a transition between two states can be caused by any one of multiple conditions, then those conditions are shown in a bullet list with an “or” (e.g., the transition from the FQ Processing state to the PLS Not Ready state, refer to Figure 664). If a transition is caused by multiple conditions which all must occur, then those conditions are shown as a single-item bullet list with an “and” (e.g., any of the transitions from the PLS Ready state, refer to Figure 663).

#### 8.2.5.1.1 PLS Not Ready State

In the PLS Not Ready state, the controller is not performing Power Loss Signaling processing. The controller enters the PLS Not Ready state following any Controller Level Reset or if the CSTS.SHST field is not cleared to 00b (i.e., the controller is in the process of shutting down or has completed shutdown).

Transitions out of this state are defined in Figure 662.

**Figure 662: PLS Not Ready State Transition Conditions**

State Transitions		Transition Condition
Starting	Ending	
PLS Not Ready	PLS Ready	<ul style="list-style-type: none"> <li>CSTS.SHST field is 00b.</li> </ul>

#### 8.2.5.1.2 PLS Ready State

In the PLS Ready state, the controller is not performing Power Loss Signaling processing and is not performing shutdown processing. The controller enters the PLS Ready state when the CSTS.SHST field is 00b

Transitions out of this state are defined in Figure 663. If the controller is in this state and the CSTS.RDY bit is cleared to ‘0’, then it is implementation specific whether the controller responds to a transition of the PLN variable from Deasserted to Asserted.

**Figure 663: PLS Ready State Transition Conditions**

State Transitions		Transition Condition
Starting	Ending	
PLS Ready	FQ Processing	<ul style="list-style-type: none"> <li>The controller is configured for FQ processing; and</li> <li>the PLN variable transitions from Deasserted to Asserted.</li> </ul>
	EPF Processing Port Disabled	<ul style="list-style-type: none"> <li>The controller is configured for EPF processing;</li> <li>the controller implements the EPF Processing Port Disabled state; and</li> <li>the PLN variable transitions from Deasserted to Asserted.</li> </ul>
	EPF Processing Port Enabled	<ul style="list-style-type: none"> <li>The controller is configured for EPF processing;</li> <li>the controller implements the EPF Processing Port Enabled state; and</li> <li>the PLN variable transitions from Deasserted to Asserted.</li> </ul>
	PLS Not Ready	<ul style="list-style-type: none"> <li>The controller processes a Controller Level Reset; or</li> <li>CSTS.SHST is not cleared to 00b.</li> </ul>

**8.2.5.1.3 FQ Processing State**

In the FQ Processing state, the controller is performing Forced Quiescence Processing, as described in section 8.2.5.2.

Transitions out of this state are defined in Figure 664.

**Figure 664: FQ Processing State Transition Conditions**

State Transitions		Transition Condition
Starting	Ending	
FQ Processing	FQ Complete	<ul style="list-style-type: none"> <li>The controller completes FQ processing.</li> </ul>
	PLS Not Ready	<ul style="list-style-type: none"> <li>The controller processes a Controller Level Reset; or</li> <li>CSTS.SHST is not cleared to 00b.</li> </ul>

**8.2.5.1.4 FQ Complete State**

In the FQ Complete state, the controller has completed Forced Quiescence Processing.

Transitions out of this state are defined in Figure 665.

**Figure 665: FQ Complete State Transition Conditions**

State Transitions		Transition Condition
Starting	Ending	
FQ Complete	PLS Not Ready	<ul style="list-style-type: none"> <li>The controller processes a Controller Level Reset; or</li> <li>CSTS.SHST is not cleared to 00b.</li> </ul>
	PLS Ready	<ul style="list-style-type: none"> <li>The PLN variable is set to Deasserted.</li> </ul>

**8.2.5.1.5 EPF Processing Port Disabled State**

In the EPF Processing Port Disabled state, the controller is performing Emergency Power Fail Processing, as described in section 8.2.5.3. The port is disabled. The following are not able to be initiated through the port:

- a) Controller Level Reset;
- b) controller shutdown;
- c) NVM Subsystem Reset; and
- d) NVM Subsystem Shutdown.

Transitions out of this state are defined in Figure 666.

**Figure 666: EPF Processing Port Disabled State Transition Conditions**

State Transitions		Transition Condition
Starting	Ending	
EPF Processing Port Disabled	EPF Complete Port Disabled	<ul style="list-style-type: none"> <li>The controller completes EPF processing.</li> </ul>

**8.2.5.1.6 EPF Complete Port Disabled State**

In the EPF Complete Port Disabled state, the controller has completed Emergency Power Fail Processing. The port is disabled. The following are not able to be initiated through the port:

- a) Controller Level Reset;
- b) controller shutdown;
- c) NVM Subsystem Reset; and
- d) NVM Subsystem Shutdown.

Transitions out of this state are defined in Figure 667.

**Figure 667: EPF Complete Port Disabled State Transition Conditions**

State Transitions		Transition Condition
Starting	Ending	
EPF Complete Port Disabled	PLS Not Ready	<ul style="list-style-type: none"> <li>Power is cycled.</li> </ul>

**8.2.5.1.7 EPF Processing Port Enabled State**

In the EPF Processing Port Enabled state, the controller is performing Emergency Power Fail Processing, as described in section 8.2.5.3. The port is enabled.

Transitions out of this state are defined in Figure 668.

**Figure 668: EPF Processing Port Enabled State Transition Conditions**

State Transitions		Transition Condition
Starting	Ending	
EPF Processing Port Enabled	EPF Complete Port Enabled	<ul style="list-style-type: none"> <li>The controller completes EPF processing.</li> </ul>
	PLS Not Ready	<ul style="list-style-type: none"> <li>The controller processes a Controller Level Reset; or</li> <li>CSTS.SHST is not cleared to 00b.</li> </ul>

**8.2.5.1.8 EPF Complete Port Enabled State**

In the EPF Complete Port Enabled state, the controller has completed Emergency Power Fail Processing. The port is enabled.

Transitions out of this state are defined in Figure 669.

**Figure 669: EPF Complete Port Enabled State Transition Conditions**

State Transitions		Transition Condition
Starting	Ending	
EPF Complete Port Enabled	PLS Not Ready	<ul style="list-style-type: none"> <li>The controller processes a Controller Level Reset; or</li> <li>CSTS.SHST is not cleared to 00b.</li> </ul>

**8.2.5.2 Forced Quiescence Processing**

This section describes the behavior of each controller in the domain when the domain is configured for Power Loss Signaling with Forced Quiescence (refer to section 5.1.25.1.19).



If the controller enters the FQ Processing state, then the controller shall perform the following actions in sequence:

1. set the PLA variable, if supported, to Asserted-FQ (refer to Figure 661);
2. stop fetching commands on all submission queues;
3. perform the following actions in parallel or in any sequence:
  - a) process commands received out-of-band on a Management Endpoint as described in the Power Loss Signaling Interactions section of the NVM Express Management Interface Specification;
  - b) if a command was fetched prior to entering this state, then process that command as described in this section; and
  - c) prepare for power loss;
4. enter the FQ Complete state (refer to Figure 664); and
5. set the PLA variable to Deasserted (refer to Figure 661).

Entry to the FQ Processing state should cause the controller to complete processing of all previously fetched commands (e.g., to post a CQE with a status code of Successful Completion, to abort the command and post a CQE with an appropriate status code). If a background operation is in progress (e.g., a device self-test operation or a sanitize operation), then that background operation should be suspended until the PLN variable is set to Deasserted and fetching and processing commands resumes.

While in the FQ Processing state, the controller shall abort a previously-fetched Set Features command specifying the Power Loss Signaling Config feature identifier with a status code of Commands Aborted due to Power Loss Notification.

While in the FQ Processing state, if power is lost and subsequently restored, then resumption of command processing:

- a) may take more time than resumption of command processing after a normal shutdown completes and then power is cycled; and
- b) typically takes less time than resumption of command processing after an abrupt shutdown completes and then power is cycled.

If the controller transitions from the FQ Processing state to the PLS Not Ready state, then the controller shall abort Forced Quiescence Processing.

Forced Quiescence Processing shall not cause a loss of communication connectivity between the host and the controller.

If the PLN variable is set to Asserted and is then set to Deasserted without an intervening loss of main power, Controller Level Reset, or shutdown (refer to Figure 660), then the controller resumes fetching and processing commands.

### 8.2.5.3 Emergency Power Fail Processing

This section describes the behavior of each controller in the domain when the domain is configured for Power Loss Signaling with Emergency Power Fail (refer to section 5.1.25.1.19).

If the controller enters the EPF Processing Port Disabled state or the EPF Processing Port Enabled state as described in Figure 663, then the controller shall perform the following actions in sequence:

1. set the PLA variable, if supported, to the value specified in Figure 661 for that state (i.e., Asserted-EPF-Disabled or Asserted-EPF-Enabled);
2. stop fetching commands on all submission queues;
3. perform the following actions in parallel or in any sequence:
  - a) process port communications as described in Figure 661; and
  - b) prepare for power loss in a manner that may or may not allow command processing to resume quickly in the event of power loss and then power resumption;

4. enter either the EPF Complete Port Enabled state (refer to Figure 668) or the EPF Complete Port Disabled state (refer to Figure 667); and
5. set the PLA variable, if supported, to Deasserted (refer to Figure 661).

Entry to the EPF Processing Port Disabled state or the EPF Processing Port Enabled state shall cause the controller to discard commands that were fetched from a submission queue prior to entry to the state, and to discard commands that were received out-of-band on a Management Endpoint prior to entry to the state.

The controller may implement the EPF Processing Port Disabled state or the EPF Processing Port Enabled state. The controller shall not implement both states. All controllers in the domain shall implement the same state.

The Emergency Power Fail Recovery Time (EPFRT) field and the Emergency Power Fail Recovery Time Scale (EPFRTS) field (refer to Figure 313) indicate the time that the controller requires to complete recovery during the first initialization following successful completion of Emergency Power Fail Processing. It is implementation specific whether the controller begins recovery before or after the host sets the CC.EN bit to '1'. Recovery may or may not be complete when the controller sets the CSTS.RDY bit to '1'.

If Emergency Power Fail Processing does not complete successfully (e.g., main power was lost before completion of processing), then the recovery time may exceed the Emergency Power Fail Recovery Time.

### 8.2.6 Virtualization Enhancements

Virtualized environments may use an NVM subsystem with multiple controllers to provide virtual or physical hosts direct I/O access. The NVM subsystem is composed of primary controller(s) and secondary controller(s), where the secondary controller(s) depend on primary controller(s) for dynamically assigned resources. A host may issue the Identify command to a primary controller specifying the Secondary Controller List to discover the secondary controllers associated with that primary controller. All secondary controllers shall be part of the same domain as the primary controller with which they are associated.

Controller resources may be assigned or removed from a controller using the Virtualization Management command (refer to section 5.2.6) issued to a primary controller. The following types of controller resources are defined:

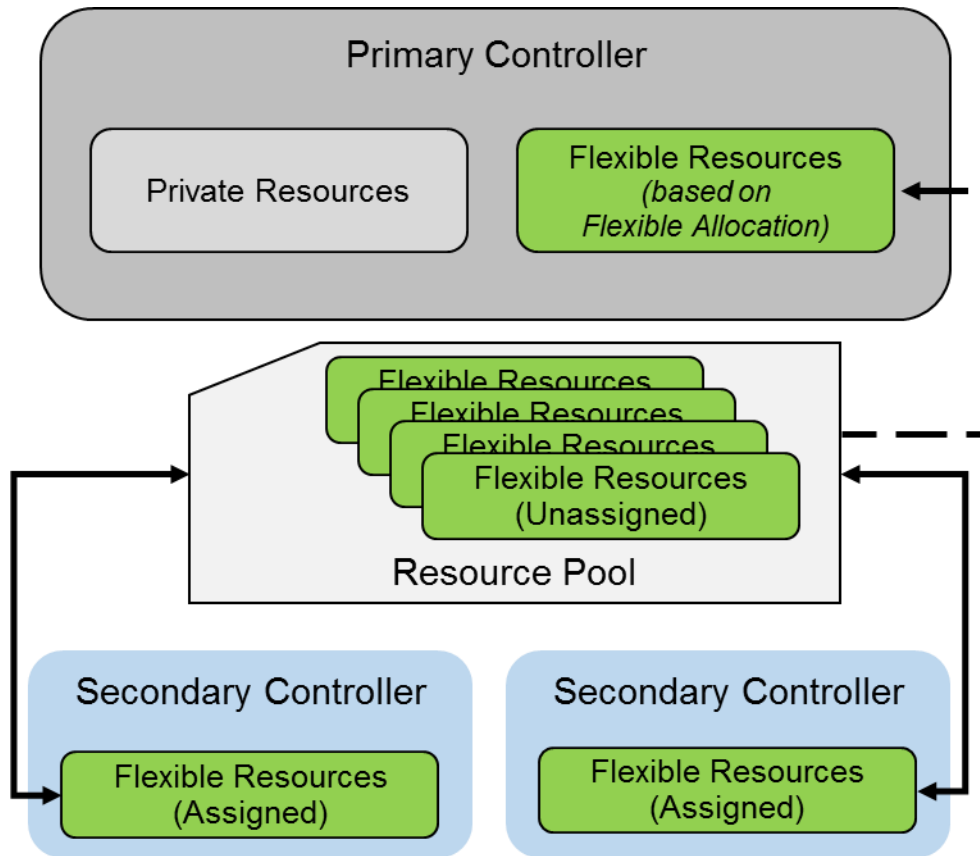
- Virtual Queue Resource (VQ Resource): a type of controller resource that manages one Submission Queue (SQ) and one Completion Queue (CQ) (refer to section 8.2.6.1); and
- Virtual Interrupt Resource (VI Resource): a type of controller resource that manages one interrupt vector (refer to section 8.2.6.2).

Flexible Resources are controller resources that may be assigned to the primary controller or one of its secondary controllers. The Virtualization Management command is used to provision the Flexible Resources between a primary controller and one of its secondary controller(s). A primary controller's allocation of Flexible Resources may be modified using the Virtualization Management command and the change takes effect after any Controller Level Reset other than a Controller Reset. A secondary controller only supports having Flexible Resources assigned or removed when in the Offline state.

Private Resources are controller resources that are permanently assigned to a primary or secondary controller. These resources are not supported by the Virtualization Management command.

The primary controller is allowed to have a mix of Private and Flexible Resources for a particular controller resource type. If there is a mix, then the Private Resources occupy the lower contiguous range of resource identifiers starting with 0. Secondary controllers shall have all Private or all Flexible Resources for a particular resource type. Controller resources assigned to a secondary controller always occupy a contiguous range of identifiers with no gaps, starting with 0. If a particular controller resource type is supported as indicated in the Controller Resource Types field of the Primary Controller Capabilities Structure, then all secondary controllers shall have that controller resource type assigned as a Flexible Resource. Figure 670 shows the controller resource allocation model for a controller resource type that is assignable as a Flexible Resource.

**Figure 670: Controller Resource Allocation**



For each controller resource type supported, the Primary Controller Capabilities Structure (refer to Figure 330) defines:

- The total number of Flexible Resources;
- The total number of Private Resources for the primary controller;
- The maximum number of Flexible Resources that may be assigned to a secondary controller using the Virtualization Management command; and
- The assignment of resources to the primary controller.

Primary and secondary controllers may implement all features of this specification, except where commands are defined as being only supported by a primary controller. It is recommended that only primary controllers support the privileged actions described in section 3.10 so that untrusted hosts using secondary controllers do not impact the entire NVM subsystem state.

The Secondary Controller List structure returned by the Identify command is used to determine the topology of secondary controllers and the resources assigned. The secondary controller shall be in the Offline state to configure resources. The Virtualization Management command is used to transition the secondary controller between the Online state and the Offline state. Refer to section 8.2.6.3 for details on the Online and Offline states.

To support the Virtualization Enhancements capability, the NVM subsystem shall support the following:

- One or more primary controllers, each of which supports:
  - One or more secondary controllers;
  - A pool of unassigned Flexible Resources that supports allocation to a primary controller and dynamic assignment to its associated secondary controllers;
  - Two or more Private Resource queue pairs;

- Indicate support for the Virtualization Management command by setting the VMS bit to '1' in the Optional Admin Command Support (OACS) field in the Identify Controller data structure;
- The Virtualization Management command;
- The Primary Controller Capabilities Structure defined in Figure 330 (Identify command with CNS value of 14h);
- The Secondary Controller List defined in Figure 330 (Identify command with CNS value of 15h); and
- The Namespace Management capability (refer to section 8.1.15);
- One or more secondary controllers; and
- Flexible Resources, each of which supports all of the following:
  - Assignment and removal by exactly one primary controller; and
  - Assignment to no more than one controller at a time.

Within an NVM subsystem that supports both the Virtualization Enhancements capability and SR-IOV (refer to section 8.2.6.4), all controllers that are SR-IOV PFs shall be primary controllers, and all controllers that are SR-IOV VFs shall be secondary controllers of their associated PFs.

### 8.2.6.1 VQ Resource Definition

A Virtual Queue Resource (VQ Resource) is a type of controller resource that manages one CQ and one SQ. For a VQ Resource that is assigned to a controller, its resource identifier is equivalent to its Queue Identifier.

The Controller Resource Types field of the Primary Controller Capabilities Structure indicates whether VQ Resources are supported. If VQ Resources are unsupported, a primary controller and its associated secondary controllers have all queues as Private Resources. The rest of this section assumes that VQ Resources are supported.

The secondary controller is assigned VQ Resources using the Virtualization Management command. The number of VQ Resources assigned is discoverable in the Secondary Controller List entry for the associated secondary controller. The number of VQ Resources assigned may also be discovered using the Get Features command with the Number of Queues Feature identifier (refer to section 5.1.25.2.1).

If a secondary controller has no assigned VQ Resources, then that controller remains in the Offline state. A secondary controller is not able to transition to the Online state until VQ Resources for an Admin Queue and one or more I/O Queues have been assigned to that controller (i.e., the minimum number of VQ Resources that may be assigned is two).

A primary controller that supports VQ Resources shall have at least two queue pairs that are Private Resources to ensure there is a minimum of an Admin Queue pair and one I/O queue pair for the primary controller at all times. A primary controller may be allocated VQ Resources using the Primary Controller Flexible Allocation action of the Virtualization Management command. The VQ resources allocated take effect after a Controller Level Reset and are persistent across power cycles and resets. The number of VQ Resources currently allocated is discoverable in the Primary Controller Capabilities Structure. The number of VQ Resources currently allocated may also be discovered using the Get Features command with the Number of Queues Feature identifier (refer to section 5.1.25.2.1).

### 8.2.6.2 VI Resource Definition

A Virtual Interrupt Resource (VI Resource) is a type of controller resource that manages one interrupt vector, such as an MSI-X vector. For a VI Resource that is assigned to a controller, its resource identifier is equivalent to its interrupt vector number.

The Controller Resource Types field of the Primary Controller Capabilities Structure indicates whether VI Resources are supported. If VI Resources are unsupported, a primary controller and its associated secondary controllers have all interrupts as Private Resources. The rest of this section assumes that VI Resources are supported.

The secondary controller is assigned VI Resources using the Virtualization Management command. The number of VI Resources assigned is discoverable in the Secondary Controller List entry for the associated secondary controller.

While a primary controller and/or its associated secondary controllers may concurrently support multiple types of interrupt vectors (e.g., MSI and MSI-X), all the controllers' VI Resources shall contain interrupt resources for interrupt vectors of the same type. In this revision, MSI-X is the only supported type of VI Resource.

For a secondary controller that supports VI Resources with MSI-X vectors, if at least one VI Resource is assigned to that controller, MSIXCAP.MXC.TS (refer to the MSI-X Capability section of the NVMe over PCIe Transport Specification) indicates the number of VI Resources assigned to the controller. Since MSIXCAP.MXC.TS is read-only, the value shall only be updated when the secondary controller is in the Offline state. MSI-X Table Entries on the secondary controller for newly assigned VI Resources shall be reset to default values.

If a secondary controller that supports VI Resources has no assigned VI Resources, then that controller remains in the Offline state. A secondary controller is not able to transition to the Online state until a VI Resource for interrupt vector 0 has been assigned to that controller. For a secondary controller that supports VI Resources with MSI-X vectors, if no VI Resources are assigned to that controller, then MSIXCAP.MXC.TS is reserved.

A primary controller that supports VI Resources shall have at least one interrupt that is a Private Resource. Interrupt vector 0 is always assigned to the primary controller. A primary controller may be allocated VI Resources using the Primary Controller Flexible Allocation action of the Virtualization Management command. The VI resources allocated take effect after a Controller Level Reset and are persistent across power cycles and resets. The number of VI Resources currently allocated is discoverable in the Primary Controller Capabilities Structure. For a primary controller that supports VI Resources with MSI-X vectors, MSIXCAP.MXC.TS indicates an MSI-X Table size equal to the total number of Private Resources and the Flexible Resources currently allocated following a Controller Level Reset.

When an I/O CQ is created, the controller supports mapping that I/O CQ to any valid interrupt vector, regardless of whether they have the same resource identifier, as long as the I/O CQ and the interrupt vector are attached to the same controller.

### 8.2.6.3 Secondary Controller States and Resource Configuration

A secondary controller shall be in one of the following states:

- **Online:** The secondary controller may be in use by a host. Required resources have been assigned. The secondary controller may be enabled in this state (CC.EN may be set to '1' and CSTS.RDY may then transition to '1'); or
- **Offline:** The secondary controller may not be used by a host. CSTS.CFS shall be set to '1'. Controller properties other than CSTS are undefined in this state.

The host may request a transition to the Online or Offline state using the Virtualization Management command. When a secondary controller transitions from the Online state to the Offline state all Flexible Resources are removed from the secondary controller.

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.

If VI Resources are supported, then following this process ensures the MSI-X Table size indicated by MSIXCAP.MXC.TS is updated to reflect the appropriate number of VI Resources before the transition to the Online state.

A primary controller or secondary controller is enabled when CC.EN and CSTS.RDY are both set to '1' for that controller. A secondary controller is able to be enabled only when in the Online state. If the primary controller associated with a secondary controller is disabled or undergoes a Controller Level Reset, then the secondary controller shall implicitly transition to the Offline state. A secondary controller shall transition to the Offline state when a shutdown occurs (refer to section 3.6, section 3.1.4.5, and section 3.1.4.20) on the primary controller associated with that secondary controller.

Resources shall only be assigned to a secondary controller when in the Offline state. If the minimum number of resources are not assigned to a secondary controller, then a request to transition to the Online state shall fail for that secondary controller. For implementations that support SR-IOV, if VF Enable is cleared to '0' or NumVFs specifies a value that does not enable the associated secondary controller, then the secondary controller shall implicitly transition to the Offline state.

#### **8.2.6.4 Single Root I/O Virtualization and Sharing (SR-IOV)**

The PCI-SIG® PCI Express Base specification defines Single Root I/O Virtualization and Sharing Specification (SR-IOV) extensions to PCI Express that allow multiple System Images (SIs), such as virtual machines running on a hypervisor, to share PCI hardware resources. The primary benefit of SR-IOV is that it eliminates the hypervisor from participating in I/O operations which may be a significant factor limiting storage performance in some virtualized environments and allows direct SI access to PCI hardware resources.

A Physical Function (PF) is a PCI Express Function that supports the SR-IOV Capability, which in turn allows that PF to support one or more dependent Virtual Functions (VFs). These PFs and VFs may support NVM Express controllers that share an underlying NVM subsystem with multi-path I/O and namespace sharing capabilities (refer to section 2.4.1).

SR-IOV Virtual Functions (VFs) with an NVM Express Class Code (refer to the PCI Header section of the NVMe over PCIe Transport Specification) shall implement fully compliant NVM Express controllers. This ensures that the same host software developed for non-virtualized environments is capable of running unmodified within an SI.

For hosts where SR-IOV is unsupported or not needed, a controller that is a PF shall support operation as a stand-alone controller.

For a controller that is a PF, the requirements for SR-IOV Capability registers VF BAR0, VF BAR1, VF BAR2, VF BAR4, and VF BAR5 are the same as the requirements for PCI registers BAR0, BAR1, BAR4, and BAR5, respectively. For a controller that is a PF, SR-IOV Capability register VF BAR2 shall not support Index/Data Pair. Refer to the PCI Header section of the NVMe over PCIe Transport Specification.

To accommodate SR-IOV address range isolation requirements, VF BAR2 and VF BAR3 may support a 64-bit prefetchable memory register space which shall only be used for MSI-X Tables and MSI-X PBAs of VFs. MSI-X Table BIR = '2' and MSI-X PBA BIR = '2' are valid for controllers that are VFs. Refer to the MSI-X Capability section of the NVMe over PCIe Transport Specification.

While the controller properties of a controller that is a VF are accessible only if SR-IOV Control.VF MSE is set to '1', clearing VF MSE from '1' to '0' does not cause a reset of that controller. In this case, controller properties are hidden, but their values are not reset.

### **8.3 Message-Based Transport Extended Capabilities (Fabrics)**

This section describes extended capabilities that are specific to the Message-based transport model.

#### **8.3.1 Automated Discovery of NVMe-oF Discovery Controllers for IP Based Fabrics**

When operating in an IP based fabric, before transmitting a Fabrics Connect command, the IP address of the fabric interface of the Discovery controller is determined by one of the following methods:

- a) administrative configuration;
- b) discovered using DNS-SD (refer to RFC 6763); or
- c) obtained by some means not defined in this specification.

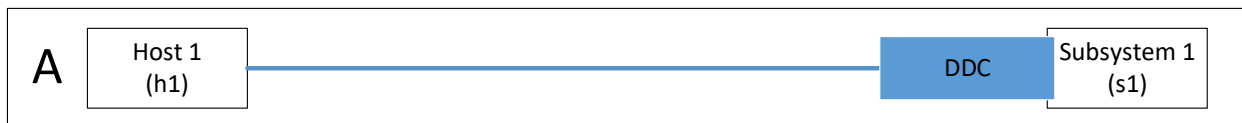
DNS-SD information may be retrieved using mDNS (refer to RFC 6762) or from a DNS server (refer to RFC 1034 and RFC 1035).

When DNS-SD is used as described in this section, hosts, CDCs and DDCs may use DNS-SD to perform automated discovery of Discovery controllers in IP fabrics consisting of:

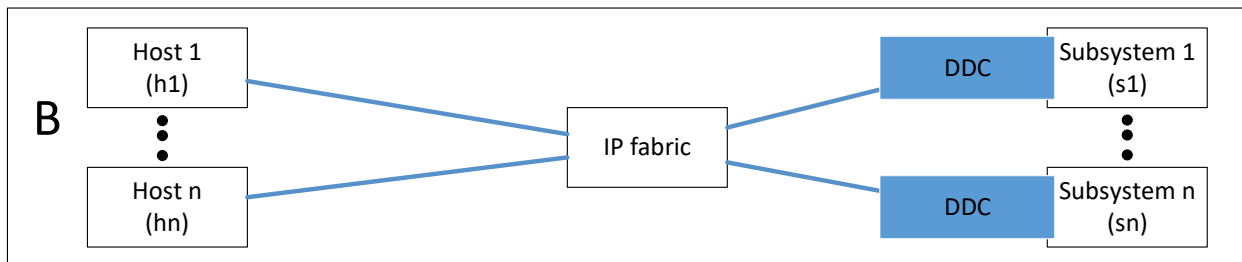
- a) Two IP interfaces (i.e., a host and an NVM subsystem) physically connected to one another (refer to Figure 671);
- b) Many IP interfaces participating in one or more Broadcast Domains (refer to Figure 672); or
- c) A Centralized Discovery controller and many IP interfaces that may reside in multiple Broadcast Domains (refer to Figure 673).

mDNS is a multicast protocol that allows discovery of IP interfaces within the same Broadcast Domain. Discovering IP interfaces outside of the Broadcast Domain using mDNS requires either the use of RFC 8766 or an mDNS NVMe-oF proxy. An mDNS NVMe-oF proxy is an mDNS responder that is responsive to queries for either the “\_nvme-disc” service or the “\_cdc.\_sub.\_nvme-disc” service and responds with the information defined in section 8.3.1.1.2.

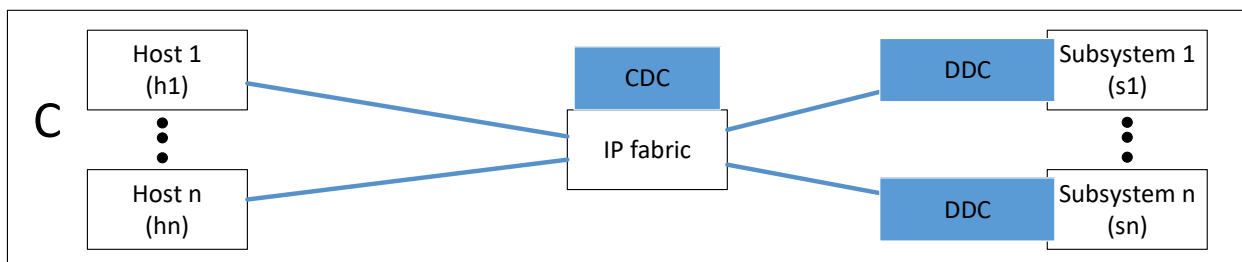
**Figure 671: Configuration A - Two IP Interfaces**



**Figure 672: Configuration B – Multiple IP Interfaces without a CDC**



**Figure 673: Configuration C – Multiple IP interfaces with a CDC**



### 8.3.1.1 Discovery of NVMe-oF Discovery Controllers

#### 8.3.1.1.1 Query

To facilitate the discovery of Discovery controller IP fabric interface addresses, hosts, CDCs and DDCs may transmit an mDNS query (refer to RFC 6762) or a DNS query (refer to RFC 1034 and RFC 1035) that includes a DNS PTR record (refer to RFC 6763) with the name in the form of:

“<Service>.<Domain>”.

The <Service> portion of the name can be further broken down into:

"<service name>.<protocol>".

For NVMe over Fabrics, the DNS PTR record included in the mDNS or DNS query shall be in the form of:

"\_nvme-disc.<protocol>.<domain>"; or

"\_<subtype>.\_sub.\_nvme-disc.<protocol>.<domain>".

The protocol field shall be set as shown in Figure 674.

The subtype field shall be set as shown in Figure 675.

The domain field shall be set as shown in Figure 676.

**Figure 674: mDNS Protocol Field**

NVMe-oF Transport	Fabric Protocol	mDNS <protocol> Field
TCP	TCP	"_tcp"
RDMA	RoCE	"_udp"
RDMA	iWARP	"_tcp"

**Figure 675: mDNS Subtype**

Subtype	Usage
"_cdc"	Used by CDC and DDC instances to detect the presence of a CDC service.
"_ddcpull"	Obsolete.

**Figure 676: mDNS Domain**

Domain	Usage
"local"	Used for mDNS
<FQDN>	A fully qualified domain name may be used when DNS-SD information is retrieved from a DNS server.

### 8.3.1.1.2 Response

The responses that may be received for an mDNS or DNS query include the following records as described in DNS-Based Service Discovery (refer to RFC 6763):

- A DNS PTR record (refer to section 3.3.12 in RFC 1035);
- a DNS SRV record (refer to RFC 2782);
- a DNS TXT record (refer to section 3.3.14 in RFC 1035); and
- an A record and/or AAAA record providing the IPv4 and IPv6 IP addresses respectively.

#### 8.3.1.1.2.1 DNS PTR record

The DNS PTR record included in the mDNS or DNS response shall be in the form of:

"<Service>.<Domain>".

The <Service> portion of the name can be further broken down into:

"<service name>.<protocol>".

For NVMe over Fabrics, the DNS PTR record included in the mDNS or DNS response shall be in the form of:

"\_nvme-disc.<protocol>.<domain>"; or

"\_<subtype>.\_sub.\_nvme-disc.<protocol>.<domain>".

The protocol field shall be set as shown in Figure 674.

The subtype field shall be set as shown in Figure 675.



The domain field shall be set as shown in Figure 676.

#### 8.3.1.1.2.2 DNS SRV record

The DNS SRV record provides the TCP port where the service instance can be reached and shall have a name in the form of:

“<Instance>.<Service>.<Domain>”.

Instance (Instance name) is a vendor defined human readable string. Although use of a serial number is discouraged in DNS-Based Service Discovery (refer to RFC 6763), an instance name that may be meaningful to an IT administrator is a combination of the Model Number (MN), Serial Number (SN) and Physical Fabric Interface (Physical Ports). For example:

“SubsystemIF-SerialNumber-SubsystemModel”.

The maximum length of the Instance Name field is 63 bytes (refer to RFC 6763).

The <Service> portion of the name (refer to section 4.1 of RFC 6763) can be further broken down into:

“<service name>.<protocol>”.

#### 8.3.1.1.2.3 DNS TXT record

The DNS TXT record provides additional information about the instance using key/value pairs in the form of “key=value” separated by commas (refer to section 6.4 in RFC 6763).

For NVMe over Fabrics, the DNS TXT record shall include a key/value pair for protocol (p) and may include a key/value pair describing an NQN.

DNS TXT record key/value pairs:

**Protocol (p):** the protocol field shall indicate the IP transport protocols that are supported by the Discovery controller being advertised. For example:

“p=tcp”, “p=roce”, or “p=iwarp”.

**NQN:** the DNS TXT record may contain an nqn key/value pair. When it is included the NQN provided shall be set to the unique NQN of the Discovery subsystem if one is available for use. Otherwise, the well-known Discovery Service NQN (nqn.2014-08.org.nvmexpress.discovery) may be used.

As described in RFC 6763, the format of the data within a DNS TXT record is one or more strings, packed together without any intervening gaps or padding bytes for word alignment.

The format of a TXT record that may be provided in a response is:

“<length byte>p=tcp<length byte>nqn=NQN.of.Discovery.subsystem”.

Using this format, an example of a TXT record that may be provided in a response is:

“05p=tcp1Enqn=NQN.of.Discovery.subsystem”.

### 8.3.1.2 Host Operation

#### 8.3.1.2.1 Host Query

As described in section 8.3.1.1.1, a host may transmit an mDNS or DNS query to discover CDC and DDC instances that are present on the transport network. When used, the mDNS or DNS query shall include a DNS PTR record (refer to RFC 6763) with the name in the form of:

“\_nvme-disc.<protocol>.<domain>”.

The protocol field shall be set as shown in Figure 674.

The domain field shall be set as shown in Figure 676.

### 8.3.1.2.2 Host Processing of DNS-SD Records

Upon reception of an mDNS or DNS response that contains a DNS SRV record with the service name set to “\_nvme-disc.<protocol>.local”. The host interface may use the IP address in the A or AAAA record as the destination IP address for a subsequent Fabrics Connect command and attempt to perform Discovery Information Registration (refer to section 8.3.2.2).

### 8.3.1.3 DDC Operation

#### 8.3.1.3.1 DDC mDNS Initialization

During initialization (e.g., following a link transition or power cycle), before the DDC’s mDNS responder function is enabled, the DDC shall probe to ensure the unique resource records the DDC are responsible for are unique on the local link (refer to section 8.1 in RFC 6762).

Upon successful completion of the probe, the DDC shall Announce (refer to section 8.2 in RFC 6762) its newly registered resource records.

Upon announcing its resource records, if a DDC:

- a. has not been configured to perform push registration (refer to section 8.3.2.2.1), or has not been configured to request a pull registration (refer to section 8.3.2.2.2) from a CDC, it may respond to mDNS queries for the service name of “\_nvme-disc.<protocol>.local”; or
- b. has been configured to perform push registration (refer to section 8.3.2.2.1) with a CDC, it should not respond to mDNS queries for the service name of “\_nvme-disc.<protocol>.local”, unless it has been administratively configured to do so or until it has performed a query and determined a CDC is not present as defined in section 8.3.1.3.3.

#### 8.3.1.3.2 DDC DNS Initialization

During initialization (e.g., following a link transition or power cycle), a DDC may dynamically update DNS records (refer to RFC 2136) by providing an update that includes the Resource Records defined in section 8.3.1.1.2.

#### 8.3.1.3.3 DDC Query

A DDC may determine if a CDC is present by transmitting a query that includes a DNS PTR record (refer to RFC 6763) with the name in the form of:

“\_cdc.\_sub.\_nvme-disc.<protocol>.<domain>”.

The protocol field shall be set as shown in Figure 674.

The domain field shall be set as shown in Figure 676.

#### 8.3.1.3.4 DDC Processing of DNS-SD records

Upon reception of an mDNS or DNS response that contains a DNS SRV record with the service name set to “\_cdc.\_sub.\_nvme-disc”, the DDC may use the IP address in the A or AAAA record as the destination IP address to either:

- a. Perform push registration (refer to section 8.3.2.2.1) with the CDC; or
- b. Request a pull registration (refer to section 8.3.2.2.2) from the CDC (e.g., using Kickstart Discovery Request PDU (KDRReq) section in the NVMe Transport Specification).

If a DDC supports mDNS and has been configured to perform push registration (refer to section 8.3.2.2.1), or has been configured to request a pull registration (refer to section 8.3.2.2.2) from a CDC, the DDC should cease responding to mDNS requests for the service name of “\_nvme-disc.<protocol>.local” if a CDC is detected as defined in section 8.3.1.3.3.

#### 8.3.1.3.5 DDC response to mDNS queries

A DDC may respond to mDNS queries for the service names of:

“\_nvme-disc.<protocol>.local”

mDNS responses to queries for these service names shall contain the information described in section 8.3.1.1.2.

DDCs should ignore mDNS queries for the service name of “\_ddcpull.\_sub.\_nvme-disc.<protocol>.local” (refer to Figure 675), as the subtype “\_ddcpull.\_sub” is obsolete.

### **8.3.1.4 CDC Operation**

#### **8.3.1.4.1 CDC mDNS Initialization**

During initialization (e.g., following a link transition or power cycle), before the CDC’s mDNS responder function is enabled, the CDC shall probe to ensure the unique resource records the CDC are responsible for are unique on the local link (refer to section 8.1 in RFC 6762).

Upon successful completion of the probe, the CDC shall announce (refer to section 8.2 in RFC 6762) its newly registered resource records.

Upon announcing its resource records, the CDC’s mDNS responder function may be enabled and respond to queries for the service name of “\_cdc.\_sub.\_nvme-disc.<protocol>.local” as described in section 8.3.1.4.5.

A CDC should query for the presence of another CDC as defined in section 8.3.1.4.3 and process responses as defined in section 8.3.1.4.4.

#### **8.3.1.4.2 CDC DNS Initialization**

During initialization (e.g., following a link transition or power cycle), a CDC may dynamically update DNS records (refer to RFC 2136) by providing an update that includes the Resource Records defined in section 8.3.1.1.2.

#### **8.3.1.4.3 CDC Query**

A CDC may query for other CDC instances. When performed the mDNS or DNS query shall include a DNS PTR record (refer to RFC 6763) with the name in the form of:

“\_cdc.\_sub.\_nvme-disc.<protocol>.<domain>.”

The protocol field shall be set as shown in Figure 674.

The domain field shall be set as shown in Figure 676.

#### **8.3.1.4.4 CDC Processing of DNS-SD records**

Upon reception of an mDNS or DNS response that contains a DNS SRV record with the service name set to “\_cdc.\_sub.\_nvme-disc.<protocol>.local” the CDC may provide an alert to the administrator to indicate the presence of more than one CDC in a broadcast domain.

A CDC should ignore an mDNS or DNS response that contains a DNS SRV record with the service name set to “\_ddcpull.\_sub.\_nvme-disc.<protocol>.local”.

#### **8.3.1.4.5 CDC response to mDNS queries**

A CDC may respond to mDNS queries for the service names of either:

“\_nvme-disc.<protocol>.local”; or

“\_cdc.\_sub.\_nvme-disc.<protocol>.local”.

mDNS responses to this query shall contain the information described in section 8.3.1.1.2.

### 8.3.2 Centralized Discovery for IP-based Fabrics

#### 8.3.2.1 Overview

In configurations that consist of multiple NVM subsystems, the burden on administrators is reduced by enabling hosts to automatically retrieve the list of NVM subsystem ports the host has been allowed to access from a centralized location. This centralized location is referred to as a Centralized Discovery controller (CDC).

Hosts and NVM subsystems may become known to the CDC by explicitly registering their discovery information as the result of a push registration (refer to section 8.3.2.2.1). If a host or Direct Discovery controller (DDC) detects the presence of multiple CDCs, then that host or DDC should register their discovery information with each CDC. Alternatively, the CDC may implicitly register discovery information as a result of processing a Connect command from a host, or as a result of receiving a pull registration request from a DDC (refer to section 8.3.2.2.2).

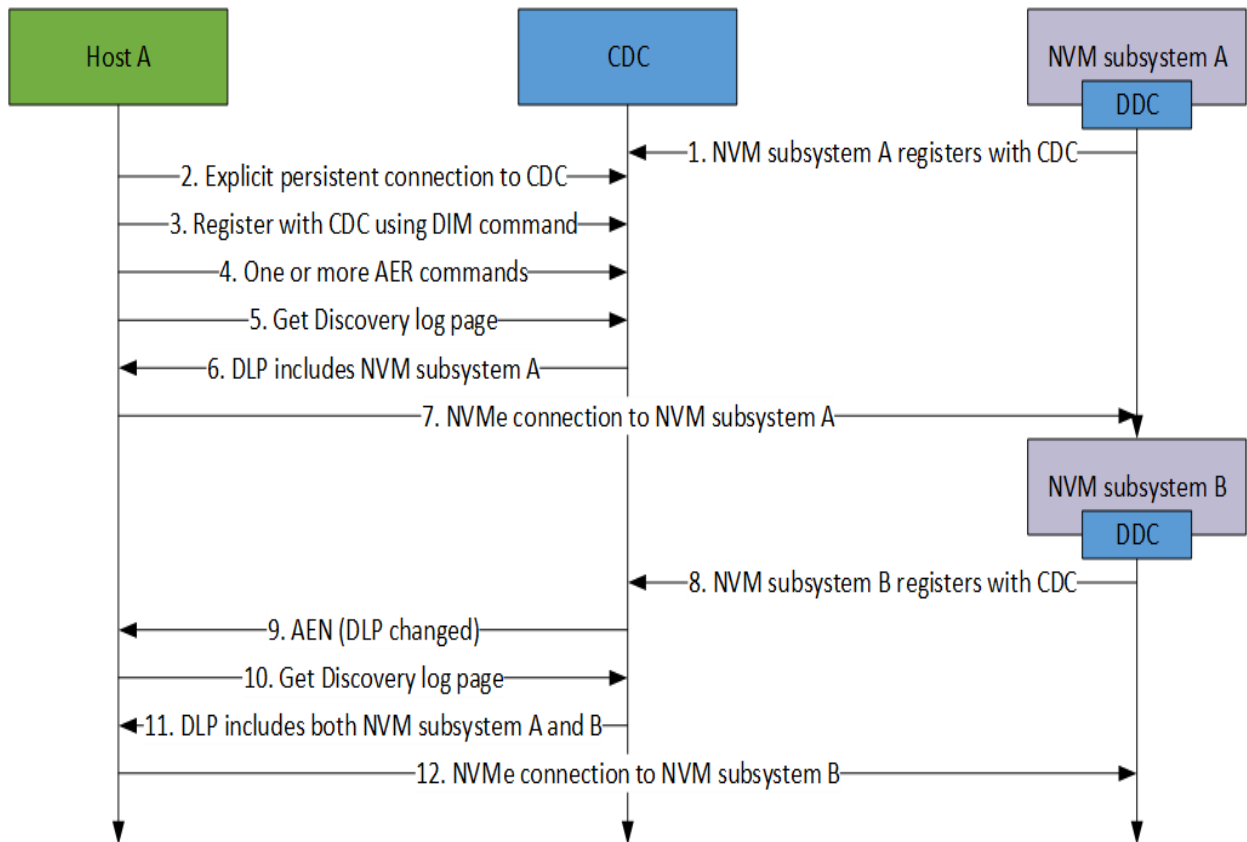
The CDC may filter the list of NVM subsystem ports returned in response to a Get Log Page command from the host that requests the Discovery log page to include only the NVM subsystem ports that provide access to namespaces allocated to that host. The process used to configure this filtering function is known as Fabric Zoning (refer to section 8.3.2.3).

An overview of the registration and discovery process is described in section 8.3.2.1.1.

##### 8.3.2.1.1 Registration and Discovery Example

The following example assumes the CDC's effective Fabric Zoning configuration allows Host A to access both NVM subsystem A and NVM subsystem B.

**Figure 677: Registration and Discovery Example**



Each of the numbered steps in Figure 677 are described below:

1. A fabric interface on NVM subsystem A is registered with the CDC. This is accomplished by either:
  - a. using the Discovery Information Management command (refer to section 5.3.3) to perform a push registration (refer to section 8.3.2.2.1);
  - b. notifying the CDC that a pull registration (refer to section 8.3.2.2.2) is required and the NVM subsystem port must be implicitly registered by the CDC; or
  - c. administrative configuration (e.g., the NVM subsystem port was manually configured on the CDC).
2. Host A should establish an explicit persistent connection with the CDC. The method that a host uses to obtain the information necessary to connect to the CDC via the NVMe Transport may be:
  - a. implementation specific;
  - b. fabric specific;
  - c. known in advance (e.g., a well-known address);
  - d. administratively configured; or
  - e. for IP based fabrics, Automated Discovery of Discovery Controllers for IP Based Fabrics (refer to section 8.3.1) may be used.
3. Host A uses the Discovery Information Management command to perform a push registration and explicitly register its discovery information with the CDC.
4. Host A should send one or more Asynchronous Event Request commands to the CDC in order to be notified about any changes that occur to the Discovery log page.
5. Host A uses the Get Log Page command to retrieve the Discovery log page from the CDC.
6. The CDC responds to the Get Log Page command with a Discovery log page containing one Discovery Log Page Entry for the interface on NVM subsystem A.
7. If the SUBTYPE field of the Discovery Log Page Entry for NVM subsystem A returned from the CDC is set to 02h (i.e., NVM subsystem), then Host A should connect to the I/O controller on NVM subsystem A.
8. A fabric interface on NVM subsystem B is registered with the CDC. This is accomplished by either:
  - a. using the Discovery Information Management command to perform a push registration;
  - b. notifying the CDC that a pull registration is required and the NVM subsystem port must be implicitly registered by the CDC; or
  - c. administratively configured (e.g., the NVM subsystem port was manually configured on the CDC).
9. The CDC sends a Discovery Log Page Change Asynchronous Event notification (Asynchronous Event Information F0h) to notify Host A that the Discovery log page has changed.
10. Host A uses the Get Log Page command to retrieve the Discovery log page from the CDC.
11. The CDC responds to the Get Log Page command with a Discovery log page containing two Discovery Log Page Entries: one for the interface on NVM subsystem A and another for the interface on NVM subsystem B.
12. If the SUBTYPE field of the Discovery log page for NVM subsystem B returned from the CDC is set to 02h (i.e., NVM subsystem), then Host A should connect to the I/O controller on NVM subsystem B. The connection between Host A and NVM subsystem A remains unchanged.

### 8.3.2.2 Discovery Information Registration and De-Registration

Discovery information registration is the process of registering host or NVM subsystem discovery information with a Centralized Discovery controller (CDC) or registering host discovery information with a Direct Discovery controller (DDC).

Discovery information registration may be performed:

- using a push registration (refer to section 8.3.2.2.1); or
- using a pull registration (refer to section 8.3.2.2.21).

Information from a Connect command (e.g., Host NQN) may be retained by a Discovery controller to provide a limited set of discovery information. If the Discovery controller determines that the host that submitted the Connect command is the same as a host for which the Discovery controller has retained host discovery information from a previous push registration, then the Discovery controller should continue to retain that host discovery information.

A host:

- should use push registrations to register host discovery information with a CDC or DDC; and
- is not able to use pull registrations to register discovery information.

A CDC:

- may use push registrations to register host discovery information with a DDC; and
- shall not use pull registrations to register discovery information.

A DDC:

- should use push registrations when registering NVM subsystem discovery information with a CDC;  
or
- may use pull registrations when registering NVM subsystem discovery information with a CDC.

Discovery information de-registration is the process of de-registering host or NVM subsystem discovery information with a CDC or de-registering host discovery information with a DDC.

Discovery information de-registration may be performed using one of the following methods:

- a push de-registration (refer to section 8.3.2.2.1); or
- a pull de-registration (refer to section 8.3.2.2.2).

A host:

- may use push de-registrations to de-register host discovery information with a CDC or DDC; and
- is not able to use pull de-registrations to de-register discovery information.

A CDC:

- may use push de-registrations to de-register host discovery information with a DDC; and
- shall not use pull de-registrations to de-register discovery information.

A DDC:

- should use push de-registrations to de-register NVM subsystem discovery information with a CDC;  
or
- may use pull de-registrations to de-register NVM subsystem discovery information with a CDC.

#### **8.3.2.2.1 Push Registrations and Push De-Registrations**

A push registration is performed using the Discovery Information Management command (refer to section 5.3.3) with the Task (TAS) field (refer to Figure 496) cleared to 0h to explicitly register discovery information for a host or NVM subsystem with a Centralized Discovery controller (CDC) or explicitly register discovery information for a host with a Direct Discovery controller (DDC).

A push de-registration is performed using the Discovery Information Management command with the TAS field set to 1h to explicitly de-register discovery information for a host or NVM subsystem with a CDC or explicitly de-register discovery information for a host with a DDC.

A DDC that performs push registrations or push de-registrations implements host functionality (e.g., submitting commands, processing command completions, providing a Host NQN, etc.).

#### **8.3.2.2.2 Pull Registrations and Pull De-Registrations**

A pull registration may be requested using a kickstart discovery request and response sequence (refer to section 8.3.2.2.2).

Upon completion of acknowledging a pull registration request (e.g., after a KDResp NVMe/TCP PDU (refer to the Kickstart Discovery Response PDU section in the NVMe TCP Transport Specification) has been sent by the Centralized Discovery controller (CDC)), the CDC performs the pull registration by:

1. sending a Connect command to the Direct Discovery controller (DDC) to establish an NVMe connection with that DDC; and
2. sending a Get Log Page command to the DDC with the Log Page Identifier (LID) field set to 70h to retrieve NVM subsystem discovery information contained in the associated Discovery log page from that DDC. The CDC may:
  - a. set the Port Local Entries Only (PLEO) bit to '1' in the Log Specific Parameter (LSP) field in that Get Log Page command to request that NVM subsystem discovery information entries for only NVM subsystem ports that are presented through the same NVM subsystem port on the DDC that received the Get Log Page command be returned; and
  - b. set the All NVM Subsystem Entries (ALLSUBE) bit to '1' in the LSP field in that Get Log Page command to request that records for all NVM subsystem ports be returned.

The DDC should match the NQN contained in the CDC NVMe Qualified Name (CDCNQN) field of the KDResp NVMe/TCP PDU with the NQN contained in the Host NVMe Qualified Name (HOSTNQN) field in the data portion of the Connect command to know that a CDC is performing a pull registration.

If the PLEO bit is cleared to '0' and the ALLSUBE bit is set '1' in the LSP field of the Get Log Page command, then the DDC shall return all NVM subsystem discovery information entries in the resulting Discovery log page sent to the CDC (i.e., the DDC shall not filter out any NVM subsystem discovery information entries).

If the PLEO bit is set to '1' and the ALLSUBE bit is set '1' in the LSP field of the Get Log Page command, then the DDC shall return all NVM subsystem discovery information entries for NVM subsystem ports that are presented through the same NVM subsystem port on the DDC that received the Get Log Page command in the resulting Discovery log page sent to the CDC (i.e., the DDC shall not filter out any NVM subsystem discovery information entries for NVM subsystem ports that are presented through the same NVM subsystem port on the DDC that received the Get Log Page command).

An additional pull registration or a pull de-registration may be requested by either:

- sending a Discovery Log Page Change Asynchronous Event notification (Asynchronous Event Information F0h) (refer to section 3.1.3.3.2); or
- requesting another pull registration.

If the CDC established an explicit persistent connection with the DDC when performing the previous pull registration, then the DDC should request the additional pull registration or the pull de-registration by sending a Discovery Log Page Change Asynchronous Event notification.

If the CDC did not establish an explicit persistent connection with the DDC when performing the previous pull registration, then the DDC requests the additional pull registration or the pull de-registration by requesting another pull registration.

Upon receiving a Discovery Log Page Change Asynchronous Event notification, the CDC performs the additional pull registration or the pull de-registration by using the following sequence:

1. sending a Get Log Page command to the DDC with the LID field set to 70h to retrieve NVM subsystem discovery information contained in the associated Discovery log page from that DDC. The CDC may:
  - a. set the Port Local Entries Only (PLEO) bit to '1' in the Log Specific Parameter (LSP) field in that Get Log Page command to request that NVM subsystem discovery information entries for only NVM subsystem ports that are presented through the same NVM subsystem port on the DDC that received the Get Log Page command be returned; and
  - b. set the All NVM Subsystem Entries (ALLSUBE) bit to '1' in the LSP field in that Get Log Page command to request that records for all NVM subsystem ports be returned;

and

2. replacing all existing NVM subsystem discovery information from the previous pull registration with the newly retrieved NVM subsystem discovery information (i.e., only the most recent NVM subsystem discovery information from that DDC is retained by the CDC).

Upon receiving another pull registration request, the CDC performs the additional pull registration or the pull de-registration by using the following sequence:

1. acknowledging that pull registration request (refer to section 8.3.2.2.2.1 for kickstart discovery); and
2. sending a Connect command to the DDC to establish an NVMe connection with that DDC ; and
3. sending a Get Log Page command to the DDC with the LID field set to 70h to retrieve NVM subsystem discovery information contained in the associated Discovery log page from that DDC. The CDC may:
  - a. set the Port Local Entries Only (PLEO) bit to '1' in the Log Specific Parameter (LSP) field of the Get Log Page command to request that NVM subsystem discovery information entries for only NVM subsystem ports that are presented through the same NVM subsystem port on the DDC that received the Get Log Page command be returned; and
  - b. set the All NVM Subsystem Entries (ALLSUBE) bit to '1' in the LSP field of the Get Log Page command to request that records for all NVM subsystem ports be returned;
 and
4. replacing all existing NVM subsystem discovery information from the previous pull registration with the newly retrieved NVM subsystem discovery information (i.e., only the most recent NVM subsystem discovery information from that DDC is retained by the CDC).

#### 8.3.2.2.2.1 Kickstart Discovery Pull Registration Requests

Figure 678 illustrates the process used during a kickstart discovery request and response sequence. The first step is to establish a TCP connection between a Direct Discovery controller (DDC) and a Centralized Discovery controller (CDC). For kickstart discovery, the CDC acts as the passive side of the TCP connection and is set to "listen" for DDC-initiated TCP connection establishment requests. The IP address used by the DDC to establish the TCP connection with the CDC may be obtained from the A or AAAA record provided in an mDNS response from the CDC, as described in section 8.3.1.3.4.

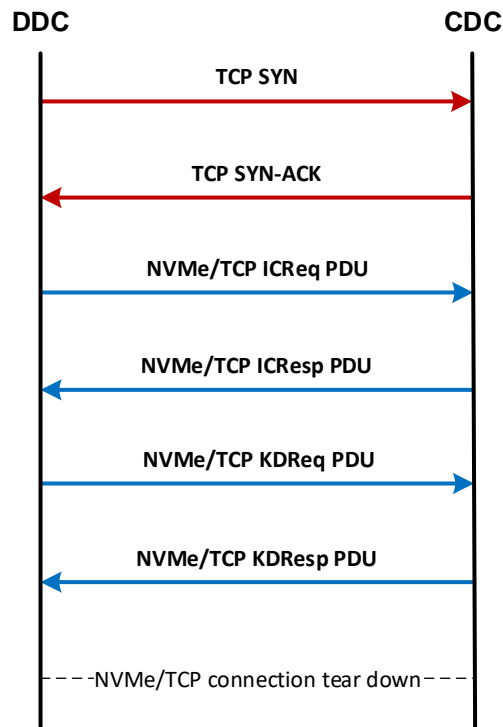
Once a TCP connection has been established, if an ICRReq NVMe/TCP PDU (refer to the Initialize Connection Request PDU section in the NVMe TCP Transport Specification) from a DDC with the Kickstart Discovery Connection (KDCONN) bit set to '1' in the FLAGS field is received by a CDC, then the CDC shall respond with an ICRResp NVMe/TCP PDU (refer to the Initialize Connection Response PDU section in the NVMe TCP Transport Specification) to establish a kickstart discovery NVMe/TCP connection.

Once a kickstart discovery NVMe/TCP connection has been established, if a KDRReq NVMe/TCP PDU (refer to the Kickstart Discovery Request PDU section in the NVMe TCP Transport Specification) from a DDC is received by a CDC, then the CDC shall respond with a KDRResp NVMe/TCP PDU (refer to the Kickstart Discovery Response PDU section in the NVMe TCP Transport Specification) to accept the pull registration request and connection parameters. The KDRResp NVMe/TCP PDU sent by the CDC shall include the NQN of that CDC in the CDC NVMe Qualified Name (CDCNQN) field. After sending a KDRResp NVMe/TCP PDU to the DDC where the Kickstart Status (KSSTAT) field is set to SUCCESS, the kickstart discovery NVMe/TCP connection should be torn down, and the CDC performs a pull registration with the DDC as described in section 8.3.2.2.2 using the connection parameters obtained from the KDRReq NVMe/TCP PDU.

If the total number of discovery information entries being registered by the pull registration (i.e., as specified by the Number of Discovery Information Entries (NUMDIE) field in the KDRReq NVMe/TCP PDU) exceeds the available capacity for new discovery information entries on the CDC, then the CDC shall send a KDRResp NVMe/TCP PDU to the DDC where the KSSTAT field is set to FAILURE and the Failure Reason (FAILRSN) field is set to Insufficient Discovery Resources.



**Figure 678: Kickstart Discovery Request and Response Sequence**

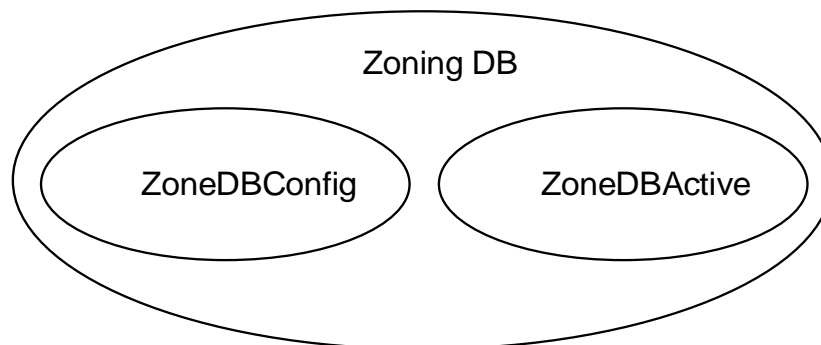


### 8.3.2.3 Fabric Zoning

#### 8.3.2.3.1 Model

A Centralized Discovery controller (CDC) may provide centralized access control services (i.e., Fabric Zoning) for an NVMe-oF IP-based fabric. CDC-based Fabric Zoning provides a way to control the Discovery log pages provided in response to a Get Log Page command issued to the CDC. Fabric Zoning should be based on a Zoning database maintained by the CDC and containing two fundamental data structures: the ZoneDBConfig and the ZoneDBActive, as shown in Figure 679.

**Figure 679: Zoning DB Abstract Representation**

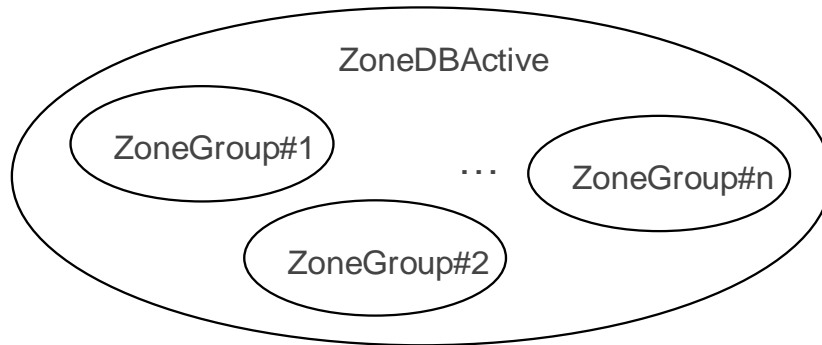


The ZoneDBActive is a list of enforced ZoneGroups. The enforcement actions include:

- Discovery log page filtering;
- Discovery Log Page Change Asynchronous Event notifications (refer to section 8.3.2.4); and
- optionally, network level restrictions (refer to section 8.3.2.3.5.2).

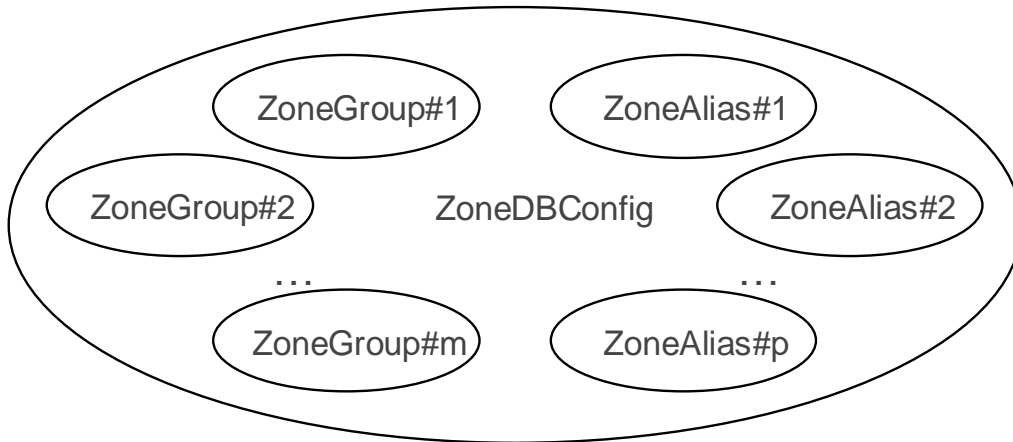
The abstract representation of the ZoneDBActive is shown in Figure 680.

**Figure 680: ZoneDBActive Abstract Representation**



The ZoneDBConfig is a list of configured ZoneGroups and ZoneAliases, as shown in Figure 681.

**Figure 681: ZoneDBConfig Abstract Representation**



**8.3.2.3.2 ZoneGroup**

A ZoneGroup is the unit of activation (i.e., a set of access control rules enforceable by the CDC). The detailed representation of a ZoneGroup is shown in Figure 682.

**Figure 682: ZoneGroup Detailed Representation**

Bytes	Description
223:00	<b>ZoneGroup Originator (ZGORIG):</b> This field contains the NQN (represented as a null-terminated string, NULL padded as necessary to the 224-byte maximum length) of either: <ul style="list-style-type: none"> <li>the CDC, if this ZoneGroup was created on the CDC; or</li> <li>the DDC that created this ZoneGroup, if this ZoneGroup was created by a DDC.</li> </ul>
253:224	<b>ZoneGroup Name (ZGNAME):</b> This field contains an ASCII encoded null-terminated string, NULL padded as necessary to the 30-byte maximum length.
255:254	<b>Number of Zones (NUMZ):</b> This field specifies the number of Zones contained in this ZoneGroup. The value of this field is represented as n.
255+LenZ <sub>1</sub> :256	<b>Zone 1 (Z1):</b> This field contains the first Zone contained in this ZoneGroup as defined in Figure 683. The length of this field is represented as LenZ <sub>1</sub> .

**Figure 682: ZoneGroup Detailed Representation**

Bytes	Description
255+LenZ <sub>1</sub> +LenZ <sub>2</sub> :256+LenZ <sub>1</sub>	<b>Zone 2 (Z2):</b> This field contains the second Zone contained in this ZoneGroup as defined in Figure 683 (if present). The length of this field is represented as LenZ <sub>2</sub> .
...	...
255+LenZ <sub>1</sub> +...+LenZ <sub>n</sub> :256+LenZ <sub>1</sub> +...+LenZ <sub>n-1</sub>	<b>Zone N (ZN):</b> This field contains the Nth Zone contained in this ZoneGroup as defined in Figure 683 (if present). The length of this field is represented as LenZ <sub>n</sub> .

A ZoneGroup is uniquely identified by the pair {ZoneGroup Name, ZoneGroup Originator}. For each ZoneGroup, the CDC shall maintain:

- a unique ZoneGroup key, used as a compact ZoneGroup identifier in the FZS and FZR commands (refer to sections 5.3.6 and 5.3.5 respectively). ZoneGroup keys should not be reused; and
- a generation number, incremented each time a ZoneGroup is updated and used by the GAZ operation (refer to section 8.3.2.3.8.2). If the value of the generation number is FFFFFFFh, then the generation number shall be set to 1h when incremented (i.e., rolls over to 1h).

### 8.3.2.3.3 Zone

A Zone is the unit of access control. Zone members belonging to the same Zone are allowed to communicate between each other, according to the rules specified in section 8.3.2.3.4.1. The detailed representation of a Zone is shown in Figure 683.

**Figure 683: Zone Detailed Representation**

Bytes	Description
123:00	<b>Zone Name (ZNAME):</b> This field contains an ASCII encoded null-terminated string, NULL padded as necessary to the 124-byte maximum length.
125:124	<b>Number of Zone Members (NUMZM):</b> This field specifies the number of Zone members contained in this Zone. The value of this field is represented as n.
127:126	<b>Number of Zone Properties (NUMZP):</b> This field specifies the number of Zone properties contained in this Zone. The value of this field is represented as k.
383:128	<b>Zone Member 1 (ZM1):</b> This field contains the first Zone member contained in this Zone.
639:384	<b>Zone Member 2 (ZM2):</b> This field contains the second Zone member contained in this Zone.
...	...
256*n+127:256*n-128	<b>Zone Member n (ZMn):</b> This field contains the Nth Zone member contained in this Zone (if present).
256*n+127+LenP <sub>1</sub> :256*n+128	<b>Zone Property 1 (ZP1):</b> This field contains the first Zone property contained in this Zone (if present). The length of this field is represented as LenP <sub>1</sub> .
256*n+127+LenP <sub>1</sub> +LenP <sub>2</sub> : 256*n+128+LenP <sub>1</sub>	<b>Zone Property 2 (ZP2):</b> This field contains the second Zone property contained in this Zone (if present). The length of this field is represented as LenP <sub>2</sub> .
...	...
256*n+127+LenP <sub>1</sub> +LenP <sub>2</sub> +...+LenP <sub>k</sub> : 256*n+128+LenP <sub>1</sub> +LenP <sub>2</sub> +...+LenP <sub>k-1</sub>	<b>Zone Property k (ZPk):</b> This field contains the Kth Zone property contained in this Zone (if present). The length of this field is represented as LenP <sub>k</sub> .

### 8.3.2.3.4 Zone Members

#### 8.3.2.3.4.1 Overview

Zone members are represented as type-length-value (TLV) data structures. Figure 684 shows the defined Zone member types. If a Zone member type does not include a particular element in the pairing for that Zone member type, then the element values of that type shall not be used for enforcement of Zoning restrictions for that Zone (e.g., Zone member type 1h does not include IP addresses in the enforcement of that Zone member type).

**Figure 684: Zone Member Types**

Type	Definition	Reference
1h	The pair {NQN, Role}	8.3.2.3.4.2
2h	The pair {(NQN, IP, Protocol), Role}	8.3.2.3.4.3
3h	The pair {(NQN, IP, Protocol, IP Protocol Port), Role}	8.3.2.3.4.4
4h	The pair {(NQN, IP, Protocol, IP Protocol Port, PortID), Role}	8.3.2.3.4.5
5h	The pair {(NQN, PortID), Role}	8.3.2.3.4.6
6h	The pair {(NQN, Protocol, PortID, ADRFAM), Role}	8.3.2.3.4.7
11h	The pair {(IP, Protocol), Role}	8.3.2.3.4.8
12h	The pair {(IP, Protocol, IP Protocol Port), Role}	8.3.2.3.4.9
EFh	A ZoneAlias name	8.3.2.3.4.10
F0h to FEh	Vendor specific	
All others	Reserved	

The members of a Zone may communicate with each other using the following rules:

- hosts may communicate with NVM subsystems;
- NVM subsystems may communicate with hosts;
- hosts shall not communicate with hosts; and
- NVM subsystems shall not communicate with NVM subsystems.

A Zone of a ZoneGroup belonging to the ZoneDBConfig may use all defined Zone member types. When a ZoneGroup belonging to the ZoneDBConfig is activated and becomes part of the ZoneDBActive, all ZoneAlias name Zone members are resolved in the group of NVMe entities referenced by the ZoneAlias name.

#### 8.3.2.3.4.2 {NQN, Role} Zone Member Type (Type 1h)

This Zone member type identifies all fabric interfaces, all IP protocols (e.g., TCP or UDP), and all IP protocol ports (e.g., TCP port 4420) that may be used by the NVMe-oF entity identified by the Zone member's NQN. The format of this Zone member type is shown in Figure 685.

**Figure 685: {NQN, Role} Zone Member Format**

Bytes	Description
00	<b>Zone Member Type (ZMTYPE):</b> This field specifies the Zone member type and shall be set to 1h.
02:01	<b>Zone Member Length (ZMLEN):</b> This field specifies the length in bytes of this Zone member type and shall be set to 100h.
03	<b>Zone Member Role (ZMROLE):</b> This field specifies what type of entity the Zone member is. This field shall be set to one of the following values: <ul style="list-style-type: none"> <li>• 1h: host; or</li> <li>• 2h: NVM subsystem.</li> </ul>
31:04	Reserved
255:32	<b>Zone Member NQN (ZMNQN):</b> This field specifies the NQN (represented as a null-terminated string, NULL padded as necessary to the 224-byte maximum length) of the referenced NVMe-oF entity.

### 8.3.2.3.4.3 {(NQN, IP, Protocol), Role} Zone Member Type (Type 2h)

This Zone member type identifies a specific fabric interface (i.e., through the IP address) and the specific IP protocol (e.g., TCP) used by the NVMe-oF entity identified by the Zone member's NQN over that fabric interface. The format of this Zone member type is shown in Figure 686.

**Figure 686: {(NQN+IP+Protocol), Role} Zone Member Format**

Bytes	Description										
00	<b>Zone Member Type (ZMTYPE):</b> This field specifies the Zone member type and shall be set to 2h.										
02:01	<b>Zone Member Length (ZMLEN):</b> This field specifies the length in bytes of this Zone member type and shall be set to 100h.										
03	<b>Zone Member Role (ZMROLE):</b> This field specifies what type of entity the Zone member is. This field shall be set to one of the following values: <ul style="list-style-type: none"> <li>• 1h: host; or</li> <li>• 2h: NVM subsystem.</li> </ul>										
04	<b>Zone Member Address Family (ZMADFAM):</b> This field specifies the IP address family. This field shall be set to one of the following values: <ul style="list-style-type: none"> <li>• 1h: IPv4 address family; or</li> <li>• 2h: IPv6 address family.</li> </ul>										
05	<b>Zone Member IP Protocol (ZMIPPROTO):</b> This field specifies the IP protocol. This field shall be set to one of the following values: <ul style="list-style-type: none"> <li>• 6h: TCP; or</li> <li>• 11h: UDP.</li> </ul>										
07:06	Reserved										
23:08	<p><b>Zone Member Transport Address (ZMTRADDR):</b> This field specifies the IP address. An IPv6 address is encoded in binary as follows:</p> <table border="1" style="margin-left: 40px;"> <thead> <tr> <th>Bytes</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:00</td> <td><b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.</td> </tr> </tbody> </table> <p>An IPv4 address is encoded in binary as follows:</p> <table border="1" style="margin-left: 40px;"> <thead> <tr> <th>Bytes</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>03:00</td> <td><b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.</td> </tr> <tr> <td>15:04</td> <td>Reserved</td> </tr> </tbody> </table>	Bytes	Description	15:00	<b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.	Bytes	Description	03:00	<b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.	15:04	Reserved
Bytes	Description										
15:00	<b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.										
Bytes	Description										
03:00	<b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.										
15:04	Reserved										
31:24	Reserved										
255:32	<b>Zone Member NQN (ZMNQN):</b> This field specifies the NQN (represented as a null-terminated string, NULL padded as necessary to the 224-byte maximum length) of the referenced NVMe-oF entity.										

### 8.3.2.3.4.4 {(NQN, IP, Protocol, IP Protocol Port), Role} Zone Member Type (Type 3h)

This Zone member type identifies a specific fabric interface (i.e., through the IP address), the specific IP protocol (e.g., TCP), and IP protocol port (e.g., TCP port 4420) used by the NVMe-oF entity identified by the Zone member's NQN over that fabric interface. The format of this Zone member type is shown in Figure 687.

**Figure 687: {(NQN+IP+Protocol+IP Protocol Port), Role} Zone Member Format**

Bytes	Description
00	<b>Zone Member Type (ZMTYPE):</b> This field specifies the Zone member type and shall be set to 3h.
02:01	<b>Zone Member Length (ZMLEN):</b> This field specifies the length in bytes of this Zone member type and shall be set to 100h.
03	<b>Zone Member Role (ZMROLE):</b> This field specifies what type of entity the Zone member is. This field shall be set to one of the following values: <ul style="list-style-type: none"> <li>• 1h: host; or</li> <li>• 2h: NVM subsystem.</li> </ul>

**Figure 687: {(NQN+IP+Protocol+IP Protocol Port), Role} Zone Member Format**

Bytes	Description										
04	<p><b>Zone Member Address Family (ZMADFAM):</b> This field specifies the IP address family. This field shall be set to one of the following values:</p> <ul style="list-style-type: none"> <li>• 1h: IPv4 address family; or</li> <li>• 2h: IPv6 address family.</li> </ul>										
05	<p><b>Zone Member IP Protocol (ZMIPPROTO):</b> This field specifies the IP protocol. This field shall be set to one of the following values:</p> <ul style="list-style-type: none"> <li>• 6h: TCP; or</li> <li>• 11h: UDP.</li> </ul>										
07:06	<p><b>Zone Member Transport Service Identifier (ZMTRSVCID):</b> This field specifies the TCP or UDP port.</p>										
23:08	<p><b>Zone Member Transport Address (ZMTRADDR):</b> This field specifies the IP address. An IPv6 address is encoded in binary as follows:</p> <table border="1" data-bbox="456 663 1271 726"> <thead> <tr> <th>Bytes</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:00</td> <td><b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.</td> </tr> </tbody> </table> <p>An IPv4 address is encoded in binary as follows:</p> <table border="1" data-bbox="456 785 1271 869"> <thead> <tr> <th>Bytes</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>03:00</td> <td><b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.</td> </tr> <tr> <td>15:04</td> <td>Reserved</td> </tr> </tbody> </table>	Bytes	Description	15:00	<b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.	Bytes	Description	03:00	<b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.	15:04	Reserved
Bytes	Description										
15:00	<b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.										
Bytes	Description										
03:00	<b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.										
15:04	Reserved										
31:24	Reserved										
255:32	<p><b>Zone Member NQN (ZMNQN):</b> This field specifies the NQN (represented as a null-terminated string, NULL padded as necessary to the 224-byte maximum length) of the referenced NVMe-oF entity.</p>										

**8.3.2.3.4.5 {(NQN, IP, Protocol, IP Protocol Port, PortID), Role} Zone Member Type (Type 4h)**

This Zone member type identifies a specific fabric interface (i.e., through the IP address), the specific IP protocol (e.g., TCP), IP protocol port (e.g., TCP port 4420), and PortID used by the NVMe-oF entity identified by the Zone member's NQN over that fabric interface. The format of this Zone member type is shown in Figure 688.

**Figure 688: {(NQN+IP+Protocol+IP Protocol Port+PortID), Role} Zone Member Format**

Bytes	Description
00	<p><b>Zone Member Type (ZMTYPE):</b> This field specifies the Zone member type and shall be set to 4h.</p>
02:01	<p><b>Zone Member Length (ZMLEN):</b> This field specifies the length in bytes of this Zone member type and shall be set to 100h.</p>
03	<p><b>Zone Member Role (ZMROLE):</b> This field specifies what type of entity the Zone member is. This field shall be set to one of the following values:</p> <ul style="list-style-type: none"> <li>• 1h: host; or</li> <li>• 2h: NVM subsystem.</li> </ul>
04	<p><b>Zone Member Address Family (ZMADFAM):</b> This field specifies the IP address family. This field shall be set to one of the following values:</p> <ul style="list-style-type: none"> <li>• 1h: IPv4 address family; or</li> <li>• 2h: IPv6 address family.</li> </ul>
05	<p><b>Zone Member IP Protocol (ZMIPPROTO):</b> This field specifies the IP protocol. This field shall be set to one of the following values:</p> <ul style="list-style-type: none"> <li>• 6h: TCP; or</li> <li>• 11h: UDP.</li> </ul>
07:06	<p><b>Zone Member Transport Service Identifier (ZMTRSVCID):</b> This field specifies the TCP or UDP port.</p>

**Figure 688: {(NQN+IP+Protocol+IP Protocol Port+PortID), Role} Zone Member Format**

Bytes	Description					
23:08	<b>Zone Member Transport Address (ZMTRADDR):</b> This field specifies the IP address. An IPv6 address is encoded in binary as follows:					
	<table border="1"> <thead> <tr> <th>Bytes</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:00</td> <td><b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.</td> </tr> </tbody> </table>	Bytes	Description	15:00	<b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.	
	Bytes	Description				
	15:00	<b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.				
An IPv4 address is encoded in binary as follows:						
<table border="1"> <thead> <tr> <th>Bytes</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>03:00</td> <td><b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.</td> </tr> <tr> <td>15:04</td> <td>Reserved</td> </tr> </tbody> </table>	Bytes	Description	03:00	<b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.	15:04	Reserved
Bytes	Description					
03:00	<b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.					
15:04	Reserved					
25:24	<b>Zone Member PortID (ZMPORTID):</b> This field specifies the NVM subsystem PortID.					
31:26	Reserved					
255:32	<b>Zone Member NQN (ZMNQN):</b> This field specifies the NQN (represented as a null-terminated string, NULL padded as necessary to the 224-byte maximum length) of the referenced NVMe-oF entity.					

#### 8.3.2.3.4.6 {(NQN, PortID), Role} Zone Member Type (Type 5h)

This Zone member type identifies a specific PortID used by the NVMe-oF entity identified by the Zone member's NQN. The format of this Zone member type is shown in Figure 689.

**Figure 689: {(NQN+PortID), Role} Zone Member Format**

Bytes	Description
00	<b>Zone Member Type (ZMTYPE):</b> This field specifies the Zone member type and shall be set to 5h.
02:01	<b>Zone Member Length (ZMLEN):</b> This field specifies the length in bytes of this Zone member type and shall be set to 100h.
03	<b>Zone Member Role (ZMROLE):</b> This field specifies what type of entity the Zone member is. This field shall be set to one of the following values: <ul style="list-style-type: none"> <li>• 1h: host; or</li> <li>• 2h: NVM subsystem.</li> </ul>
05:04	<b>Zone Member PortID (ZMPORTID):</b> This field specifies the NVM subsystem PortID.
31:06	Reserved
255:32	<b>Zone Member NQN (ZMNQN):</b> This field specifies the NQN (represented as a null-terminated string, NULL padded as necessary to the 224-byte maximum length) of the referenced NVMe-oF entity.

#### 8.3.2.3.4.7 {(NQN, Protocol, PortID, ADRFAM), Role} Zone Member Type (Type 6h)

This Zone member type identifies the specific IP protocol (e.g., TCP), PortID, and Address Family used by the NVMe-oF entity identified by the Zone member's NQN over that fabric interface. The format of this Zone member type is shown in Figure 690.

**Figure 690: {(NQN+Protocol+PortID+ADRFAM), Role} Zone Member Format**

Bytes	Description
00	<b>Zone Member Type (ZMTYPE):</b> This field specifies the Zone member type and shall be set to 6h.
02:01	<b>Zone Member Length (ZMLEN):</b> This field specifies the length in bytes of this Zone member type and shall be set to 100h.
03	<b>Zone Member Role (ZMROLE):</b> This field specifies what type of entity the Zone member is. This field shall be set to one of the following values: <ul style="list-style-type: none"> <li>• 1h: host; or</li> <li>• 2h: NVM subsystem.</li> </ul>
04	<b>Zone Member Address Family (ZMADRFAM):</b> This field specifies the IP address family. This field shall be set to one of the following values: <ul style="list-style-type: none"> <li>• 1h: IPv4 address family; or</li> <li>• 2h: IPv6 address family.</li> </ul>

**Figure 690: {(NQN+Protocol+PortID+ADRFAM), Role} Zone Member Format**

Bytes	Description
05	<b>Zone Member IP Protocol (ZMIPPROTO):</b> This field specifies the IP protocol. This field shall be set to one of the following values: <ul style="list-style-type: none"> <li>• 6h: TCP; or</li> <li>• 11h: UDP.</li> </ul>
07:06	<b>Zone Member PortID (ZMPORTID):</b> This field specifies the NVM subsystem PortID.
31:08	Reserved
255:32	<b>Zone Member NQN (ZMNQN):</b> This field specifies the NQN (represented as a null-terminated string, NULL padded as necessary to the 224-byte maximum length) of the referenced NVMe-oF entity.

**8.3.2.3.4.8 {(IP, Protocol), Role} Zone Member Type (Type 11h)**

This Zone member type identifies the fabric interface (i.e., through the IP address) of an NVMe-oF entity and the specific IP protocol (e.g., TCP) used by the NVMe-oF entity over that fabric interface. The format of this Zone member type is shown in Figure 691.

**Figure 691: {(IP+Protocol), Role} Zone Member Format**

Bytes	Description										
00	<b>Zone Member Type (ZMTYPE):</b> This field specifies the Zone member type and shall be set to 11h.										
02:01	<b>Zone Member Length (ZMLEN):</b> This field specifies the length in bytes of this Zone member type and shall be set to 100h.										
03	<b>Zone Member Role (ZMROLE):</b> This field specifies what type of entity the Zone member is. This field shall be set to one of the following values: <ul style="list-style-type: none"> <li>• 1h: host; or</li> <li>• 2h: NVM subsystem.</li> </ul>										
04	<b>Zone Member Address Family (ZMADRFAM):</b> This field specifies the IP address family. This field shall be set to one of the following values: <ul style="list-style-type: none"> <li>• 1h: IPv4 address family; or</li> <li>• 2h: IPv6 address family.</li> </ul>										
05	<b>Zone Member IP Protocol (ZMIPPROTO):</b> This field specifies the IP protocol. This field shall be set to one of the following values: <ul style="list-style-type: none"> <li>• 6h: TCP; or</li> <li>• 11h: UDP.</li> </ul>										
07:06	Reserved										
23:08	<b>Zone Member Transport Address (ZMTRADDR):</b> This field specifies the IP address. An IPv6 address is encoded in binary as follows: <table border="1" data-bbox="456 1404 1271 1465"> <thead> <tr> <th>Bytes</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:00</td> <td><b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.</td> </tr> </tbody> </table> An IPv4 address is encoded in binary as follows: <table border="1" data-bbox="456 1524 1271 1612"> <thead> <tr> <th>Bytes</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>03:00</td> <td><b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.</td> </tr> <tr> <td>15:04</td> <td>Reserved</td> </tr> </tbody> </table>	Bytes	Description	15:00	<b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.	Bytes	Description	03:00	<b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.	15:04	Reserved
Bytes	Description										
15:00	<b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.										
Bytes	Description										
03:00	<b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.										
15:04	Reserved										
255:24	Reserved										

**8.3.2.3.4.9 {(IP, Protocol, IP Protocol Port), Role} Zone Member Type (Type 12h)**

This Zone member type identifies the fabric interface (i.e., through the IP address) of an NVMe-oF entity, the specific IP protocol (e.g., TCP), and IP protocol port (e.g., TCP port 4420) used by the NVMe-oF entity over that fabric interface. The format of this Zone member type is shown in Figure 692.



**Figure 692: {(IP+Protocol+IP Protocol Port), Role} Zone Member Format**

Bytes	Description					
00	<b>Zone Member Type (ZMTYPE):</b> This field specifies the Zone member type and shall be set to 12h.					
02:01	<b>Zone Member Length (ZMLEN):</b> This field specifies the length in bytes of this Zone member type and shall be set to 100h.					
03	<b>Zone Member Role (ZMROLE):</b> This field specifies what type of entity the Zone member is. This field shall be set to one of the following values: <ul style="list-style-type: none"> <li>• 1h: host; or</li> <li>• 2h: NVM subsystem.</li> </ul>					
04	<b>Zone Member Address Family (ZMADFAM):</b> This field specifies the IP address family. This field shall be set to one of the following values: <ul style="list-style-type: none"> <li>• 1h: IPv4 address family; or</li> <li>• 2h: IPv6 address family.</li> </ul>					
05	<b>Zone Member IP Protocol (ZMIPPROTO):</b> This field specifies the IP protocol. This field shall be set to one of the following values: <ul style="list-style-type: none"> <li>• 6h: TCP; or</li> <li>• 11h: UDP.</li> </ul>					
07:06	<b>Zone Member Transport Service Identifier (ZMTRSVCID):</b> This field specifies the TCP or UDP port.					
23:08	<b>Zone Member Transport Address (ZMTRADDR):</b> This field specifies the IP address. An IPv6 address is encoded in binary as follows: <table border="1" data-bbox="456 884 1271 947"> <thead> <tr> <th>Bytes</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>15:00</td> <td><b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.</td> </tr> </tbody> </table>	Bytes	Description	15:00	<b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.	
	Bytes	Description				
	15:00	<b>IPv6 Address (IPV6ADDR):</b> This field contains an IPv6 address.				
	An IPv4 address is encoded in binary as follows: <table border="1" data-bbox="456 1003 1271 1087"> <thead> <tr> <th>Bytes</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>03:00</td> <td><b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.</td> </tr> <tr> <td>15:04</td> <td>Reserved</td> </tr> </tbody> </table>	Bytes	Description	03:00	<b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.	15:04
Bytes	Description					
03:00	<b>IPv4 Address (IPV4ADDR):</b> This field contains an IPv4 address.					
15:04	Reserved					
255:24	Reserved					

### 8.3.2.3.4.10 {ZoneAlias} Zone Member Type

This Zone member type is used to identify a ZoneAlias in a Zone definition. The format of this Zone member type is shown in Figure 693.

**Figure 693: {ZoneAlias} Zone Member Format**

Bytes	Description
00	<b>Zone Member Type (ZMTYPE):</b> This field specifies the Zone member type and shall be set to EFh.
02:01	<b>Zone Member Length (ZMLEN):</b> This field specifies the length in bytes of this Zone member type and shall be set to 100h.
31:03	Reserved
155:32	<b>Zone Member ZoneAlias Name (ZMZANAME):</b> This field specifies an ASCII encoded null-terminated string, NULL padded as necessary to the 124-byte maximum length.
255:156	Reserved

### 8.3.2.3.5 Zone Properties

#### 8.3.2.3.5.1 Overview

Zone properties are represented as type-length-value (TLV) data structures. Unrecognized Zone properties may be ignored. Figure 694 shows the defined Zone property types.

**Figure 694: Zone Property Types**

Type	Definition
1h	Fabric Enforced Zone

**Figure 694: Zone Property Types**

Type	Definition
FEh to F0h	Vendor specific
All others	Reserved

**8.3.2.3.5.2 Fabric Enforced Zone Property**

This Zone property specifies if a Zone is intended to be enforced through packet-by-packet network level restrictions. The format of this Zone property is shown in Figure 695.

**Figure 695: Fabric Enforced Zone Property Format**

Bytes	Description
00	<b>Zone Property Type (ZPTYPE):</b> This field specifies the Zone property type and shall be set to 1h.
02:01	<b>Zone Property Length (ZPLEN):</b> This field specifies the length in bytes of this Zone property type and shall be set to 4h.
03	<p><b>Zone Property Value (ZPVAL):</b> This field specifies the value of this Zone property. This field shall be set to the following value:</p> <ul style="list-style-type: none"> <li>1h: Network level enforcement requested.</li> </ul> <p>All other values are ignored.</p>

**8.3.2.3.6 ZoneAlias**

A ZoneAlias is a convenient grouping of NVMe entities identified and is referenceable by a ZoneAlias name. The detailed representation of a ZoneAlias is shown in Figure 696.

**Figure 696: ZoneAlias Detailed Representation**

Bytes	Description
123:00	<b>ZoneAlias Name (ZNAME):</b> This field contains an ASCII encoded null-terminated string, NULL padded as necessary to the 124-byte maximum length.
125:124	<b>Number of ZoneAlias Members (NUMZAM):</b> This field specifies the number of ZoneAlias members contained in this ZoneAlias. The value of this field is represented as n.
127:126	Reserved
ZoneAlias Member List	
383:128	<b>ZoneAlias Member 1:</b> This field contains the first ZoneAlias member contained in this ZoneAlias.
639:384	<b>ZoneAlias Member 2:</b> This field contains the second ZoneAlias member contained in this ZoneAlias (if present).
...	...
256*n+127: 256*n-128	<b>ZoneAlias Member n:</b> This field contains the last ZoneAlias member contained in this ZoneAlias (if present).

**8.3.2.3.7 ZoneAlias Members****8.3.2.3.7.1 Overview**

ZoneAlias members are represented as type-length-value (TLV) data structures. ZoneAlias members may be any of the Zone member types specified in Figure 684 except the ZoneAlias Zone member type (i.e., ZMTYPE field set to EFh).

**8.3.2.3.8 Zoning Operations****8.3.2.3.8.1 Overview**

ZoneGroups are data structures maintained and managed (i.e., created, read, modified, or deleted) in the Zoning database of the CDC. During access to a ZoneGroup (e.g., the ZoneGroup is created or modified) through an administrative interface (e.g., an administrative interface on the CDC outside the scope of this

specification or the protocol defined in this specification), the ZoneGroup may become locked to access from any other interface.

Management of ZoneGroups generally happens through an administrative interface on the CDC. Push model Direct Discovery controllers (DDCs) may be able to manage their own ZoneGroups (i.e., ZoneGroups having the DDC NQN as ZoneGroup Originator) as specified in section 8.3.2.3.8.2. Pull model DDCs may be able to manage their own ZoneGroups (i.e., ZoneGroups having the DDC NQN as ZoneGroup Originator) as specified in section 8.3.2.3.8.3.

By default, a ZoneGroup should be accessible only to an administrative interface outside the scope of this specification on the CDC and to the ZoneGroup originator DDC through the protocol defined in this specification.

If Fabric Zoning is not enabled on the CDC, then that CDC:

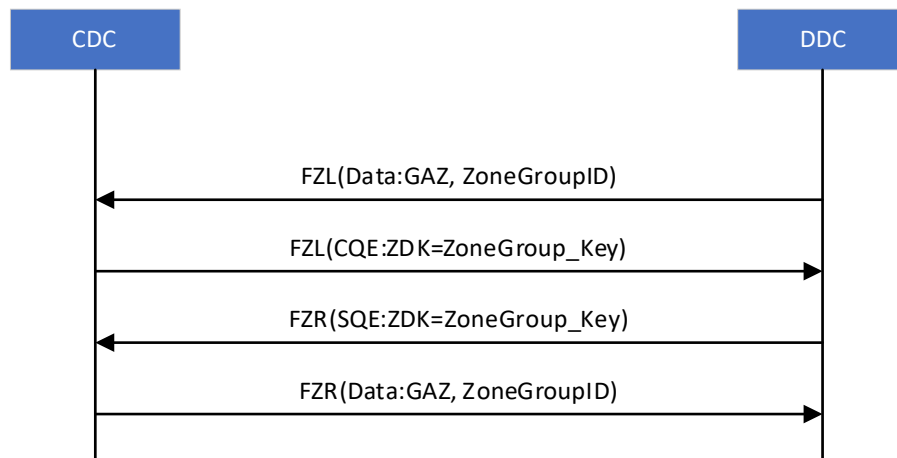
- shall abort all Fabric Zoning commands (i.e., any Fabric Zoning command that is issued as part of a Zoning operation) issued by a push model DDC with a status code of Requested Function Disabled; and
- shall not react to Pull Model DDC Request asynchronous event notifications issued by a pull model DDC.

### 8.3.2.3.8.2 Push Model DDC Operations

#### 8.3.2.3.8.2.1 Get Active ZoneGroup (GAZ)

The Get Active ZoneGroup (GAZ) operation allows a DDC to retrieve from the CDC an active ZoneGroup associated with the DDC initiating the GAZ operation. For a push model DDC, the GAZ operation is mapped to an FZL command to lookup the key of the ZoneGroup to retrieve in ZoneDBActive in that CDC, followed by one or more FZR commands to retrieve that ZoneGroup from the CDC, as shown in Figure 697.

**Figure 697: GAZ for Push Model DDC**



The identifier of the ZoneGroup to get is provided in the FZL buffer, as shown in Figure 698.

**Figure 698: FZL Data for GAZ**

Bytes	Description
00	<b>Operation Type (OTYP):</b> 1h (i.e., Lookup for Get Active ZoneGroup)
03:01	Reserved
227:04	<b>ZoneGroup Originator (ZGORIG)</b>
257:228	<b>ZoneGroup Name (ZGNAME)</b>

The FZL command returns the key of the ZoneGroup to retrieve as a Zoning Data key value in the Completion Queue. For the FZL command of a GAZ operation:

- if the requested ZoneGroup does not exist on the CDC, then the CDC shall abort the command with a status code of Zoning Data Structure Not Found; or
- if the requested ZoneGroup is locked on the CDC (i.e., another administrative interface is modifying that ZoneGroup), then the CDC shall abort the command with a status code of Zoning Data Structure Locked.

The Zoning Data key value returned by the FZL command is used in Command Dword 10 of an FZR command (refer to Figure 506) to retrieve that ZoneGroup or a fragment of that ZoneGroup. The ZoneGroup definition is retrieved through one or more subsequent FZR commands and is returned in the FZR buffer, as shown in Figure 699. The FZR completion queue entry sending the last fragment shall have the Last Fragment (LF) bit set to '1' in Completion Queue Entry Dword 0 (refer to Figure 510).

**Figure 699: FZR Data for GAZ**

Bytes	Description
00	<b>Operation Type (OTYP):</b> 2h (i.e., Get Active ZoneGroup for a push model DDC)
03:01	Reserved
07:04	<b>ZoneGroup Generation Number (ZGGN)</b>
11:08	<b>ZoneGroup Fragment Length (ZGFL)</b>
15:12	Reserved
ZGFL+15:16	<b>ZoneGroup Fragment (ZGF)</b>

For the FZR command of a GAZ response operation:

- if the ZoneGroup identified by the specified ZoneGroup key does not exist on the CDC, then the CDC shall abort the command with a status code of Zoning Data Structure Not Found; or
- if the ZoneGroup identified by the specified ZoneGroup key is locked on the CDC (i.e., another administrative interface is modifying that ZoneGroup), then the CDC shall abort the command with a status code of Zoning Data Structure Locked.

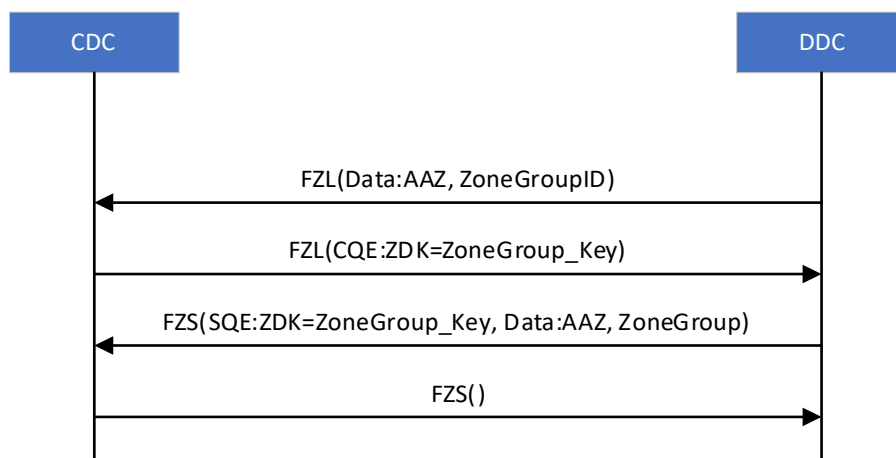
When a ZoneGroup is transferred in multiple fragments, the receiver shall verify that the ZoneGroup generation number stays constant across all FZR commands. If the ZoneGroup generation number changes, then the GAZ operation shall be aborted. The DDC shall not process received ZoneGroup information until the full ZoneGroup (i.e., all of the fragments of the ZoneGroup) is received.

The CDC may enforce access restrictions to the Zoning data structures. In this case, the CDC shall check if the DDC issuing the FZL command or FZR command is authorized to read the ZoneGroup specified in the FZL data (e.g., if the CDC allows access to a ZoneGroup only to the DDC that created that ZoneGroup, verify that the ZoneGroup Originator field matches the NQN contained in the HOSTNQN field of the Connect command sent from the DDC to that CDC). If that DDC is not authorized to access the specified ZoneGroup, then the CDC shall abort the FZL command and the FZR command with a status code of ZoneGroup Originator Invalid.

#### **8.3.2.3.8.2.2 Add/Replace Active ZoneGroup (AAZ)**

The Add/Replace Active ZoneGroup (AAZ) operation allows a DDC to add or replace in the CDC an active ZoneGroup associated with the DDC initiating the AAZ operation. For a push model DDC, the AAZ operation is mapped to an FZL command to lookup the key of the ZoneGroup to add or replace in ZoneDBActive in that CDC, followed by one or more FZS commands to provide the CDC with the ZoneGroup to add or replace, as shown in Figure 700.

**Figure 700: AAZ for Push Model DDC**



The identifier of the ZoneGroup to add or replace is provided in the FZL buffer, as shown in Figure 701.

**Figure 701: FZL Data for AAZ**

Bytes	Description
00	<b>Operation Type (OTYP):</b> 3h (i.e., Lookup for Add/Replace Active ZoneGroup)
03:01	Reserved
227:04	<b>ZoneGroup Originator (ZGORIG)</b>
257:228	<b>ZoneGroup Name (ZGNAME)</b>

The FZL command returns the key of the ZoneGroup to add or replace as a Zoning Data key value in the Completion Queue. For the FZL command of an AAZ operation:

- if the ZoneGroup that matches the specified FZL data exists in ZoneDBActive in that CDC and is locked by another administrative interface, then the CDC shall abort the command with a status code of Zoning Data Structure Locked;
- if the ZoneGroup that matches the specified FZL data exists in ZoneDBActive in that CDC and is not locked by another administrative interface, then that ZoneGroup should be locked by the submitting DDC and its key shall be returned as the Zoning Data key in the Completion Queue; or
- if the ZoneGroup that matches the specified FZL data does not exist in ZoneDBActive in that CDC, then an empty ZoneGroup having the provided identifier shall be created in ZoneDBActive in that CDC, that ZoneGroup should be locked by the submitting DDC, and its key shall be returned as the Zoning Data key in the Completion Queue.

Successful completion of the FZL command for the AAZ operation results in the identified ZoneGroup on the CDC being locked by the DDC performing the operation.

The ZoneGroup to add or replace is provided in the FZS buffer of subsequent FZS commands with the appropriate Zoning Data key in Command Dword 10 (refer to Figure 512) and is transferred in one or more fragments, as needed, as shown in Figure 702. The FZS command sending the last fragment shall have the Last Fragment (LF) bit set to '1' in Command Dword 12 (refer to Figure 514).

**Figure 702: FZS Data for AAZ**

Bytes	Description
00	<b>Operation Type (OTYP):</b> 4h (i.e., Add/Replace Active ZoneGroup for a push model DDC)
03:01	Reserved
NUMD*4+03:04	<b>ZoneGroup Fragment (ZGF)</b>

For the FZS command of an AAZ request operation:

- if the ZoneGroup identified by the specified ZoneGroup key does not exist on the CDC, then the CDC shall abort the command with a status code of Zoning Data Structure Not Found; or
- if the ZoneGroup identified by the specified ZoneGroup key is locked by another entity on the CDC (i.e., another administrative interface is modifying that ZoneGroup), then the CDC shall abort the command with a status code of Zoning Data Structure Locked.

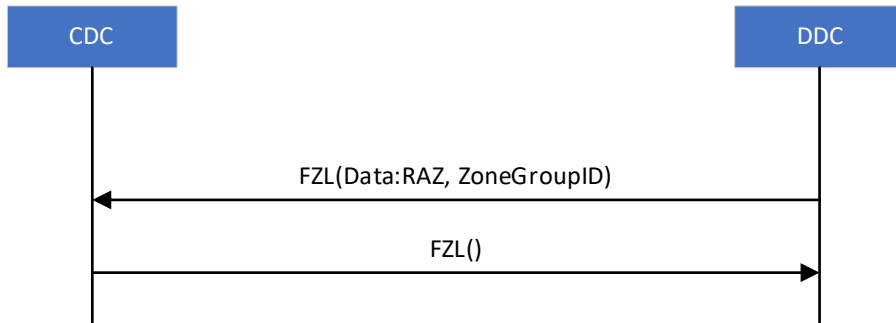
The CDC shall update a ZoneGroup and increment its generation number only upon successful reception of the full ZoneGroup (i.e., all of the fragments of the ZoneGroup). Successful receipt of the full ZoneGroup for the AAZ operation shall unlock the identified ZoneGroup on the CDC. If the full ZoneGroup is not received within 30 seconds from the establishment of the lock (during processing of the related FZL command), then all the received data shall be discarded and the lock shall be released (i.e., ZoneDBActive in that CDC is not changed).

The CDC may enforce access restrictions to the Zoning data structures. In this case, the CDC shall check if the DDC issuing the FZL command or FZS command is authorized to write the ZoneGroup specified in the FZL data (e.g., if the CDC allows access to a ZoneGroup only to the DDC that created that ZoneGroup, verify that the ZoneGroup Originator field matches the NQN contained in the HOSTNQN field of the Connect command sent from the DDC to that CDC). If that DDC is not authorized to access the specified ZoneGroup, then the CDC shall abort the FZL command and the FZS command with a status code of ZoneGroup Originator Invalid.

**8.3.2.3.8.2.3 Remove Active ZoneGroup (RAZ)**

The Remove Active ZoneGroup (RAZ) operation allows a DDC to remove from the CDC an active ZoneGroup associated with the DDC initiating the RAZ operation. For a push model DDC, the RAZ operation is mapped to an FZL command to provide to the CDC with the identifier of the ZoneGroup to remove, as shown in Figure 703.

**Figure 703: RAZ for Push Model DDC**



The identifier of the ZoneGroup to remove is provided in the FZL buffer, as shown in Figure 704.

**Figure 704: FZL Data for RAZ**

Bytes	Description
00	<b>Operation Type (OTYP):</b> 5h (i.e., Lookup for Remove Active ZoneGroup)
03:01	Reserved
227:04	<b>ZoneGroup Originator (ZGORIG)</b>
257:228	<b>ZoneGroup Name (ZGNAME)</b>

For the FZS command of an RAZ request operation:

- if the requested ZoneGroup does not exist on the CDC, then the CDC shall abort the command with a status code of Zoning Data Structure Not Found; or
- if the requested ZoneGroup is locked on the CDC (i.e., another administrative interface is modifying that ZoneGroup), then the CDC shall abort the command with a status code of Zoning Data Structure Locked.

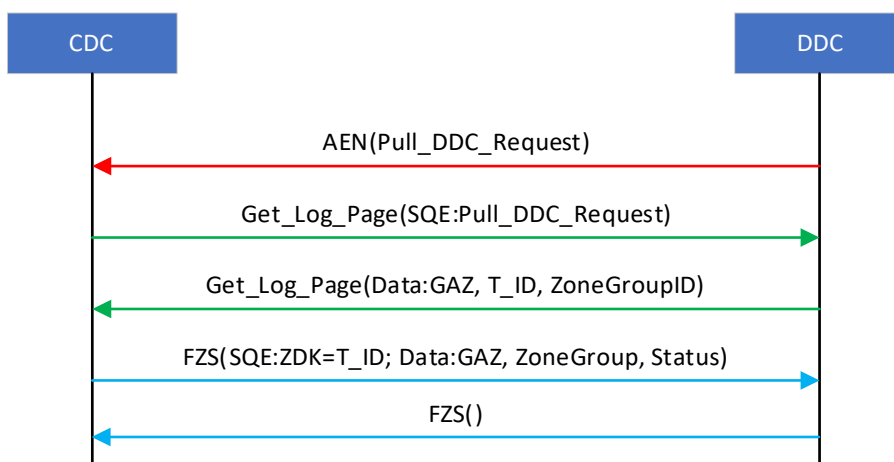
The CDC may enforce access restrictions to the Zoning data structures. In this case, the CDC shall check if the DDC issuing the FZL command is authorized to remove the ZoneGroup specified in the FZL data (e.g., if the CDC allows access to a ZoneGroup only to the DDC that created that ZoneGroup, verify that the ZoneGroup Originator field matches the NQN contained in the HOSTNQN field of the Connect command sent from the DDC to that CDC). If that DDC is not authorized to access the specified ZoneGroup, then the CDC shall abort the FZL command with a status code of ZoneGroup Originator Invalid.

### 8.3.2.3.8.3 Pull Model DDC Operations

#### 8.3.2.3.8.3.1 Get Active ZoneGroup (GAZ)

The Get Active ZoneGroup (GAZ) operation allows a DDC to retrieve from the CDC an active ZoneGroup associated with that DDC. For a pull model DDC the GAZ operation is mapped to a Pull Model DDC Request asynchronous event notification (refer to Figure 151). The CDC responds to that asynchronous event notification with a Get Log Page command requesting the Pull Model DDC Request log page (refer to section 5.1.12.3.4), to which the DDC responds with a log page requesting a GAZ operation for a specified ZoneGroup. The GAZ operation is then completed by the CDC by issuing one or more FZS commands to send that ZoneGroup and the operation status to the DDC, as shown in Figure 705.

**Figure 705: GAZ for Pull Model DDC**



The format of the operation specific parameters of a Pull Model DDC Request log page requesting a GAZ operation is shown in Figure 706.

**Figure 706: GAZ Operation Specific Parameters for Pull Model DDC Request Log Page**

Bytes	Description
03:00	<b>Transaction ID (T_ID)</b>
227:04	<b>ZoneGroup Originator (ZGORIG)</b>
257:228	<b>ZoneGroup Name (ZGNAME)</b>

The Transaction ID field is used to relate the Pull Model DDC Request log page for a pull model GAZ to the subsequent FZS command(s). The Zoning Data Key in Command Dword 10 (refer to Figure 512) in the FZS command(s) shall be set to the received Transaction ID value. The ZoneGroup definition is sent through one or more subsequent FZS commands and is provided in the FZS buffer, as shown in Figure 707. The FZS command sending the last fragment shall have the Last Fragment (LF) bit set to '1' in Command Dword 12 (refer to Figure 514).

**Figure 707: FZS Data for Pull Model GAZ**

Bytes	Description
00	<b>Operation Type (OTYP): 7h</b> (i.e., Get Active ZoneGroup for a pull model DDC)

**Figure 707: FZS Data for Pull Model GAZ**

Bytes	Description
03:01	Reserved
07:04	<b>GAZ Operation Status (GAZOS)</b>
11:08	<b>ZoneGroup Fragment Length (ZGFL)</b>
15:12	Reserved
ZGFL+15:16	<b>ZoneGroup Fragment (ZGF)</b>

The GAZ Operation Status field is used to encode status information for the pull model GAZ operation, as shown in Figure 715.

When a ZoneGroup is transferred in multiple fragments, if the CDC detects the ZoneGroup definition has changed while sending the fragments, then the CDC shall issue an FZS command with a zero length ZoneGroup fragment and GAZ operation status set to 5h (i.e., ZoneGroup Changed), and the GAZ operation shall be aborted.

If the ZoneGroup indicated in the Pull Model DDC Request log page for a pull model GAZ operation does not exist on the CDC, then the CDC shall issue an FZS command with a zero length ZoneGroup fragment and GAZ operation status set to 2h (i.e., Zoning Data Structure Not Found), and the GAZ operation shall be aborted.

If the ZoneGroup indicated in the Pull Model DDC Request log page for a pull model GAZ operation is locked on the CDC (i.e., another administrative interface outside the scope of this specification is modifying that ZoneGroup), then the CDC shall issue an FZS command with a zero length ZoneGroup fragment and GAZ operation status set to 3h (i.e., Zoning Data Structure Locked), and the GAZ operation shall be aborted.

If the ZoneGroup indicated in the Pull Model DDC Request log page for a pull model GAZ operation is not locked on the CDC, then the CDC shall continue the GAZ operation by issuing one or more subsequent FZS commands containing the fragments of the requested ZoneGroup. The last fragment shall specify GAZ operation status cleared to 0h (i.e., Operation Successful), and the other fragments shall specify GAZ operation status set to 1h (i.e., Operation in Progress).

The DDC shall not process received ZoneGroup information until the entire ZoneGroup (i.e., all of the fragments of the ZoneGroup) is received.

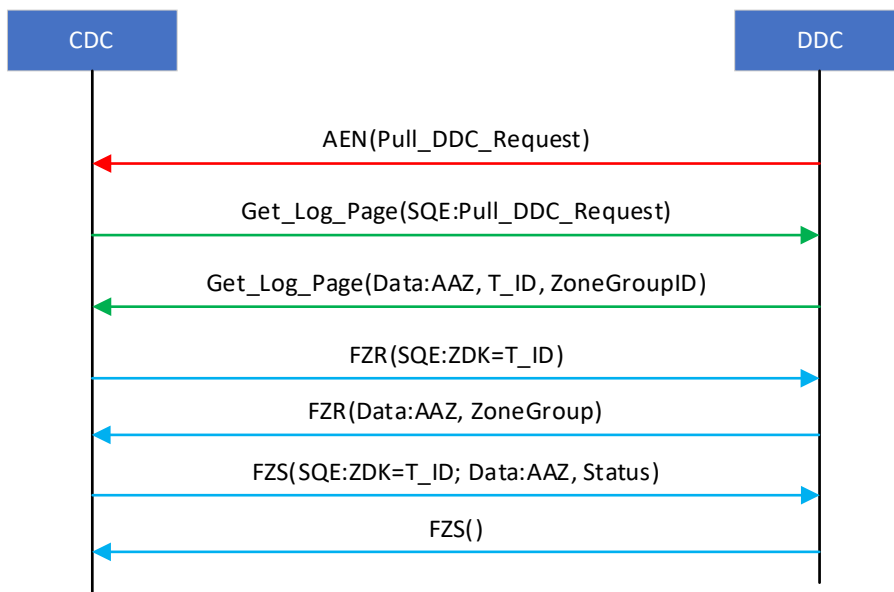
The CDC may enforce access restrictions to the Zoning data structures. In this case, the CDC shall check if the DDC requesting the GAZ operation is authorized to read the ZoneGroup indicated in Pull Model DDC Request log page for Pull Model GAZ operation (e.g., if the CDC allows access to a ZoneGroup only to the DDC that created that ZoneGroup, verify that the ZoneGroup Originator field matches the NQN contained in the SUBNQN field of the Connect command sent from the CDC to that DDC). If that DDC is not authorized to access the specified ZoneGroup, then the CDC shall issue an FZS command with a zero length ZoneGroup fragment and GAZ operation status set to 4h (i.e., ZoneGroup Originator Invalid), and the GAZ operation shall be aborted.

#### **8.3.2.3.8.3.2 Add/Replace Active ZoneGroup (AAZ)**

The Add/Replace Active ZoneGroup (AAZ) operation allows a DDC to add or replace in the CDC an active ZoneGroup associated with that DDC. For a pull model DDC the AAZ operation is mapped to a Pull Model DDC Request asynchronous event notification (refer to Figure 151). The CDC responds to that asynchronous event notification with a Get Log Page command requesting the Pull Model DDC Request log page (refer to section 5.1.12.3.4), to which the DDC responds with a log page requesting an AAZ operation for a specified ZoneGroup. The AAZ operation is then completed by the CDC by issuing one or more FZR commands to retrieve that ZoneGroup from the DDC, and one FZS command to provide an operation status to the DDC, as shown in Figure 708.



**Figure 708: AAZ for Pull Model DDC**



The format of the operation specific parameters of a Pull Model DDC Request log page requesting an AAZ is shown in Figure 709.

**Figure 709: AAZ Operation Specific Parameters for Pull Model DDC Request Log Page**

Bytes	Description
03:00	<b>Transaction ID (T_ID)</b>
227:04	<b>ZoneGroup Originator (ZGORIG)</b>
257:228	<b>ZoneGroup Name (ZGNAME)</b>

The Transaction ID field is used to relate the Pull Model DDC Request log page for a pull model AAZ operation to the subsequent FZR and FZS command(s). The Zoning Data Key in Command Dword 10 (refer to Figure 506) of the FZR command(s) shall be set to the received Transaction ID value. The Zoning Data Key in Command Dword 10 (refer to Figure 512) of the FZS command shall be set to the received Transaction ID value.

If the specified ZoneGroup is locked on the CDC, then the AAZ operation shall be aborted by the CDC by issuing one FZS command with AAZ operation status set to 3h (i.e., Zoning Data Structure Locked) to the DDC. The Transaction ID value returned by the DDC in the Pull Model DDC Request log page for a pull model AAZ operation shall be the same in the FZS command for this AAZ operation.

If the specified ZoneGroup:

- does not exist in ZoneDBActive in that CDC; or
- exists in ZoneDBActive in that CDC and is not locked by another administrative interface,

then the CDC shall lock the specified ZoneGroup in ZoneDBActive and complete the AAZ operation by issuing one or more FZR commands to request from the DDC the ZoneGroup definition to add/replace, followed by an FZS command to report status information to the DDC. The ZoneGroup definition is sent through one or more FZR commands and is provided in the FZR buffer, as shown in Figure 710. The FZR completion queue entry sending the last fragment shall have the Last Fragment (LF) bit set to '1' in Completion Queue Entry Dword 0 (refer to Figure 182).

**Figure 710: FZR Data for Pull Model AAZ**

Bytes	Description
00	<b>Operation Type (OTYP):</b> 9h (i.e., Add/Replace Active ZoneGroup for a pull model DDC)
07:01	Reserved
11:08	<b>ZoneGroup Fragment Length (ZGFL)</b>
15:12	Reserved
ZGFL+15:16	<b>ZoneGroup Fragment (ZGF)</b>

The CDC shall not process received ZoneGroup information until the entire ZoneGroup (i.e., all of the fragments of the ZoneGroup) is received. Upon receiving the ZoneGroup information, the CDC shall update that ZoneGroup in ZoneDBActive in that CDC, increment the ZoneGroup generation number, and issue an FZS command with AAZ operation status cleared to 0h (i.e., Operation Successful) to the DDC.

The AAZ Operation Status is sent through the FZS command and is provided in the FZS buffer, as shown in Figure 711.

**Figure 711: FZS Data for Pull Model AAZ**

Bytes	Description
00	<b>Operation Type (OTYP):</b> Ah (i.e., AAZ status for a pull model DDC)
03:01	Reserved
07:04	<b>AAZ Operation Status (ZAAOS)</b>

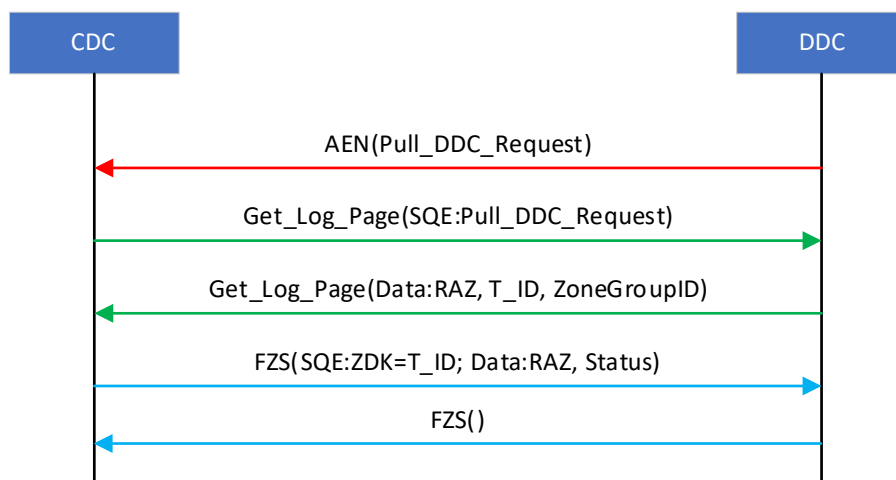
The AAZ Operation Status field is used to encode status information for the pull model AAZ operation, as shown in Figure 715.

The CDC may enforce access restrictions to the Zoning data structures. In this case, the CDC shall check if the DDC requesting the AAZ operation is authorized to write the ZoneGroup indicated in Pull Model DDC Request log page for Pull Model AAZ operation (e.g., if the CDC allows access to a ZoneGroup only to the DDC that created that ZoneGroup, verify that the ZoneGroup Originator field matches the NQN contained in the SUBNQN field of the Connect command sent from the CDC to that DDC). If that DDC is not authorized to access the specified ZoneGroup, then the CDC shall issue an FZS command with AAZ operation status set to 4h (i.e., ZoneGroup Originator Invalid), and the AAZ operation shall be aborted.

### 8.3.2.3.8.3.3 Remove Active ZoneGroup (RAZ)

The Remove Active ZoneGroup (RAZ) operation allows a DDC to remove from the CDC an active ZoneGroup associated with that DDC. For a pull model DDC the RAZ operation is mapped to a Pull Model DDC Request asynchronous event notification (refer to Figure 151). The CDC responds to that asynchronous event notification with a Get Log Page command requesting the Pull Model DDC Request log page (refer to section 5.1.12.3.4), to which the DDC responds with a log page requesting an RAZ operation for a specified ZoneGroup. The RAZ operation is then completed by the CDC by issuing one FZS command to provide an operation status to the DDC, as shown in Figure 712.

**Figure 712: RAZ for Pull Model DDC**



The format of the operation specific parameters of a Pull Model DDC Request log page requesting an RAZ operation is shown in Figure 713.

**Figure 713: RAZ Operation Specific Parameters for Pull Model DDC Request Log Page**

Bytes	Description
03:00	<b>Transaction ID (T_ID)</b>
227:04	<b>ZoneGroup Originator (ZGORIG)</b>
257:228	<b>ZoneGroup Name (ZGNAME)</b>

The RAZ operation is then completed by the CDC by issuing one FZS command to report status information to the DDC. The Transaction ID field is used to relate the Pull Model DDC Request log page for a pull model RAZ operation to the subsequent FZS command. The Zoning Data Key in Command Dword 10 (refer to Figure 512) of the FZS command shall be set to the received Transaction ID value.

The RAZ Operation Status is sent through the FZS command and is provided in the FZS buffer, as shown in Figure 714.

**Figure 714: FZS Data for Pull Model RAZ**

Bytes	Description
00	<b>Operation Type (OTYP):</b> Ch (i.e., RAZ status for a pull model DDC)
03:01	Reserved
07:04	<b>RAZ Operation Status (RAZOS)</b>

The RAZ Operation Status field is used to encode status information for the pull model RAZ operation, as shown in Figure 715.

If the ZoneGroup indicated in the Pull Model DDC Request log page for a pull model RAZ operation does not exist on the CDC, then the CDC shall issue an FZS command with RAZ operation status set to 2h (i.e., Zoning Data Structure Not Found), and the RAZ operation shall be aborted.

If the ZoneGroup indicated in the Pull Model DDC Request log page for a pull model RAZ operation is locked on the CDC (i.e., another administrative interface is modifying that ZoneGroup), then the CDC shall issue an FZS command with RAZ operation status set to 3h (i.e., Zoning Data Structure Locked), and the RAZ operation shall be aborted.

If the ZoneGroup indicated in the Pull Model DDC Request log page for a pull model RAZ operation is not locked on the CDC, then the CDC shall continue the GAZ operation by removing the requested ZoneGroup.

from ZoneDBActive in that CDC and by issuing one subsequent FZS command with RAZ operation status cleared to 0h (i.e., Operation Successful).

The CDC may enforce access restrictions to the Zoning data structures. In this case, the CDC shall check if the DDC requesting the RAZ operation is authorized to remove the ZoneGroup indicated in the Pull Model DDC Request log page for Pull Model RAZ operation (e.g., if the CDC allows access to a ZoneGroup only to the DDC that created that ZoneGroup, verify that the ZoneGroup Originator field matches the NQN contained in the SUBNQN field of the Connect command sent from the CDC to that DDC). If that DDC is not authorized to access the specified ZoneGroup, then the CDC shall issue an FZS command with RAZ operation status set to 4h (i.e., ZoneGroup Originator Invalid), and the RAZ operation shall be aborted.

**8.3.2.3.8.3.4 Pull Model DDC Zoning Operations Status Values**

The Pull Model DDC Zoning operations status values are listed in Figure 715.

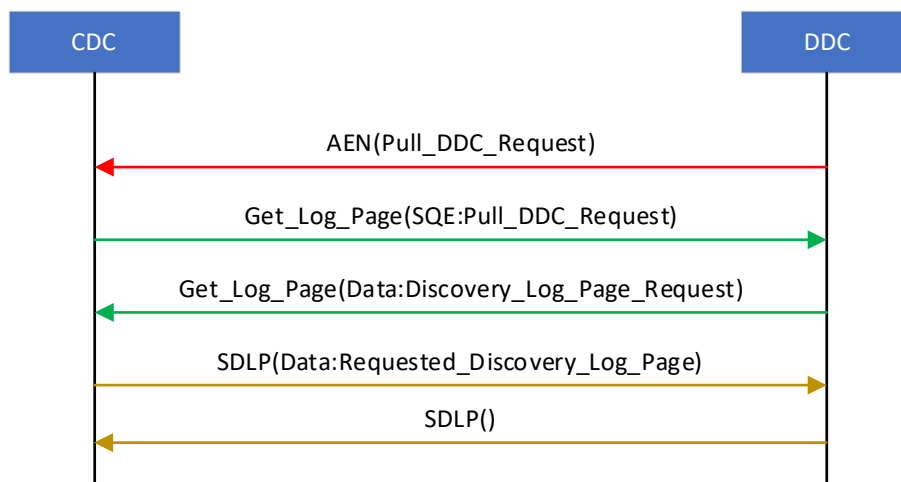
**Figure 715: Pull Model DDC Zoning Operations Status Values**

Value	Definition
0h	Operation Successful
1h	Operation in Progress
2h	Zoning Data Structure Not Found
3h	Zoning Data Structure Locked
4h	ZoneGroup Originator Invalid
5h	ZoneGroup Changed
All others	Reserved

**8.3.2.3.8.3.5 Pull Model DDC Discovery Log Page Request**

A Pull Model DDC retrieves a discovery log page from the CDC through a Pull Model DDC Request asynchronous event notification (refer to Figure 151). The CDC responds to that asynchronous event notification with a Get Log Page command requesting the Pull Model DDC Request log page (refer to section 5.1.12.3.4), to which the DDC responds with a log page requesting a Discovery Log Page Request operation. The Discovery Log Page Request operation is then completed by the CDC by issuing a SDLP command (refer to section 5.1.20) to provide the requested discovery log page to the DDC, as shown in Figure 716.

**Figure 716: Log Page Request Operation**



The format of the operation specific parameters of a Pull Model DDC Request log page requesting a Log Page Request operation is shown in Figure 717.

**Figure 717: Log Page Request Operation Specific Parameters for Pull Model DDC Request Log Page**

Bytes	Description
00	<b>Log Page Request Log Page Identifier (LPRLID)</b>
01	<b>Log Page Request Log Specific Parameter (LPRLSP)</b>
03:02	Reserved

The LPRLID field and the LPRLSP field have respectively the same format and semantics of the LID field and the LSP field in Command Dword 10 of a Get Log Page command. For example, to retrieve the Host Discovery log page, the LPRLID field is set to 71h.

#### 8.3.2.4 Asynchronous Events

A Centralized Discovery controller (CDC) reports a Discovery Log Page Change Asynchronous Event notification (Asynchronous Event Information field set to F0h) to each host that has requested asynchronous event notifications of this type (refer to Figure 151) as specified in section 3.1.3.3.2 when a Fabric Zoning configuration changes. In particular:

- if a Zone member with the Role set to 2h (i.e., an NVM subsystem) is added or removed from a Zone, then the CDC shall report an AEN to all Zone members of that Zone having the Role set to 1h (i.e., all hosts in that Zone);
- if a Zone member with the Role set to 1h (i.e., a host) is added or removed from a Zone, then the CDC shall report an AEN to all Zone members of that Zone having the Role set to 2h (i.e., all NVM subsystems in that Zone); and
- If a Zone is added or removed, then the CDC shall report an AEN to all Zone members of the added or removed Zone.

#### 8.3.3 Exporting NVM Resources

Exported Resource Management is a capability that may be supported in an NVM subsystem that only contains controllers that use transports other than the Memory-Based Transport Model (e.g., the PCIe transport).

Exporting an Underlying Namespace may be achieved by:

- creating an Exported NVM Subsystem (refer to section 5.3.2);
- assigning an Underlying Namespace to an Exported NVM Subsystem (refer to section 5.3.7.1.1);
- attaching a transport to the Exported NVM Subsystem to enable remote access to the Exported NVM Subsystem (refer to section 5.3.9.1.1); and
- optionally assigning access control policies for the Exported NVM Subsystem (refer to sections 5.3.8.1.3 and 5.3.8.1.4).

Exported NVM Subsystems shall not expose information about Underlying NVM Subsystem resources or entities that are not associated with Exported NVM Subsystem entities. This includes entities in Underlying NVM Subsystems (e.g., NSID, NVM Set Identifiers, ANA Group Identifiers).

##### 8.3.3.1 Management of Exported NVM Resources (Informative)

Commands for managing Exported NVM Resources are processed by a controller in an Underlying NVM Subsystem.

This section describes example flows to manage Exported NVM Resources through the use of the:

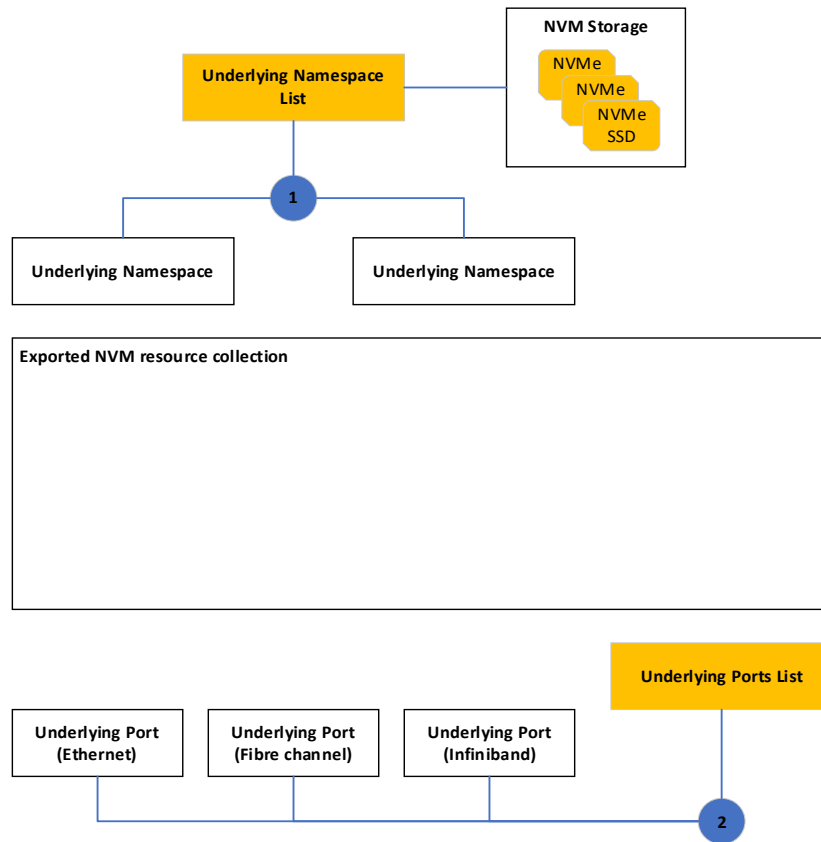
- Identify command with CNS set to 1Dh to retrieve the Underlying Namespace List;
- Identify command with CNS set to 1Eh to retrieve the list of Underlying Ports that may be used to export NVMe over Fabrics Subsystems;
- Create Exported NVM Subsystem command to create an Exported NVM Subsystem;
- Manage Exported NVM Subsystem command to manage access to an Exported NVM Subsystem;

- Manage Exported Namespace command to configure an Exported NVM Subsystem with namespaces;
- Manage Exported Port command to manage fabric port configurations on an Exported NVM Subsystem;
- Manage Exported NVM Subsystem command to delete Exported NVM Subsystems; and
- Clear Exported NVM Resource Configuration command to remove all configured Exported NVM Resources.

### 8.3.3.1.1 Configuring an Exported NVM Subsystem

Prior to configuring exported NVM resources and exposing them for access, an administrative entity (e.g., administrator, resource manager, orchestrator, administration console, centralized configuration manager) must first determine what Underlying resources (e.g., namespaces, ports) are available; this may be through a-priori knowledge or by using a Identify command with CNS set to 1Dh (refer to section 5.1.13.4.1 (refer to step '1' in Figure 718, Figure 719, and in Figure 720) and Identify command with CNS set to 1Eh (refer to section 5.1.13.4.2) (refer to step '2' in Figure 718, Figure 719, and Figure 720). Figure 718 shows collections of Underlying Namespaces and Underlying Ports and an empty collection of Exported NVM resources.

**Figure 718: Example reference for retrieving Underlying Namespaces and Underlying Ports**

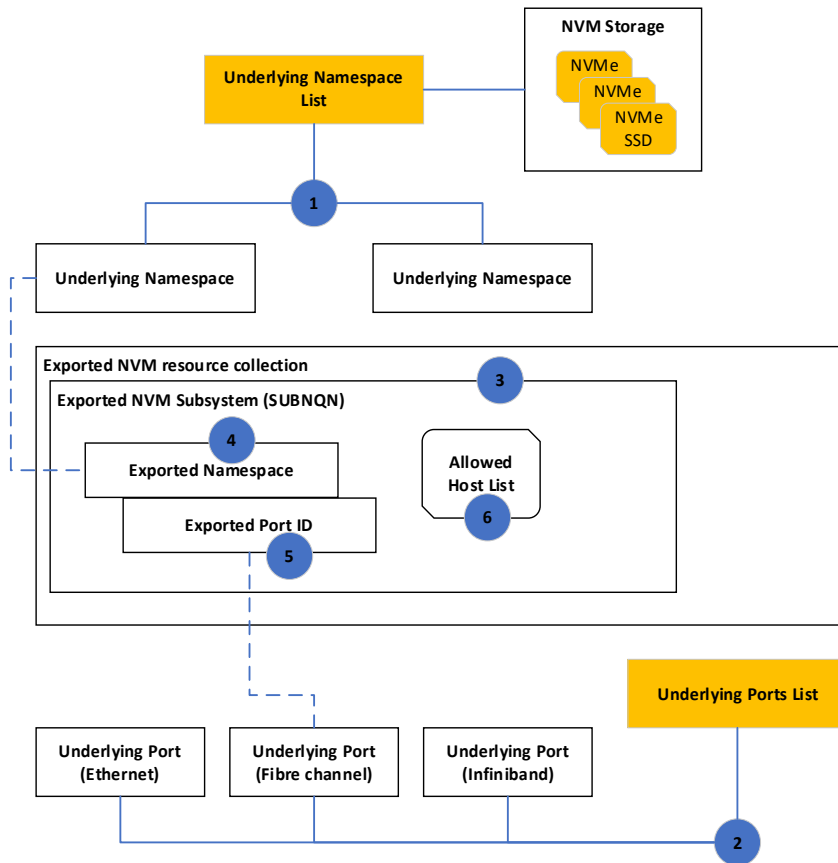


Once the administrative entity has determined the available underlying resources (refer to step '1' and step '2' in Figure 718, Figure 719, and Figure 720), the administrative entity may create, configure, and expose Exported NVM Subsystems. An example flow to configure an Exported NVM Subsystem:

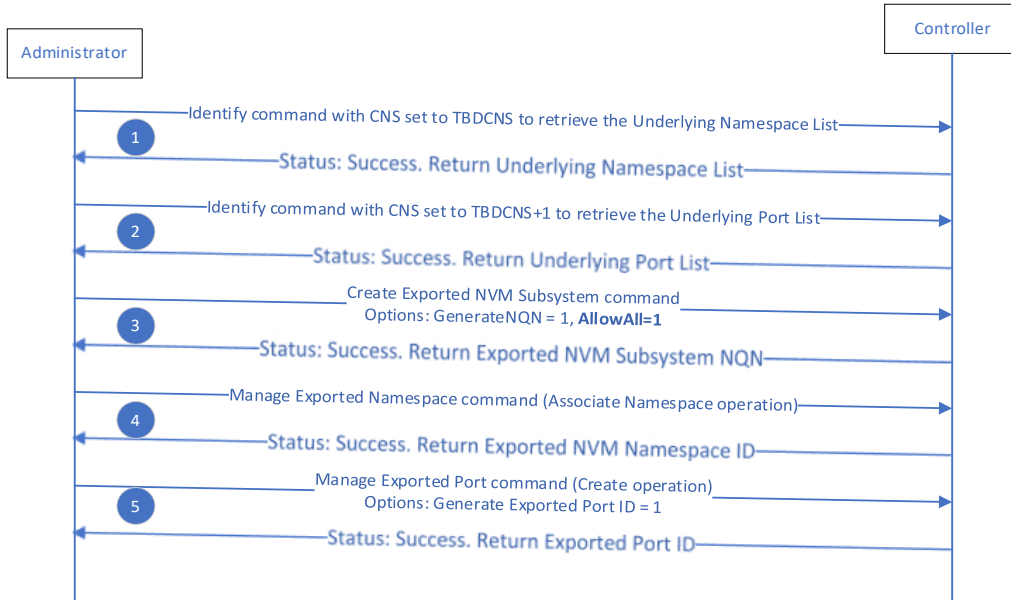
- Create an Exported NVM Subsystem by Issuing a Create Exported NVM Subsystem command (refer to section 5.3.2) (refer to step '3' in Figure 719 and Figure 720).

- Associate individual Underlying Namespaces with an Exported NVM Subsystem by issuing a Manage Exported Namespace command with an Associate Namespace operation (refer to section 5.3.7.1.1) (refer to step '4' in Figure 719 and Figure 720).
- Associate individual Underlying Ports with an Exported NVM Subsystem by issuing a Manage Exported Port command with a Create operation (refer to section 5.3.9.1.1) (refer to step '5' in Figure 719 and Figure 720).

**Figure 719: Example reference for retrieving Underlying Namespaces and Underlying Ports**



**Figure 720: Example steps for retrieving underlying resources and configuring an Exported NVM Subsystem**



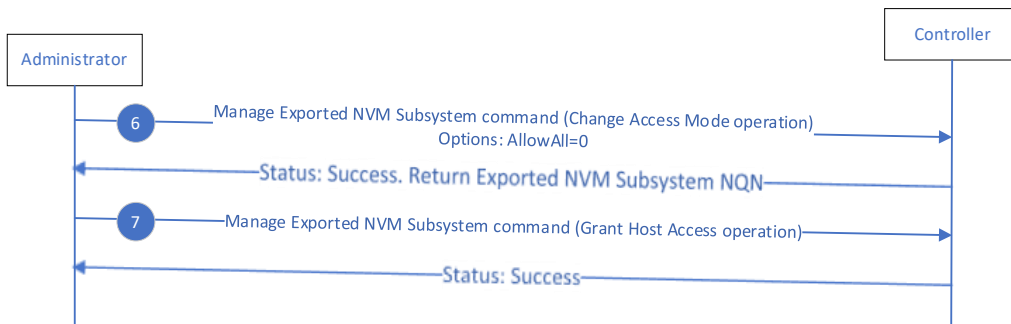
**8.3.3.1.2 Managing Host access to an Exported NVM Subsystem**

Exported NVM Subsystems are created with a specified Access Mode (i.e., restricted, unrestricted).

This section describes an example flow for changing an Exported NVM Subsystem configured for unrestricted access to restricted access (where only hosts in the Allowed Host List associated to the specified Exported NVM Subsystem have permissions to use the Exported Namespaces linked to that Exported NVM Subsystem).

The access mode of Exported NVM Subsystems may be changed by issuing a Manage Exported NVM Subsystem command with a Change Access Mode operation (refer to section 5.3.8.1.2) and specifying the desired Access Mode (refer to step ‘6’ in Figure 721). The example Figure 721 shows changing the access mode of an Exported NVM Subsystem to restricted access. The example Figure 721 also shows Exported NVM Subsystem access to specific hosts (refer to step ‘7’ in Figure 721).

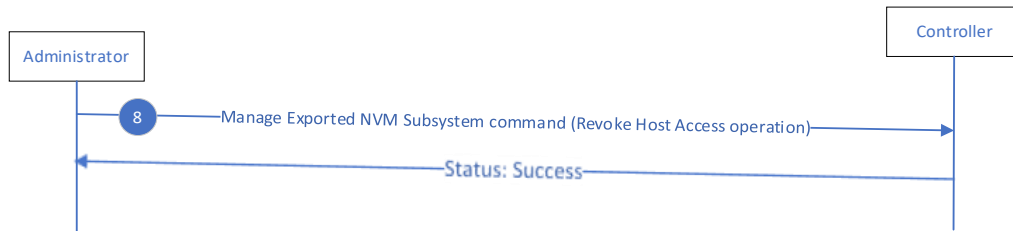
**Figure 721: Example steps for restricting access to an Exported NVM Subsystem to specific hosts**



Similarly, hosts may be restricted from access to specified Exported NVM Subsystems configured for restricted access, on specified ports, by issuing Manage Exported NVM Subsystem commands with the Revoke Host Access operation (refer to section 5.3.8.1.4). Refer to step ‘8’ in Figure 722).



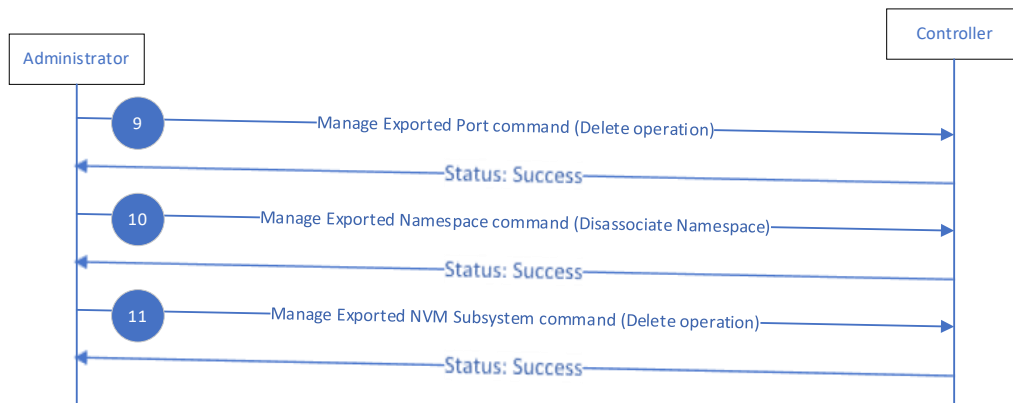
**Figure 722: Example steps for revoking specific hosts access to an Exported NVM Subsystem**



An example flow for removing an Exported NVM Subsystem and associated Namespaces would be issuing:

1. a Manage Exported NVM Port command with a Delete operation (refer to section 5.3.9.1.2) for each port associated with the Exported NVM Subsystem to be removed (refer to step '9' of Figure 723).
2. a Manage Exported Namespace command with a Disassociate Namespace operation (refer to section 5.3.7.1.2) for each Exported Namespaces associated with the Exported NVM Subsystem to be removed (refer to step '10' of Figure 723).
3. a Manage Exported NVM Subsystem command with a Delete operation (refer to section 5.3.8.1.1) of a Manage Exported NVM Subsystem command (refer to step '11' of Figure 723).

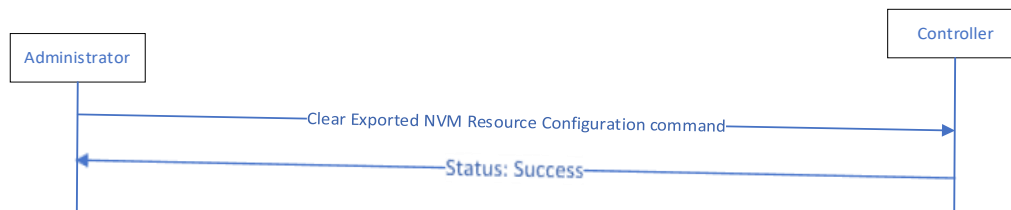
**Figure 723: Example steps for removing an Exported NVM Subsystem**



### 8.3.3.1.3 Clearing all Exported NVM Subsystems

Issuing a Clear Exported NVM Resource Configuration command (refer to section 5.3.1) clears all Exported NVM Subsystems (refer to Figure 724).

**Figure 724: Example steps for clearing all Exported NVM Subsystems**



Upon successful completion of a Clear Exported NVM Resource Configuration command there are no Exported NVM Subsystems.

### 8.3.4 NVMe over Fabrics Secure Channel and In-band Authentication

NVMe over Fabrics supports both fabric secure channel (that includes authentication) and NVMe in-band authentication. Fabric authentication is part of establishing a fabric secure channel via an NVMe Transport

specific protocol that provides authentication, encryption, and integrity checking (e.g., IPsec; refer to RFC 4301 or TLS; refer to RFC 8446). NVMe in-band authentication is performed immediately after a Connect command (refer to section 6.3) succeeds using the Authentication Send and Authentication Receive commands (refer to section 6.1 and section 6.2) to tunnel authentication protocol commands between the host and the controller.

Enrollment of the host and controller in an authentication mechanism, including provisioning of authentication credentials to the host and controller, is outside the scope of this specification.

If both fabric secure channel and NVMe in-band authentication are used, the identities for these two instances of authentication may differ for the same NVMe Transport connection. For example, if an iWARP NVMe Transport is used with IPsec as the fabric secure channel technology, the IPsec identities for authentication are associated with the IP network (e.g., DNS host name or IP address), whereas NVMe in-band authentication uses NVMe identities (i.e., Host NQNs). The NVMe Transport binding specification may provide further guidance and requirements on the relationship between these two identities, but determination of which NVMe Transport identities are authorized to be used with which NVMe identities is part of the security policy for the deployed NVM subsystem.

### 8.3.4.1 Fabric Secure Channel

The Transport Requirements field in the Fabrics Discovery Log Page Entry (refer to Figure 294) indicates whether a fabric secure channel shall be used for an NVMe Transport connection to an NVM subsystem. The secure channel mechanism is specific to the type of fabric.

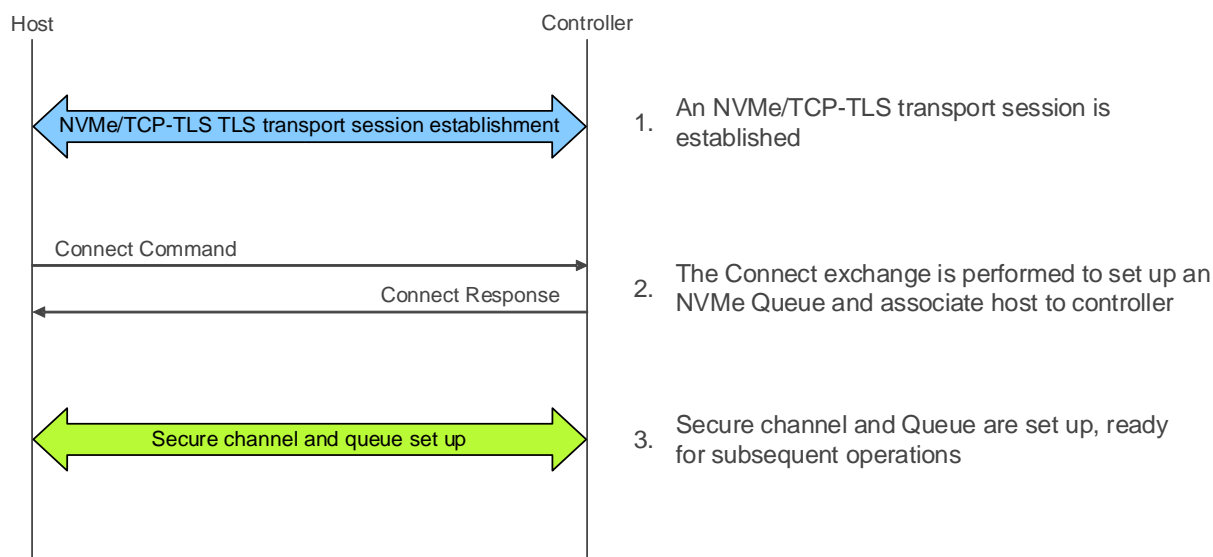
If establishment of a secure channel fails or a secure channel is not established when required by the controller, the resulting errors are fabric-specific and may not be reported to the NVMe layer on the host. Such errors may result in the controller being inaccessible to the host via the NVMe Transport connection on which the failure to establish a fabric secure channel occurred.

An NVM subsystem that requires use of a fabric secure channel (i.e., as indicated by the TSC field in the associated Discovery Log Page Entry) shall not allow capsules to be transferred until a secure channel has been established for the NVMe Transport connection.

All Discovery Log Page Entries for an NVM subsystem should report the same value of TREQ to each host. Discovery Log Page Entries for an NVM subsystem may report different values of TREQ to different hosts.

Figure 725 shows an example of secure channel establishment using TLS, the fabric secure channel protocol for NVMe/TCP.

**Figure 725: Example of TLS secure channel establishment**



### 8.3.4.2 NVMe In-band Authentication

The Authentication and Security Requirements (AUTHREQ) field in the Connect response capsule (refer to Figure 547) indicates whether NVMe in-band authentication is required.

If one or more of the bits in the AUTHREQ field are set to '1', then the controller requires that the host authenticate on that queue in order to proceed with Fabrics, Admin, and I/O commands. Authentication success is defined by the specific security protocol that is used for authentication. If any command other than Connect, Authentication Send, or Authentication Receive is received prior to authentication success, then the controller shall abort the command with Authentication Required status.

If all bits in the AUTHREQ field are cleared to '0', then the controller does not require the host to authenticate, and the NVM subsystem shall not abort any command with a status code value of Authentication Required.

Refer to section 8.3.4.2.1 for considerations on Discovery subsystems.

#### 8.3.4.2.1 Special considerations for In-band Authentication of Discovery subsystems

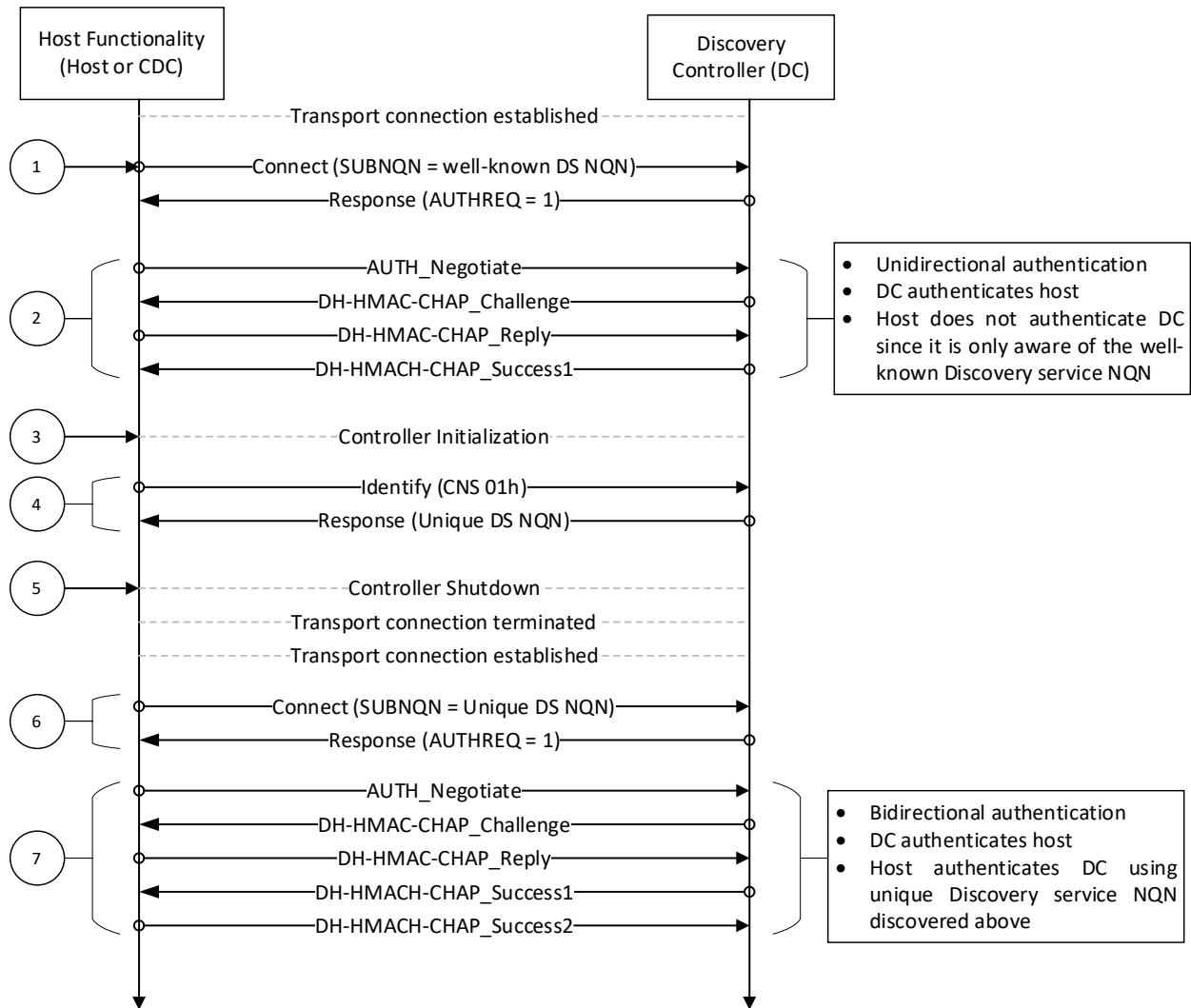
Hosts that have been configured to authenticate Discovery subsystems with an in-band authentication protocol that supports both unidirectional authentication and bidirectional authentication (e.g., DH-HMAC-CHAP, refer to section 8.3.4.5) should behave as follows:

- If the host connected to a Discovery subsystem using the well-known Discovery Service NQN (i.e., nqn.2014-08.org.nvmexpress.discovery) and the Discovery subsystem did not request authentication, then the host should not perform an authentication transaction;
- If the host connected to a Discovery subsystem using the well-known Discovery Service NQN (i.e., nqn.2014-08.org.nvmexpress.discovery) and the Discovery subsystem requested authentication, then the host should perform only unidirectional authentication (i.e., the Discovery subsystem may authenticate the host, but the host should not authenticate the well-known Discovery Service NQN); or
- If the host connected to a Discovery subsystem using the unique Discovery Service NQN for that Discovery subsystem (refer to section 3.1.3.3), regardless of whether the Discovery subsystem requested authentication, then the host may perform unidirectional authentication or bidirectional authentication (i.e., the host may authenticate the unique Discovery Service NQN for that Discovery subsystem).

Figure 726 illustrates a process that a host is able to use to retrieve the unique Discovery Service NQN and perform in-band authentication for a Discovery subsystem if that host has been configured to authenticate Discovery subsystems and the Discovery subsystem that the host connects to also requires authentication (i.e., the AUTHREQ field is not cleared to zero) using DH-HMAC-CHAP for authentication. This process includes:

1. connect to the Discovery subsystem using the well-known Discovery Service NQN (i.e., nqn.2014-08.org.nvmexpress.discovery);
2. perform unidirectional authentication with the Discovery subsystem;
3. perform controller initialization (refer to section 3.5);
4. retrieve the unique Discovery Service NQN (refer to section 3.1.3.3);
5. perform controller shutdown (refer to section 3.6);
6. reconnect to the Discovery subsystem using the unique Discovery Service NQN for that Discovery subsystem; and
7. perform bidirectional authentication with the Discovery subsystem using the unique Discovery Service NQN for that Discovery subsystem.

**Figure 726: Unique Discovery Service NQN retrieval for bidirectional authentication**



The host may initiate a subsequent authentication transaction at any time for reauthentication purposes. Initiating reauthentication shall not invalidate a prior authentication. If the reauthentication transaction concludes with the controller sending an `AUTH_Failure1` message (refer to section 8.3.4.4.2), then the controller shall terminate all commands with a status code of Operation Denied and disconnect the NVMe over Fabrics connection. If the reauthentication transaction concludes with the host sending an `AUTH_Failure2` message, then the host shall disconnect the NVMe over Fabrics connection.

The state of an in-progress authentication transaction is soft-state. If the subsequent command in an authentication transaction is not received by the controller within a timeout equal to:

- the Keep Alive Timeout value (refer to Figure 545), if the Keep Alive Timer is enabled; or
- the default Keep Alive Timeout value (i.e., two minutes), if the Keep Alive Timer is disabled;

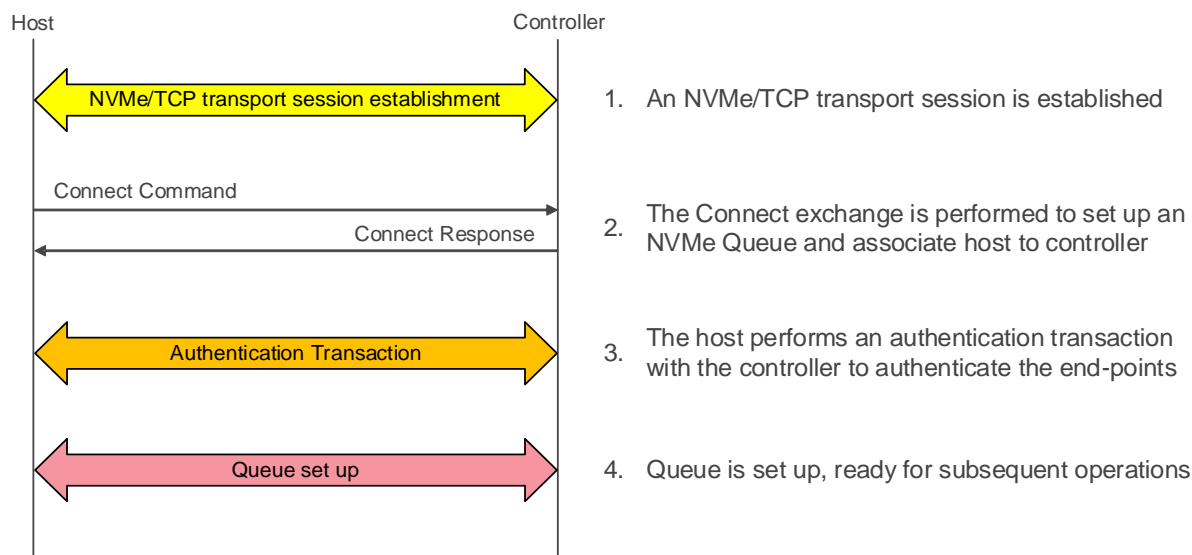
then the authentication transaction has timed out and the controller should discard the authentication transaction state (including the `T_ID` value, refer to section 8.3.4.4.1).

For an initial authentication, an authentication transaction timeout should be treated as an authentication failure with termination of the transport connection. For reauthentication, an authentication transaction timeout should not be treated as an authentication failure. Authentication commands used to continue that

transaction after an authentication transaction timeout should be aborted with a status code of Command Sequence Error.

Figure 727 shows an example of authentication transaction for NVMe/TCP.

**Figure 727: Example of authentication transaction for NVMe/TCP**



### 8.3.4.2.2 NVMe In-band Authentication Protocol-Specific Requirements

Authentication requirements for security commands are based on the security protocol indicated by the SECP field in the command.

The authentication protocols defined by this specification use the security protocol identifier E9h (assigned to NVMe by SPC-5, a SCSI standard). The messages of the defined authentication protocols are self-identifying, therefore the SPSP0 field and the SPSP1 field of the Authentication Send and Authentication Receive commands shall be set to 01h. Authentication messages are mapped to NVMe over Fabrics command and response pairs. The mapping of authentication messages to the Authentication Send command is shown in Figure 728.

**Figure 728: Mapping of authentication messages to the Authentication Send command**

Field <sup>1</sup>	Value
<b>SPSP0</b>	01h
<b>SPSP1</b>	01h
<b>SECP</b>	E9h
<b>TL</b>	Specifies the amount of data to transfer in bytes

Notes:

1. Refer to section 6.2.

The mapping of authentication messages to the Authentication Receive command is shown in Figure 729. Security processing requirements associated with the Authentication Receive command (e.g., delays in third-party authentication verification) may result in delays in controller completion of an Authentication Receive command. The host should consider these possible delays associated with the Authentication Receive command.

**Figure 729: Mapping of authentication messages to the Authentication Receive command**

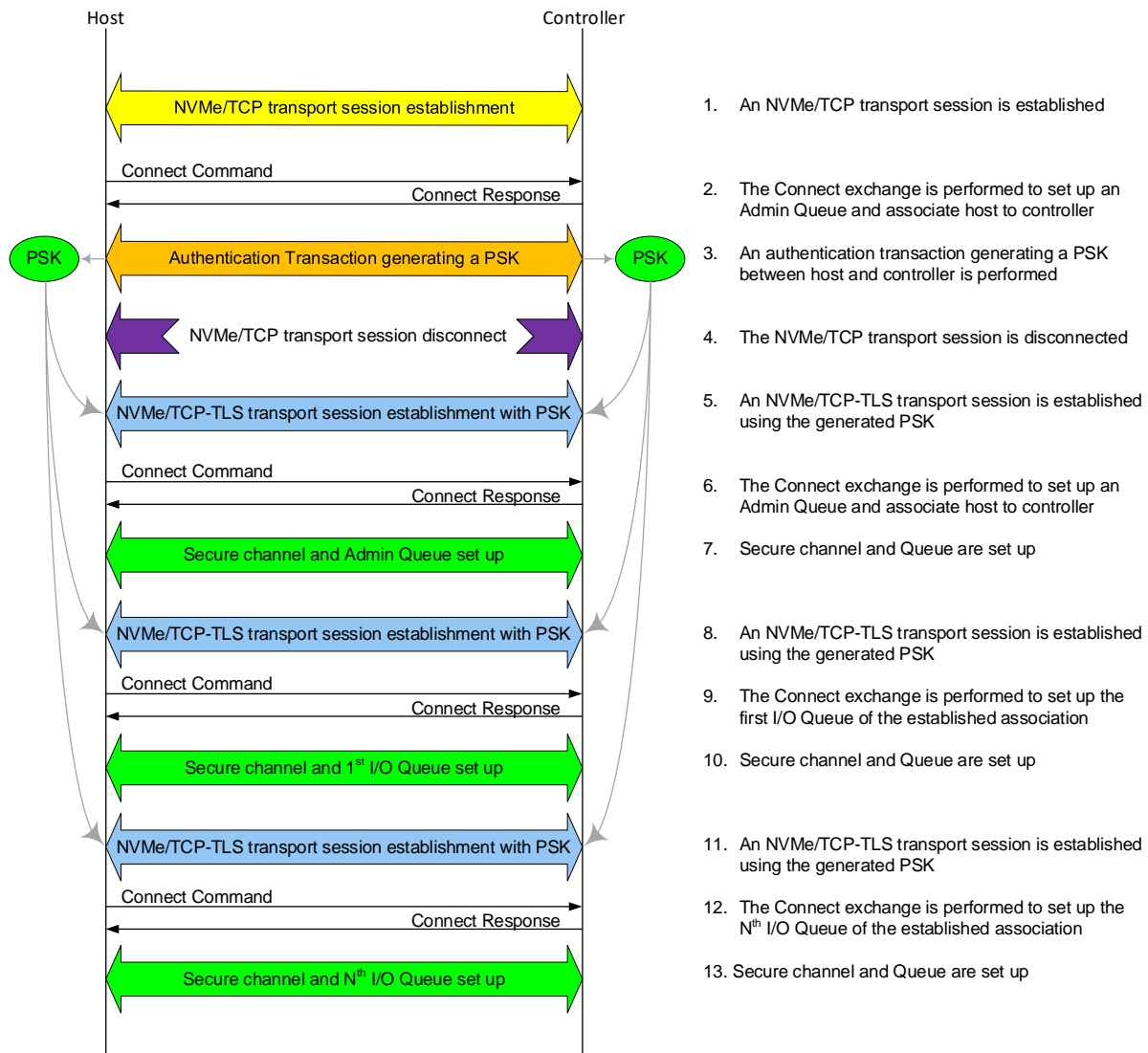
Field <sup>1</sup>	Value
<b>SPSP0</b>	01h
<b>SPSP1</b>	01h
<b>SECP</b>	E9h
<b>AL</b>	Specifies the amount of data to transfer in bytes <sup>2</sup>
Notes:	
1. Refer to section 6.1.	
2. The size of the largest authentication message that could be received.	

#### 8.3.4.3 Secure Channel Concatenation

It is possible to leverage an authentication transaction to generate shared key material to use as pre-shared key (PSK) to establish a secure channel (e.g., with IPsec or TLS). This PSK is generated by an authentication transaction on an Admin Queue over an unsecure channel. Once the authentication transaction is completed, that Admin Queue transport connection shall be disconnected by the host. The generated PSK may then be used to set up secure channels for subsequent Admin Queue(s) and I/O Queues.

The process of generating a PSK on an Admin Queue over an insecure channel, disconnecting that Admin Queue transport connection, and setting up an Admin Queue over a secure channel established using that generated PSK is called secure channel concatenation. Figure 730 shows an example of this process for TLS with NVMe/TCP, where steps 1 to 7 show TLS secure channel concatenation for the Admin Queue and steps 8 to 13 show the setup of TLS secure channels for the I/O Queues.

**Figure 730: Example of TLS secure channel concatenation with NVMe/TCP**



Secure channel concatenation is prohibited over any secure channel that has been established in compliance with the requirements of an NVMe transport specification (e.g., the NVM Express TCP Transport Specification). The controller response to a host that requests secure channel concatenation in this situation is specified in section 8.3.4.4.1. In contrast, replacing a PSK for such a secure channel is permitted (refer to section 8.3.4.4.1).

### 8.3.4.4 Common Authentication Messages

#### 8.3.4.4.1 AUTH\_Negotiate Message

The AUTH\_Negotiate message is sent from the host to the controller and is used to indicate the authentication protocols the host is able to use in this authentication transaction and which secure channel protocol, if any, to concatenate to this authentication transaction. The AUTH\_Negotiate message format is shown in Figure 731.

**Figure 731: AUTH\_Negotiate message format**

Bytes	Description
0	<b>Authentication Type (AUTH_TYPE):</b> 00h (i.e., common messages)
1	<b>Authentication Identifier (AUTH_ID):</b> 00h (i.e., AUTH_Negotiate)
3:2	Reserved
5:4	<b>Transaction Identifier (T_ID):</b> 16-bit transaction identifier
6	<b>Transaction Identifier (SC_C)</b>
7	<b>Number of Authentication Protocol Descriptors (NAPD):</b> This field specifies the number of Authentication Protocol Descriptors are in the Authentication Protocol Descriptor list.
<b>Authentication Protocol Descriptor List</b>	
71:8	<b>Authentication Protocol Descriptor 1</b>
135:72	<b>Authentication Protocol Descriptor 2</b>
...	...
NAPD*64+7:(NAPD-1)*64+8	<b>Authentication Protocol Descriptor NAPD</b>

The SC\_C field determines if a secure channel concatenation to the authentication transaction is requested and with which secure channel protocol, as shown in Figure 732.

**Figure 732: Secure channel protocol identifiers**

Value	Definition	Transport Applicability
00h	<b>NOSC:</b> No secure channel concatenation	n/a
01h	Obsolete (refer to NVM Express Base Specification 2.0)	
02h	<b>NEWTLSPSK:</b> Used on an Admin Queue over a TCP channel without TLS to generate a PSK and associated PSK identity. This {PSK, PSK Identity} pair may be used to set up TLS secure channels for subsequent Admin and I/O queues.	TCP
03h	<b>REPLACETLSPSK:</b> Used on an Admin Queue over a TLS secure channel to generate a PSK and associated PSK identity. This {PSK, PSK Identity} pair replaces the {PSK, PSK Identity} pair that was used to set up the TLS secure channel over which the authentication transaction is performed.	TCP with TLS
All other values	Reserved	

An authentication transaction with the SC\_C field set to NOSC is allowed on any Admin or I/O Queue and does not generate a PSK (i.e., it performs authentication only).

An authentication transaction with the SC\_C field set to NEWTLSPSK:

- is allowed on an Admin Queue over a TCP channel without TLS and generates a new PSK;
- is prohibited on an I/O Queue over a TCP channel without TLS; and
- is prohibited on an Admin Queue or I/O queue over a TLS secure channel.

An authentication transaction with the SC\_C field set to REPLACETLSPSK:

- is allowed on an Admin Queue over a TLS secure channel and replaces the association's PSK;
- is prohibited on an Admin Queue or I/O queue over a TCP channel without TLS; and
- is prohibited on an I/O Queue over a TLS secure channel.

The AUTH\_Negotiate message is structured as a list of 64-byte authentication protocol descriptors to enable extensibility to define additional authentication protocols. Currently only one authentication protocol is defined (i.e., DH-HMAC-CHAP), therefore the AUTH\_Negotiate message carries only one authentication protocol descriptor (i.e., NAPD=1). Implementations should support more than one descriptor to enable protocol extensibility. The first byte of an authentication protocol descriptor identifies the specific authentication protocol, as shown in Figure 733.



**Figure 733: Authentication protocol identifiers**

Value	Definition
01h	DH-HMAC-CHAP (refer to section 8.3.4.5)
All other values	Reserved

Upon receiving an AUTH\_Negotiate message, if the SC\_C value indicated by the host:

- does not satisfy the security requirements of the controller (e.g., the host did not request secure channel concatenation, but the controller’s security configuration requires secure channel concatenation);
- is prohibited for the Admin Queue or I/O Queue via which the AUTH\_Negotiate message was received, as specified in this section; or
- requests secure channel concatenation and that value is contained in an AUTH\_Negotiate message received over a secure channel established in compliance with the requirements of an NVMe transport specification (e.g., the NVM Express TCP Transport Specification),

then the controller shall:

- reply to the AUTH\_Negotiate message with an AUTH\_Failure1 message having reason code ‘Authentication failure’ and reason code explanation ‘Secure channel concatenation mismatch’; and
- disconnect the NVMe over Fabrics connection upon transmitting the AUTH\_Failure1 message.

Upon receiving an AUTH\_Negotiate message, if the protocol descriptors proposed by the host do not satisfy the security requirements of the controller, then the controller shall:

- reply to the AUTH\_Negotiate message with an AUTH\_Failure1 message having reason code ‘Authentication failure’ and reason code explanation ‘Authentication protocol not usable’; and
- disconnect the NVMe over Fabrics connection upon transmitting the AUTH\_Failure1 message.

#### 8.3.4.4.2 AUTH\_Failure Messages

The AUTH\_Failure1 message is sent from the controller to the host, the AUTH\_Failure2 message is sent from the host to the controller. The format of the AUTH\_Failure1 message and of the AUTH\_Failure2 message is shown in Figure 734.

**Figure 734: AUTH\_Failure1 and AUTH\_Failure2 message format**

Bytes	Description
0	<b>Authentication Type (AUTH_TYPE):</b> 00h (i.e., common messages)
1	<b>Authentication Identifier (AUTH_ID):</b> F0h (i.e., AUTH_Failure2) F1h (i.e., AUTH_Failure1)
3:2	Reserved
5:4	<b>Transaction Identifier (T_ID):</b> 16-bit transaction identifier
6	<b>Reason Code (RCODE)</b>
7	<b>Reason Code Explanation (RCODEEX)</b>

The AUTH\_Failure reason codes are listed in Figure 735.

**Figure 735: AUTH\_Failure reason codes**

Value	Definition
01h	<b>Authentication failure:</b> The authentication transaction failed
All other values	Reserved

The AUTH\_Failure reason code explanations are listed in Figure 736.

**Figure 736: AUTH\_Failure reason code explanations**

Value	Definition
01h	<b>Authentication failed:</b> Authentication of the involved host or NVM subsystem failed.
02h	<b>Authentication protocol not usable:</b> The protocol descriptors proposed by the host do not satisfy the security requirements of the controller (refer to section 8.3.4.4.1).
03h	<b>Secure channel concatenation mismatch:</b> The SC_C value indicated by the host does not satisfy the security requirements of the controller (refer to section 8.3.4.4.1).
04h	<b>Hash function not usable:</b> The HashIDList proposed by the host does not satisfy the security requirements of the controller (refer to section 8.3.4.5.2).
05h	<b>DH group not usable:</b> The DHgIDList proposed by the host does not satisfy the security requirements of the controller (refer to section 8.3.4.5.2).
06h	<b>Incorrect payload:</b> The payload of the received message is not correct.
07h	<b>Incorrect protocol message:</b> The received message is not the expected next message in the authentication protocol sequence.
All other values	Reserved

#### 8.3.4.4.3 Mapping of Common Authentication Messages to Authentication Commands

The AUTH\_Negotiate message and the AUTH\_Failure2 message are sent from the host to the controller, therefore they are mapped to the Authentication Send command. The AUTH\_Failure1 message is sent from the controller to the host, therefore it is mapped to the Authentication Receive command.

#### 8.3.4.5 DH-HMAC-CHAP Protocol

##### 8.3.4.5.1 Protocol Operations

DH-HMAC-CHAP is a key based Authentication and key management protocol that uses the Challenge Handshake Authentication Protocol (CHAP, refer to RFC 1994) enhanced to use the Hashed Message Authentication Code (HMAC) mechanism (refer to RFC 2104) with stronger hash functions and augmented with an optional Diffie-Hellman (DH) exchange (refer to RFC 2631, clause 2.2.1). DH-HMAC-CHAP provides bidirectional or unidirectional Authentication between a host and a controller.

The Diffie-Hellman part of the protocol is optional. When the Diffie-Hellman part of the protocol is not used, DH-HMAC-CHAP is referred to as HMAC-CHAP. If insufficiently random keys are used (refer to section 8.3.4.5.7), HMAC-CHAP potentially allows a passive eavesdropper to discover the key through an off-line dictionary attack, so its usage should be minimized. DH-HMAC-CHAP provides strong protection from passive eavesdroppers. However, an active attacker could reduce the operation of this protocol so that only HMAC-CHAP is used, and as a result gain sufficient information to mount an off-line dictionary attack on the HMAC-CHAP key.

An implementation that supports DH-HMAC-CHAP authentication shall support DH-HMAC-CHAP with a NULL DH exchange. All implementations of DH-HMAC-CHAP shall be configurable to require a DH exchange (i.e., to not use HMAC-CHAP).

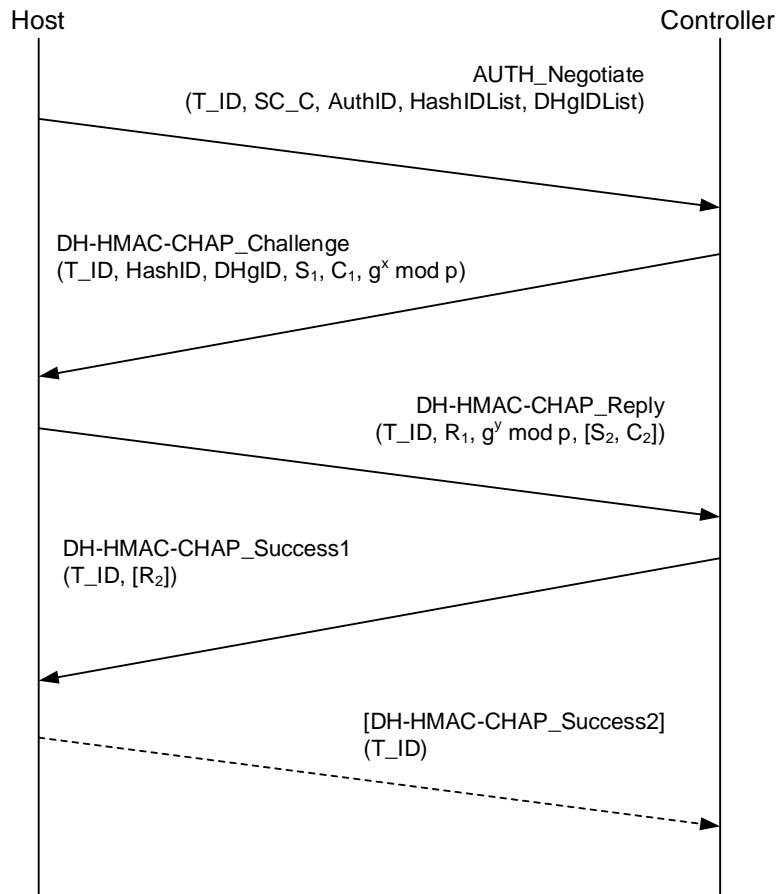
In order to authenticate with the DH-HMAC-CHAP protocol, each host and NVM subsystem shall be provided with a DH-HMAC-CHAP key that is associated with the entity's NQN. Two entities may impersonate one another if they have the same key, therefore when the assigned keys are not different for each entity there is a security vulnerability (refer to section 8.3.4.5.7).

To authenticate another entity, an entity is required to either:

- a) know the key associated with the entity to be authenticated; or
- b) rely on a third party that knows the key to verify the authentication (refer to section 8.3.4.5.11).

An example of a DH-HMAC-CHAP authentication transaction is shown in Figure 737, with the notation shown in Figure 738. The DH-HMAC-CHAP\_Success2 message that is shown as a dashed line is used only for bidirectional authentication.

**Figure 737: Example of DH-HMAC-CHAP authentication transaction**



**Figure 738: Mathematical notations for DH-HMAC-CHAP**

Symbols	Description
$NQN_c, NQN_h$	NQN of the NVM subsystem that contains the controller and NQN of the host
$K_c, K_h$	DH-HMAC-CHAP key of the NVM subsystem that contains the controller and DH-HMAC-CHAP key of the host
$p, g$	Modulus ( $p$ ) and generator ( $g$ ) of the chosen DH group (refer to Figure 741)
$x, y$	Random numbers used as exponents in a DH exchange
$C_1, C_2$	Random challenge values
$Ca_1, Ca_2$	Augmented challenge values
$S_1, S_2$	32-bit sequence numbers
$R_1, R_2$	Reply values
$T\_ID$	Authentication transaction identifier
$SC\_C$	Secure channel concatenation indication
$H()$	One-way hash function (refer to Figure 740)
$HMAC(K, Str)$	HMAC function (refer to RFC 2104) with key $K$ on string $Str$ using hash function $H()$
$  $	Concatenation operation
$K_s$	Session key

When used with a non-NULL DH exchange, the DH-HMAC-CHAP protocol is able to generate a session key  $K_s$  to be used to establish a TLS session between host and controller (refer to section 8.3.4.5.9).

For an NVM subsystem, the controller is the entity running the protocol, using the identity and credentials of the NVM subsystem. The DH-HMAC-CHAP protocol proceeds in the following order:

- 1) The authentication transaction shall begin with the host sending the common AUTH\_Negotiate message to negotiate the authentication protocol to use and its associated parameters (refer to section 8.3.4.4.1). The AUTH\_Negotiate message carries the transaction identifier (T\_ID) for the entire authentication transaction and the list of authentication protocol descriptors for the authentication protocols that may be used in this authentication transaction. For DH-HMAC-CHAP, the authentication protocol descriptor includes the list of hash functions (HashIDList) and Diffie-Hellman group identifiers (DHgIDList) that may be used in this authentication protocol transaction.
- 2) If the parameters of the received DH-HMAC-CHAP protocol descriptor are compatible with the controller's policies, then the controller shall reply with a DH-HMAC-CHAP\_Challenge message (refer to section 8.3.4.5.3) carrying the same transaction identifier value (T\_ID) received in the AUTH\_Negotiate message, the identifiers of the hash function (HashID) and the DH group (DHgID) selected for use among the ones proposed by the host in the AUTH\_Negotiate message, a sequence number (S<sub>1</sub>), a random challenge value (C<sub>1</sub>), and the DH exponential ( $g^x \bmod p$ ). If the controller selects a NULL DH group identifier, then the DH portion of the DH-HMAC-CHAP protocol shall not be used, and the protocol reduces to a HMAC-CHAP transaction.
- 3) If the received DH-HMAC-CHAP\_Challenge message is valid, then the host shall send a DH-HMAC-CHAP\_Reply message (refer to section 8.3.4.5.11) carrying the same transaction identifier value (T\_ID), the response R<sub>1</sub> to the challenge value C<sub>1</sub>, and its own DH exponential ( $g^y \bmod p$ ). The DH Value Length shall be cleared to 0h if the controller has sent a NULL DH group identifier in the DH-HMAC-CHAP\_Challenge message. If bidirectional authentication is requested, then the DH-HMAC-CHAP\_Reply message shall carry also a sequence number S<sub>2</sub> and a random challenge value C<sub>2</sub> that differs from the challenge value C<sub>1</sub> received in the DH-HMAC-CHAP\_Challenge message.
- 4) If the authentication verification by the controller succeeds, then the controller shall reply with a DH-HMAC-CHAP\_Success1 message (refer to section 8.3.4.5.5) carrying the same transaction identifier value (T\_ID). If bidirectional authentication was requested, then the DH-HMAC-CHAP\_Success1 message shall also carry the response R<sub>2</sub> to the challenge value C<sub>2</sub>. If the authentication verification fails, then the controller shall send an AUTH\_Failure1 message and disconnect the NVMe over Fabrics connection upon transmitting it.
- 5) The authentication transaction ends here, unless bidirectional authentication has been requested. In this case, as shown by the dashed arrow in Figure 737, if the authentication verification by the host succeeds, then the host shall send a DH-HMAC-CHAP\_Success2 message (refer to section 8.3.4.5.6) carrying the same transaction identifier value (T\_ID). If the authentication verification fails, then the host shall send an AUTH\_Failure2 message and disconnect the NVMe over Fabrics connection upon transmitting it.

If the controller receives a message that is not the expected next message in the DH-HMAC-CHAP protocol sequence, then the controller shall:

- reply with an AUTH\_Failure1 message having reason code 'Authentication failure' and reason code explanation 'Incorrect protocol message'; and
- disconnect the NVMe over Fabrics connection upon transmitting the AUTH\_Failure1 message.

If the host receives a message that is not the expected next message in the DH-HMAC-CHAP protocol sequence, then the host shall:

- reply with an AUTH\_Failure2 message having reason code 'Authentication failure' and reason code explanation 'Incorrect protocol message'; and
- disconnect the NVMe over Fabrics connection upon transmitting the AUTH\_Failure2 message.

The payload format of a message shall be validated before performing any other security computation.

#### 8.3.4.5.2 DH-HMAC-CHAP Authentication Protocol Descriptor

The authentication protocol descriptor for DH-HMAC-CHAP (refer to section 8.3.4.4.1) is shown in Figure 739.

**Figure 739: Authentication protocol descriptor for DH-HMAC-CHAP**

Bytes	Description
0	<b>Authentication Protocol Identifier (AuthID):</b> (01h for DH-HMAC-CHAP)
1	Reserved
2	<b>HashIDList Length (HALEN):</b> Number of hash function identifiers (1 to 30)
3	<b>DHgidList Length (DHLEN):</b> Number of Diffie-Hellman group identifiers (1 to 30)
3+HALEN:4	<b>Hash Function Identifier List (HashIDList):</b> Array of hash function identifiers (one byte per identifier)
33:4+HALEN	<b>Pad (PAD):</b> Padding bytes cleared to 0h, if present
33+DHLEN:34	<b>Diffie-Hellman Group Identifier List (DHgidList):</b> Array of Diffie-Hellman Group identifiers (one byte per identifier)
63:34+DHLEN	<b>Pad1 (PAD1):</b> Padding bytes cleared to 0h, if present

The one-way hash functions used by DH-HMAC-CHAP are shown in Figure 740.

**Figure 740: DH-HMAC-CHAP hash function identifiers**

Identifier	Hash Function	Hash Length (bytes)	Hash Block Size <sup>1</sup> (bytes)	Reference
00h	Reserved			
01h	SHA-256	32	64	RFC 6234
02h	SHA-384	48	128	RFC 6234
03h	SHA-512	64	128	RFC 6234
04h-DFh	Reserved			
E0h-FEh	Vendor specific			
FFh	Reserved			
Notes:				
1. The hash block size is used by the HMAC calculation				

The SHA-256 hash function shall be supported. Use of the SHA-256 hash function may be prohibited by the requirements of security policies that are not defined by NVM Express (e.g., CNSA 1.0 requires use of SHA-384).

Upon receiving an AUTH\_Negotiate message, if the HashIDList proposed by the host does not satisfy the security requirements of the controller (e.g., the host proposed SHA-256, but the controller's security policy requires a SHA-384 hash), then the controller shall:

- reply to the AUTH\_Negotiate message with an AUTH\_Failure1 message having reason code 'Authentication failure' and reason code explanation 'Hash function not usable'; and
- disconnect the NVMe over Fabrics connection upon transmitting the AUTH\_Failure1 message.

The Diffie-Hellman (DH) groups used by DH-HMAC-CHAP are shown in Figure 741.

**Figure 741: DH-HMAC-CHAP Diffie-Hellman group identifiers**

Identifier	DH group size	Generator (g)	Modulus (p) and Reference
00h	NULL	n/a	n/a
01h	2048-bit	2	refer to RFC 7919
02h	3072-bit	2	refer to RFC 7919
03h	4096-bit	2	refer to RFC 7919
04h	6144-bit	2	refer to RFC 7919
05h	8192-bit	2	refer to RFC 7919
06h-DFh	Reserved		
E0h-FEh	Vendor specific		
FFh	Reserved		

The 00h identifier indicates that no Diffie-Hellman exchange is performed, which reduces the DH-HMAC-CHAP protocol to the HMAC-CHAP protocol. The 00h identifier shall not be proposed in an AUTH\_Negotiate message that requests secure channel concatenation (i.e., with the SC\_C field set to a non-zero value).

The 2048-bit DH group and the 3072-bit DH group shall be supported. A mechanism shall be provided to disable (i.e., prohibit) use of the 2048-bit DH group. Use of the 2048-bit DH group may be prohibited by the requirements of security policies that are not defined by NVM Express (e.g., CNSA 1.0 requires use of a 3072-bit or larger DH group).

Upon receiving an AUTH\_Negotiate message, if the DHgIDList proposed by the host:

- does not satisfy the security requirements of the controller (e.g., the host proposed only the NULL DH group, but the controller's security policy requires a DH group whose size is 3072-bit or larger); or
- contains the NULL DH group (i.e., identifier 00h) and the AUTH\_Negotiate message is requesting secure channel concatenation (i.e., with the SC\_C field set to a non-zero value),

then the controller shall:

- reply to the AUTH\_Negotiate message with an AUTH\_Failure1 message having reason code 'Authentication failure' and reason code explanation 'DH group not usable'; and
- disconnect the NVMe over Fabrics connection upon transmitting the AUTH\_Failure1 message.

#### 8.3.4.5.3 DH-HMAC-CHAP\_Challenge Message

The DH-HMAC-CHAP\_Challenge message is sent from the controller to the host. The format of the DH-HMAC-CHAP\_Challenge message is shown in Figure 742.

**Figure 742: DH-HMAC-CHAP\_Challenge message format**

Bytes	Description
0	<b>Authentication Type (AUTH_TYPE):</b> 01h (i.e., DH-HMAC-CHAP)
1	<b>Authentication Identifier (AUTH_ID):</b> 01h (i.e., DH-HMAC-CHAP_Challenge)
3:2	Reserved
5:4	<b>Transaction Identifier (T_ID):</b> 16-bit transaction identifier
6	<b>Hash Length (HL):</b> Length in bytes of the selected hash function
7	Reserved
8	<b>Hash Identifier (HashID):</b> Identifier of selected hash function
9	<b>Diffie-Hellman Group Identifier (DHgID):</b> Identifier of selected Diffie-Hellman group
11:10	<b>DH Value Length (DHVLEN):</b> Length in bytes of DH value. If no DH value is included in the message, then this field is cleared to 0h
15:12	<b>Sequence Number (SEQNUM):</b> Sequence number S <sub>1</sub>
15+HL:16	<b>Challenge Value (CVAL):</b> Challenge C <sub>1</sub>
15+HL+DHVLEN:16+HL	<b>DH Value (DHV):</b> DH exponential $g^x \text{ mod } p$ . This field is not present (i.e., the CVAL field is the last field in the message) if DHVLEN is cleared to 0h

**Hash Length (HL):** Shall be set to the length in bytes of the selected hash function, as specified in Figure 740.

**HashID:** Shall be set to the hash function identifier (refer to Figure 740) selected for this authentication transaction among those proposed in the DH-HMAC-CHAP protocol descriptor in the AUTH\_Negotiate message. The controller shall select a hash function in accord with its applicable policy.

**DHgID:** Shall be set to the DH group identifier (refer to Figure 742) selected for this authentication transaction among those proposed in the DH-HMAC-CHAP protocol descriptor in the AUTH\_Negotiate message. The controller shall select a DH group identifier in accord with its applicable policy. If this field is cleared to 0h, then the DH portion of the DH-HMAC-CHAP protocol shall not be performed in this authentication transaction. The controller shall not clear this field to 00h if the AUTH\_Negotiate message

(refer to section 8.3.4.4.1) for this instance of the DH-HMAC-CHAP protocol requested secure channel concatenation (i.e., the SC\_C field in that message was set to a non-zero value).

**DH Value Length (DHVLEN):** Diffie-Hellman exponential length. This length shall be a multiple of 4. If the DH group identifier is cleared to 0h (i.e., NULL DH exchange), this field shall be cleared to 0h. Otherwise, it shall be set to the length in bytes of the DH Value.

**Sequence Number (SEQNUM):** 32-bit sequence number  $S_1$ . A random non-zero value shall be used as the initial value. The sequence number is incremented modulo  $2^{32}$  after each use, except that the value 0h is skipped (i.e., incrementing the value FFFFFFFh results in the value 0000001h).

**Challenge Value (CVAL):** Shall be set to a random challenge value  $C_1$  (refer to section 8.3.4.5.7). Each challenge value should be unique and unpredictable, since repetition of a challenge value in conjunction with the same key may reveal information about the key or the correct response to this challenge. The algorithm for generating the challenge value is outside the scope of this specification. Randomness of the challenge value is crucial to the security of the protocol (refer to section 8.3.4.5.7). The CVAL length is the same as the length of the selected hash function (i.e., HL).

**DH Value (DHV):** Diffie-Hellman exponential. If the DH Value Length is cleared to 0h, this field is not present. The DH value shall be set to the value of  $g^x \text{ mod } p$ , where  $x$  is a random number selected by the controller that shall be at least 256 bits long (refer to section 8.3.4.5.7) and  $p$  and  $g$  shall have the values indicated in Figure 741, based on the selected DH group identifier.

Upon receiving a DH-HMAC-CHAP\_Challenge message, if:

- the Hash Length (HL) does not match the value specified in Figure 740 for the selected hash function;
- the Sequence Number (SEQNUM) is cleared to 0h;
- DHgID is cleared to 0h and the AUTH\_Negotiate message (refer to section 8.3.4.4.1) that the host sent for this instance of the DH-HMAC-CHAP protocol requested secure channel concatenation (i.e., the SC\_C field in that message is set to a non-zero value);
- DHgID is non-zero and the DH Value Length (DHVLEN) is cleared to 0h; or
- DHgID is non-zero and the DH Value (DHV) is 0, 1, or  $p-1$ ;

then the host shall:

- reply with an AUTH\_Failure2 message having reason code 'Authentication failure' and reason code explanation 'Incorrect payload'; and
- disconnect the NVMe over Fabrics connection.

#### 8.3.4.5.4 DH-HMAC-CHAP\_Reply Message

The DH-HMAC-CHAP\_Reply message is sent from the host to the controller. The host may request authentication of the controller to enable bidirectional authentication, by including a DH-HMAC-CHAP challenge value  $C_2$  in this message. The challenge value  $C_2$  shall be different from the challenge value  $C_1$  received in the DH-HMAC-CHAP\_Challenge message.

The format of the DH-HMAC-CHAP\_Reply message is shown in Figure 743.

**Figure 743: DH-HMAC-CHAP\_Reply message format**

Bytes	Description
0	<b>Authentication Type (AUTH_TYPE):</b> 01h (i.e., DH-HMAC-CHAP)
1	<b>Authentication Identifier (AUTH_ID):</b> 02h (i.e., DH-HMAC-CHAP_Reply)
3:2	Reserved
5:4	<b>Transaction Identifier (T_ID):</b> 16-bit transaction identifier
6	<b>Hash Length (HL):</b> Length in bytes of the selected hash function
7	Reserved

Figure 743: DH-HMAC-CHAP\_Reply message format

Bytes	Description		
8	<b>Challenge Valid (CVALID):</b>	<b>Value</b>	<b>Definition</b>
		00h	The Challenge Value is not valid
		01h	The Challenge Value is valid
		All other values	Reserved
9	Reserved		
11:10	<b>DH Value Length (DHVLEN):</b> Length in bytes of DH value. If no DH value is included in the message, then this field is cleared to 0h.		
15:12	<b>Sequence Number (SEQNUM):</b> Sequence number $S_2$ .		
15+HL:16	<b>Response Value (RVAL):</b> Response $R_1$ .		
15+2*HL:16+HL	<b>Challenge Value (CVAL):</b> Challenge $C_2$ , if valid (i.e., if the CVALID field is set to 01h), cleared to 0h otherwise.		
15+2*HL+DHVLEN:6+2*HL	<b>DH Value (DHV):</b> DH exponential $g^y \text{ mod } p$ . This field is not present (i.e., the CVAL field is the last field in the message) if DHVLEN is cleared to 0h.		

**Hash Length (HL):** Shall be set to the length in bytes of the selected hash function, as specified in Figure 740.

**Challenge Valid:** If the host does not require bidirectional authentication or no establishment of a secure channel after unidirectional authentication is sought (refer to section 8.3.4.5.9), this field shall be cleared to 0h. Otherwise, this field shall be set to 01h.

**DH Value Length (DHVLEN):** Diffie-Hellman exponential length. This length shall be a multiple of 4. If the DH group identifier is cleared to 0h (i.e., NULL DH exchange), this field shall be cleared to 0h. Otherwise, it shall be set to the length in bytes of the DH Value.

**Sequence Number (SEQNUM):** 32-bit sequence number  $S_2$ . A random non-zero value shall be used as the initial value. The sequence number is incremented modulo  $2^{32}$  after each use, except that the value 0h is skipped (i.e., incrementing the value FFFFFFFFh results in the value 00000001h). The value 0h is used to indicate that bidirectional authentication is not performed, but a challenge value  $C_2$  is carried in order to generate a pre-shared key (PSK) for subsequent establishment of a secure channel (refer to section 8.3.4.5.9).

**Response Value (RVAL):** DH-HMAC-CHAP response value  $R_1$ . The value of  $R_1$  is computed using the hash function  $H(\ )$  selected by the HashID parameter in the DH-HMAC-CHAP\_Challenge message, and the augmented challenge  $C_{a1}$ . If the NULL DH group has been selected, the augmented challenge  $C_{a1}$  is equal to the challenge  $C_1$  received from the controller (i.e.,  $C_{a1} = C_1$ ). If a non-NULL DH group has been selected, the augmented challenge is computed applying the HMAC function using the hash function  $H(\ )$  selected by the HashID parameter in the DH-HMAC-CHAP\_Challenge message with the hash of the ephemeral DH key resulting from the combination of the random value  $y$  selected by the host with the DH exponential (i.e.,  $g^x \text{ mod } p$ ) received from the controller as HMAC key (refer to RFC 2104) to the challenge  $C_1$  (i.e.,  $C_{a1} = \text{HMAC}(H((g^x \text{ mod } p)^y \text{ mod } p), C_1) = \text{HMAC}(H(g^{xy} \text{ mod } p), C_1)$ ). The value of  $R_1$  shall be computed applying the HMAC function using the hash function  $H(\ )$  selected by the HashID parameter in the DH-HMAC-CHAP\_Challenge message with key  $K_h$  as HMAC key to the concatenation of the augmented challenge  $C_{a1}$ , the sequence number  $S_1$ , the transaction identifier  $T\_ID$ , the secure channel concatenation indication  $SC\_C$  sent in the AUTH\_Negotiate message, the eight ASCII characters "HostHost" to indicate the host is computing the reply, the host NQN not including the null terminator, a 00h byte, and the NVM subsystem NQN not including the null terminator (i.e.,  $R_1 = \text{HMAC}(K_h, C_{a1} \parallel S_1 \parallel T\_ID \parallel SC\_C \parallel \text{"HostHost"} \parallel \text{NQN}_h \parallel 00h \parallel \text{NQN}_c)$ ). Using C language notation:

$$C_{a1} = (\text{DHgID} == 00h) ? C_1 : \text{HMAC}(H((g^x \text{ mod } p)^y \text{ mod } p), C_1)$$

$$R_1 = \text{HMAC}(K_h, C_{a1} \parallel S_1 \parallel T\_ID \parallel SC\_C \parallel \text{"HostHost"} \parallel \text{NQN}_h \parallel 00h \parallel \text{NQN}_c)$$

**Challenge Value (CVAL):** Shall be set to a random challenge value  $C_2$  (refer to section 8.3.4.5.7). Each challenge value should be unique and unpredictable, since repetition of a challenge value in conjunction



with the same key may reveal information about the key or the correct response to this challenge. The algorithm for generating the challenge value is outside the scope of this specification. Randomness of the challenge value is crucial to the security of the protocol (refer to section 8.3.4.5.7). The CVAL length is the same as the length of the selected hash function (i.e., HL).

**DH Value (DHV):** Diffie-Hellman exponential. If the DH Value Length is cleared to 0h, this field is not present. The DH Value shall be set to the value of  $g^y \text{ mod } p$ , where  $y$  is a random number selected by the host that shall be at least 256 bits long (refer to section 8.3.4.5.7) and  $p$  and  $g$  shall have the values indicated in Figure 741, based on the selected DH group identifier.

Upon receiving a DH-HMAC-CHAP\_Reply message, if:

- the Hash Length (HL) does not match the value specified in Figure 740 for the selected hash function;
- DHgID is non-zero and the DH Value Length (DHVLEN) is cleared to 0h; or
- DHgID is non-zero and the DH Value (DHV) is 0, 1, or  $p-1$ ;

then the controller shall:

- reply with an AUTH\_Failure1 message having reason code 'Authentication failure' and reason code explanation 'Incorrect payload'; and
- disconnect the NVMe over Fabrics connection.

In addition, the controller shall:

- check the challenge value  $C_2$ , if the Challenge Valid field is set to 01h, to verify it is different from the challenge value  $C_1$  the controller previously sent. If  $C_2$  is equal to  $C_1$ , the controller shall:
  - reply with an AUTH\_Failure1 message having reason code 'Authentication failure' and reason code explanation 'Authentication failed'; and
  - disconnect the NVMe over Fabrics connection; and
- verify the response value  $R_1$  using the negotiated hash function. If verification of the response value  $R_1$  does not succeed, the controller shall:
  - reply with an AUTH\_Failure1 message having reason code 'Authentication failure' and reason code explanation 'Authentication failed'; and
  - disconnect the NVMe over Fabrics connection.

If verification of the response value  $R_1$  succeeds, the host has been authenticated and the controller shall continue with a DH-HMAC-CHAP\_Success1 message.

### 8.3.4.5.5 DH-HMAC-CHAP\_Success1 Message

The DH-HMAC-CHAP\_Success1 message is sent from the controller to the host and indicates that the controller has successfully authenticated the host. The format of the DH-HMAC-CHAP\_Success1 message is shown in Figure 744.

**Figure 744: DH-HMAC-CHAP\_Success1 message format**

Bytes	Description
0	<b>Authentication Type (AUTH_TYPE):</b> 01h (i.e., DH-HMAC-CHAP)
1	<b>Authentication Identifier (AUTH_ID):</b> 03h (i.e., DH-HMAC-CHAP_Success1)
3:2	Reserved
5:4	<b>Transaction Identifier (T_ID):</b> 16-bit transaction identifier
6	<b>Hash Length (HL):</b> Length in bytes of the selected hash function
7	Reserved

Figure 744: DH-HMAC-CHAP\_Success1 message format

Bytes	Description								
8	<b>Response Valid (RVALID):</b> <table border="1" data-bbox="617 352 1182 470"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00h</td> <td>The Response Value is not valid</td> </tr> <tr> <td>01h</td> <td>The Response Value is valid</td> </tr> <tr> <td>All other values</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00h	The Response Value is not valid	01h	The Response Value is valid	All other values	Reserved
Value	Definition								
00h	The Response Value is not valid								
01h	The Response Value is valid								
All other values	Reserved								
15:9	Reserved								
15+HL:16	<b>Response Value (RVAL):</b> Response R <sub>2</sub> , if valid (i.e., if the RVALID field is set to 01h), cleared to 0h otherwise								

**Hash Length (HL):** Shall be set to the length in bytes of the selected hash function, as specified in Figure 740.

**Response Valid:** If the host did not request authentication of the controller (i.e., bidirectional authentication) this field shall be cleared to 0h to indicate that no response is conveyed (i.e., the Response Value field is not valid). If the host did request authentication of the controller, this field shall be set to 01h.

**Response Value (RVAL):** DH-HMAC-CHAP response value R<sub>2</sub>. The value of R<sub>2</sub> is computed using the hash function H( ) selected by the HashID parameter of the DH-HMAC-CHAP\_Challenge message, and the augmented challenge C<sub>a2</sub>. If the NULL DH group has been selected, the augmented challenge C<sub>a2</sub> is equal to the challenge C<sub>2</sub> received from the host (i.e., C<sub>a2</sub> = C<sub>2</sub>). If a non-NULL DH group has been selected, the augmented challenge is computed applying the HMAC function using the hash function H( ) selected by the HashID parameter in the DH-HMAC-CHAP\_Challenge message with the hash of the ephemeral DH key resulting from the combination of the random value x selected by the controller with the DH exponential (i.e., g<sup>y</sup> mod p) received from the host as HMAC key (refer to RFC 2104) to the challenge C<sub>2</sub> (i.e., C<sub>a2</sub> = HMAC(H((g<sup>y</sup> mod p)<sup>x</sup> mod p), C<sub>2</sub>) = HMAC(H(g<sup>xy</sup> mod p), C<sub>2</sub>). The value of R<sub>2</sub> shall be computed applying the HMAC function using the hash function H( ) selected by the HashID parameter in the DH-HMAC-CHAP\_Challenge message with key K<sub>c</sub> as HMAC key to the concatenation of the augmented challenge C<sub>a2</sub>, the sequence number S<sub>2</sub>, the transaction identifier T\_ID, the secure channel concatenation indication SC\_C received in the AUTH\_Negotiate message, the ten ASCII characters "Controller" to indicate the controller is computing the reply, the NVM Subsystem NQN not including the null terminator, a 00h byte, and the host NQN not including the null terminator (i.e., R<sub>2</sub> = HMAC(K<sub>c</sub>, C<sub>a2</sub> || S<sub>2</sub> || T\_ID || SC\_C || "Controller" || NQN<sub>c</sub> || 00h || NQN<sub>h</sub>)). Using C language notation:

$$C_{a2} = (\text{DHgID} == 00\text{h}) ? C_2 : \text{HMAC}(\text{H}((g^y \text{ mod } p)^x \text{ mod } p), C_2)$$

$$R_2 = \text{HMAC}(K_c, C_{a2} || S_2 || T\_ID || SC\_C || \text{"Controller"} || \text{NQN}_c || 00\text{h} || \text{NQN}_h)$$

Upon receiving a DH-HMAC-CHAP\_Success1 message:

- if the Hash Length (HL) does not match the value specified in Figure 740 for the selected hash function, the host shall:
  - reply with an AUTH\_Failure2 message having reason code 'Authentication failure' and reason code explanation 'Incorrect payload'; and
  - disconnect the NVMe over Fabrics connection; and
- if the Response Valid field is set to 01h, the host shall verify the response value R<sub>2</sub> using the negotiated hash function and DH group. If verification of the response value R<sub>2</sub> does not succeed, the host shall:
  - reply with an AUTH\_Failure2 message having reason code 'Authentication failure' and reason code explanation 'Authentication failed'; and
  - disconnect the NVMe over Fabrics connection.

If verification of the response value R<sub>2</sub> succeeds, the controller has been authenticated and the host shall continue with a DH-HMAC-CHAP\_Success2 message.

### 8.3.4.5.6 DH-HMAC-CHAP\_Success2 Message

The DH-HMAC-CHAP\_Success2 message is sent from the host to the controller and indicates that the host has successfully authenticated the controller. The format of the DH-HMAC-CHAP\_Success2 message is shown in Figure 745.

**Figure 745: DH-HMAC-CHAP\_Success2 message format**

Bytes	Description
0	<b>Authentication Type (AUTH_TYPE):</b> 01h (i.e., DH-HMAC-CHAP)
1	<b>Authentication Identifier (AUTH_ID):</b> 04h (i.e., DH-HMAC-CHAP_Success2)
3:2	Reserved
5:4	<b>Transaction Identifier (T_ID):</b> 16-bit transaction identifier
15:6	Reserved

### 8.3.4.5.7 DH-HMAC-CHAP Security Requirements

In order to authenticate with the DH-HMAC-CHAP protocol, each host or controller uses a DH-HMAC-CHAP key that is associated with the entity's NQN. A DH-HMAC-CHAP key is unidirectional (i.e., used only for one direction of an authentication transaction). A DH-HMAC-CHAP key should not be associated with more than one NQN as this opens security vulnerabilities. All DH-HMAC-CHAP implementations should check for use of the same key with more than one NQN and should generate an administrative warning if this situation occurs (e.g., as a result of configuring a DH-HMAC-CHAP key to verify authentication of another entity).

The DH-HMAC-CHAP key is derived from an administratively configured secret (refer to section 8.3.4.5.8). Each host and NVM subsystem shall support:

- transforming the provided secret into a key applying the HMAC function using the hash function specified in the secret representation (refer to section 8.3.4.5.8) with the secret as HMAC key to the concatenation of its own NQN not including the null terminator and the seventeen ASCII characters "NVMe-over-Fabrics" (i.e., key = HMAC(secret, NQN || "NVMe-over-Fabrics")). This transformation ensures the resulting key is uniquely associated with the entity identified by the NQN; and
- using the provided secret as a key. This is intended for use with key management solutions able to ensure that key is uniquely associated with the entity identified by the NQN.

NVM subsystems should support the ability to use a different NVM subsystem key with each host. Hosts should support the ability to use a different host key with each NVM subsystem. NVM subsystems should support the ability to use a different NVM subsystem secret with each host. Hosts should support the ability to use a different host secret with each NVM subsystem.

If an implementation of NVMe over Fabrics is capable of functioning as both a host and an NVM subsystem, then that implementation shall use either:

- one NQN for the host functionality and a different NQN for the NVM subsystem functionality; or
- one NQN for both host functionality and NVM subsystem functionality.

DH-HMAC-CHAP implementations may reuse a DH exponential (e.g.,  $g^x \bmod p$  or  $g^y \bmod p$ ). The primary risk in allowing reuse of a DH exponential is replay of a prior authentication sequence based on the attacker reusing the other exponential. For DH-HMAC-CHAP, replay is prevented with extremely high probability by the requirement that all challenges be randomly generated. See section 2.12 of RFC 7296 for guidance on DH exponential reuse.

The security of the DH-HMAC-CHAP protocol requires secrets, challenges, and DH exponents (i.e.,  $x$  and  $y$ ) to be generated from actual randomness. For a discussion of randomness and sources of randomness, refer to RFC 4086.

Implementations shall use a cryptographic random number generator that should be seeded with at least 256 bits of entropy to generate random numbers for this protocol. The secret provisioning mechanism for

each host and controller is outside of scope of this specification. For instance, secrets could be provisioned via an encrypted HTTPS-based connection.

#### 8.3.4.5.8 Secret Representation

In order to facilitate provisioning, management, and interchange (e.g., copy & paste in an administrative configuration tool) of secrets, all NVMe over Fabrics entities shall support the following ASCII representation of secrets:

DHHC-1:xx:<Base64 encoded string>:

Where:

- "DHHC-1" indicates this is a version 1 representation of a secret for the DH-HMAC-CHAP protocol;
- ':' is used both as a separator and a terminator;
- xx indicates the hash function to be used to transform the secret in key (refer to section 8.3.4.5.7), encoded as the ASCII representation of the hexadecimal value specified in Figure 740 (e.g., the two ASCII characters "01" indicate SHA-256). The two ASCII characters "00" indicate no transform (i.e., use the secret as a key); and
- The Base64 (refer to RFC 4648) string encodes the secret (32, 48, or 64 bytes binary) followed by the CRC-32 (refer to RFC 1952) of the secret (4 bytes binary).

As an example, the 32-byte secret:

89AEB31A 874EAF84 841B4673 6B0DFDF2 BA58D30A A2A545A3 E235A352 1E07594Ch

has the CRC-32 A70D69FAh, that is represented in little endian format (i.e., FA690DA7h) for concatenation to the secret, resulting in the following:

"DHHC-1:00:ia6zGodOr4SEG0Zzaw398rpY0wqipUWj4jWjUh4HWUz6aQ2n:"

when intended to be used as a key without transform.

When provided with a secret in this format, NVMe over Fabrics entities shall verify the validity of the provided secret by computing the CRC-32 value of the secret and checking the computed value with the provided value. If they do not match, then the secret shall not be used.

#### 8.3.4.5.9 Generated PSK for TLS

When used with a non-NUL DH exchange, the DH-HMAC-CHAP protocol is able to generate a session key  $K_S$  used to generate a pre-shared key (PSK) to establish a secure channel session with the TLS protocol between host and controller for NVMe/TCP. A TLS session is concatenated to an authentication transaction when the SC\_C indication is set to NEWTLSPSK in the AUTH\_Negotiate message (refer to section 8.3.4.1). A TLS session shall not be concatenated to an authentication transaction if the involved host and controller are administratively configured with a PSK for use with each other. In this case, host and controller shall only establish a TLS session based on the retained PSK derived from that configured PSK.

The session key  $K_S$  shall be computed from the ephemeral DH key (i.e.,  $g^{xy} \bmod p$ ) generated during the DH-HMAC-CHAP transaction by applying the hash function  $H(\ )$  selected by the HashID parameter in the DH-HMAC-CHAP\_Challenge message (i.e.,  $K_S = H(g^{xy} \bmod p)$ ). The size of the session key  $K_S$  is determined by the selected hash function, as shown in Figure 740. Specifically:

- The host computes  $K_S$  as the hash of the ephemeral DH key resulting from the combination of the random value  $y$  selected by the host with the DH exponential (i.e.,  $g^x \bmod p$ ) received from the controller (i.e.,  $K_S = H((g^x \bmod p)^y \bmod p) = H(g^{xy} \bmod p)$ ).
- The controller computes  $K_S$  as the hash of the ephemeral DH key resulting from the combination of the random value  $x$  selected by the controller with the DH exponential (i.e.,  $g^y \bmod p$ ) received from the host (i.e.,  $K_S = H((g^y \bmod p)^x \bmod p) = H(g^{xy} \bmod p)$ ).

The generated PSK for TLS shall be computed applying the HMAC function using the hash function  $H(\ )$  selected by the HashID parameter in the DH-HMAC-CHAP\_Challenge message with the session key  $K_S$  as key to the concatenation of the two challenges  $C_1$  and  $C_2$  (i.e., generated PSK =  $HMAC(K_S, C_1 || C_2)$ ).

This generated PSK should be replaced periodically (e.g., every hour) or on demand by performing a reauthentication on the Admin queue of the association with the appropriate SC\_C value (refer to section 8.3.4.1).

The host may request secure channel concatenation with the TLS protocol by setting the SC\_C field in the AUTH\_Negotiate message to NEWTLSPSK while performing only unidirectional authentication. In this case the host shall still send a challenge value C<sub>2</sub> to the controller and clear the sequence number S<sub>2</sub> to 0h to indicate that controller authentication is not requested.

#### 8.3.4.5.10 Mapping of DH-HMAC-CHAP Messages to Authentication Commands

The DH-HMAC-CHAP\_Reply message and the DH-HMAC-CHAP\_Success2 message are sent from the host to the controller, therefore they are mapped to the Authentication Send command. The DH-HMAC-CHAP\_Challenge message and the DH-HMAC-CHAP\_Success1 message are sent from the controller to the host, therefore they are mapped to the Authentication Receive command.

#### 8.3.4.5.11 DH-HMAC-CHAP Configuration

DH-HMAC-CHAP implementations that do not use an AVE (refer to section 8.3.4.6 for AVE information) shall be able to be provisioned with their own DH-HMAC-CHAP secret and with a verification secret per each remote entity that is able to be authenticated. DH-HMAC-CHAP implementations that use an AVE shall be able to be provisioned with their own DH-HMAC-CHAP secret and the parameters for accessing the AVE (refer to section 8.3.4.6.2).

This section uses the term configuration to indicate an NVMe internal state that controls the use of DH-HMAC-CHAP. That internal state may or may not be externally configurable for a host or NVM subsystem. For an NVMe/TCP implementation that supports DH-HMAC-CHAP, a DH-HMAC-CHAP configuration may apply to:

- a single connection;
- a group of connections; or
- all connections.

DH-HMAC-CHAP implementations that do not support secure channel concatenation with the TLS protocol shall support configuring use of DH-HMAC-CHAP using one or more of the configurations shown in Figure 746.

**Figure 746: DH-HMAC-CHAP Configurations**

Configuration	Description
Authentication Disabled	Authentication of a remote entity not allowed
Authentication Permitted	Authentication of a remote entity allowed
Authentication Required	Authentication of a remote entity required

NVM subsystems that do not support secure channel concatenation with the TLS protocol shall set the ATR and ASCR bits in the AUTHREQ field in the response of a successful Connect command (refer to Figure 548) as defined in Figure 747.

**Figure 747: NVM Subsystem AUTHREQ Settings for DH-HMAC-CHAP**

Configuration	ASCR	ATR
Authentication Disabled	0	0
Authentication Permitted	0	0
Authentication Required	0	1

Hosts that do not support secure channel concatenation with the TLS protocol shall behave as defined in Figure 748, according to the ATR and ASCR bits in the AUTHREQ field received in the response of a successful Connect command (refer to Figure 548).

**Figure 748: DH-HMAC-CHAP Host Behavior**

Host Configuration	ASCR	ATR	Action
Authentication Disabled	0	0	Do not begin an authentication transaction
	0	1	Disconnect from the NVM subsystem
	1	any	
Authentication Permitted	0	0	If the TASC field in the Discovery Log Page Entry for the remote entity is set to 01b or set to 10b, then begin an authentication transaction with the SC_C field set to NOSC. If the TASC field in the Discovery Log Page Entry for the remote entity is cleared to 00b or set to 11b or if no Discovery Log Page Entry for the remote entity has been retrieved, then do not begin an authentication transaction.
		1	any
	0	1	Begin an authentication transaction with the SC_C field set to NOSC
Authentication Required	0	any	Begin an authentication transaction with the SC_C field set to NOSC
	1	any	Disconnect from the NVM subsystem

NVM subsystems that do not support secure channel concatenation with the TLS protocol shall behave as defined in Figure 749, according to the SC\_C field in a received AUTH\_Negotiate message (refer to section 8.3.4.4.1).

**Figure 749: DH-HMAC-CHAP NVM Subsystem Behavior**

Subsystem Configuration	SC_C Value	Action
Authentication Disabled	Any	Disconnect from the host
Authentication Permitted	NOSC	Participate in the authentication transaction
	NEWTLSPSK or REPLACETLSPSK	Disconnect from the host
Authentication Required	NOSC	Participate in the authentication transaction
	NEWTLSPSK or REPLACETLSPSK	Disconnect from the host

For example, consider a host that supports DH-HMAC-CHAP without secure channel concatenation with the TLS protocol configured with Authentication Permitted and an NVM subsystem that supports DH-HMAC-CHAP without secure channel concatenation with the TLS protocol configured with Authentication Required. As defined in Figure 747, the NVM subsystem clears the ASCR bit to '0' and sets the ATR bit to '1' in the Connect response. As defined in Figure 748, upon receiving the Connect response the host begins an authentication transaction with the SC\_C field set to NOSC. As defined in Figure 749, the NVM subsystem participates in the authentication transaction.

DH-HMAC-CHAP implementations that support secure channel concatenation with the TLS protocol shall support configuring use of DH-HMAC-CHAP with secure channel concatenation with the TLS protocol using one or more of the configurations shown in Figure 750.

**Figure 750: DH-HMAC-CHAP with TLS Concatenation Configurations**

Configuration	Description
Authentication Disabled	TLS Disabled Authentication and set up of a secure channel with a remote entity not allowed
Authentication Permitted	TLS Disabled Authentication of a remote entity allowed, set up of a secure channel not allowed
	TLS Permitted Authentication and set up of a secure channel with a remote entity allowed

**Figure 750: DH-HMAC-CHAP with TLS Concatenation Configurations**

Configuration		Description
Authentication Required	TLS Disabled	Authentication of a remote entity required, set up of a secure channel not allowed
	TLS Permitted	Authentication of a remote entity required, set up of a secure channel allowed
	TLS Required	Authentication and set up of a secure channel with a remote entity required

NVM subsystems that support secure channel concatenation with the TLS protocol shall set the ATR and ASCR bits in the AUTHREQ field in the response of a successful Connect command on an Admin Queue over a TCP channel without TLS (refer to Figure 548) as defined in Figure 751.

**Figure 751: NVM Subsystem AUTHREQ Settings for DH-HMAC-CHAP with TLS Concatenation**

Configuration		ASCR	ATR
Authentication Disabled	TLS Disabled	0	0
	TLS Permitted	0	0
Authentication Permitted	TLS Disabled	0	1
	TLS Permitted	0	1
	TLS Required	1	0

Hosts that support secure channel concatenation with the TLS protocol shall behave as defined Figure 752, according to the ATR and ASCR bits in the AUTHREQ field received in the response of a successful Connect command on an Admin Queue over a TCP channel without TLS (refer to Figure 548).

**Figure 752: DH-HMAC-CHAP with TLS Concatenation Host Behavior**

Host Configuration		ASCR	ATR	Action
Authentication Disabled	TLS Disabled	0	0	Do not begin an authentication transaction
		0	1	Disconnect from the NVM subsystem
		1	any	
Authentication Permitted	TLS Disabled	0	0	If the TASC field in the Discovery Log Page Entry for the remote entity is set to 01b or set to 10b, then begin an authentication transaction with the SC_C field set to NOSC.
		0	1	Begin an authentication transaction with the SC_C field set to NOSC
		0	any	If the TASC field in the Discovery Log Page Entry for the remote entity is cleared to 00b or set to 11b or if no Discovery Log Page Entry for the remote entity has been retrieved, then do not begin an authentication transaction.
		1	any	Disconnect from the NVM subsystem

**Figure 752: DH-HMAC-CHAP with TLS Concatenation Host Behavior**

Host Configuration		ASCR	ATR	Action		
	TLS Permitted	0	0	If the TASC field in the Discovery Log Page Entry for the remote entity is set to 01b, then begin an authentication transaction with the SC_C field set to NOSC.  If the TASC field in the Discovery Log Page Entry for the remote entity is set to 10b, then begin an authentication transaction with the SC_C field set to NEWTLSPSK.  If the TASC field in the Discovery Log Page Entry for the remote entity is cleared to 00b or set to 11b or if no Discovery Log Page Entry has been retrieved for the remote entity, then do not begin an authentication transaction.		
				0	1	Begin an authentication transaction with the SC_C field set to NOSC
				1	any	Begin an authentication transaction with the SC_C field set to NEWTLSPSK
Authentication Required	TLS Disabled	0	any	Begin an authentication transaction with the SC_C field set to NOSC		
		1	any	Disconnect from the NVM subsystem		
	TLS Permitted	0	any	Begin an authentication transaction with the SC_C field set to NOSC		
		1	any	Begin an authentication transaction with the SC_C field set to NEWTLSPSK		
TLS Required	any	any	Begin an authentication transaction with the SC_C field set to NEWTLSPSK			

NVM subsystems that support secure channel concatenation with the TLS protocol shall behave as defined in Figure 753, according to the SC\_C field in a received AUTH\_Negotiate message on an Admin Queue over a TCP channel without TLS (refer to section 8.3.4.4.1).

**Figure 753: DH-HMAC-CHAP with TLS Concatenation NVM Subsystem Behavior**

Subsystem Configuration		SC_C Value	Action
Authentication Disabled	TLS Disabled	any	Disconnect from the host
Authentication Permitted	TLS Disabled	NOSC	Participate in the authentication transaction
		NEWTLSPSK or REPLACETLSPSK	Disconnect from the host
	TLS Permitted	NOSC	Participate in the authentication transaction
		NEWTLSPSK	Participate in the authentication transaction and establish a TLS channel as described in section 8.3.4.3
	REPLACETLSPSK	Disconnect from the host	



**Figure 753: DH-HMAC-CHAP with TLS Concatenation NVM Subsystem Behavior**

Subsystem Configuration		SC_C Value	Action
Authentication Required	TLS Disabled	NOSC	Participate in the authentication transaction
		NEWTLSPSK or REPLACETLSPSK	Disconnect from the host
	TLS Permitted	NOSC	Participate in the authentication transaction
		NEWTLSPSK	Participate in the authentication transaction and establish a TLS channel as described in section 8.3.4.3
		REPLACETLSPSK	Disconnect from the host
	TLS Required	NOSC or REPLACETLSPSK	Disconnect from the host
NEWTLSPSK		Participate in the authentication transaction and establish a TLS channel as described in section 8.3.4.3	

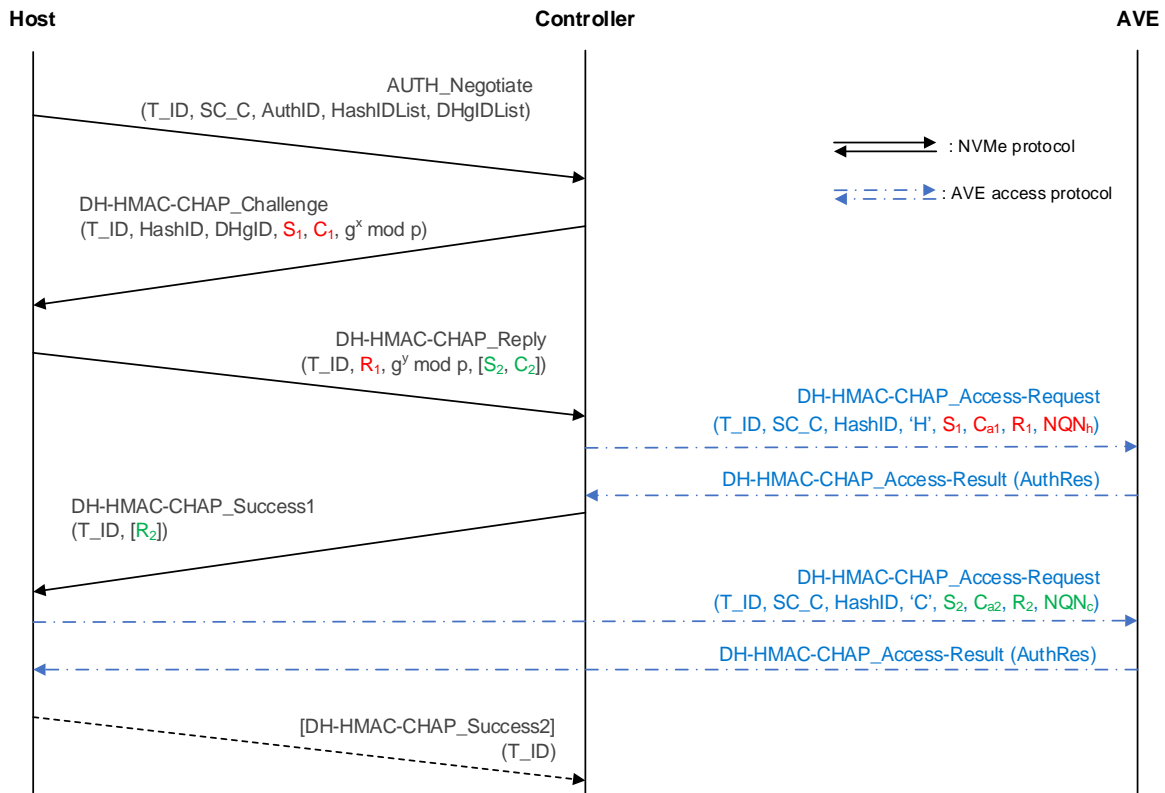
As an example, consider a host that supports DH-HMAC-CHAP with secure channel concatenation with the TLS protocol configured with Authentication Permitted, TLS Permitted and an NVM subsystem that supports DH-HMAC-CHAP with secure channel concatenation with the TLS protocol configured with Authentication Required, TLS Required. As defined in Figure 751, the NVM subsystem sets the ASCR bit to '1' and clears the ATR bit to '0' in the Connect response. As defined in Figure 752, upon receiving the Connect response the host begins an authentication transaction with the SC\_C field set to NEWTLSPSK. As defined in Figure 753, the NVM subsystem participates in the authentication transaction and establishes a TLS channel.

### 8.3.4.6 DH-HMAC-CHAP Authentication Verification Entity

#### 8.3.4.6.1 Overview

A DH-HMAC-CHAP Authentication Verification Entity (AVE) is a service that performs the DH-HMAC-CHAP authentication verification function on behalf of an NVMe entity (i.e., a host or a controller). An example of a DH-HMAC-CHAP authentication transaction with AVE is shown in Figure 754, using the notation shown in Figure 738.

**Figure 754: Example of DH-HMAC-CHAP authentication transaction with AVE**



As shown in Figure 754, a controller using the AVE service delegates to the AVE the verification of the response  $R_1$  received from the host by passing the relevant DH-HMAC-CHAP authentication transaction parameters to the AVE through a DH-HMAC-CHAP\_Access-Request message. A host using the AVE service delegates to the AVE the verification of the response  $R_2$  received from the controller by passing the relevant DH-HMAC-CHAP authentication transaction parameters to the AVE through a DH-HMAC-CHAP\_Access-Request message. In both cases, the AVE replies with a DH-HMAC-CHAP\_Access-Result message containing the result of the authentication verification (refer to section 8.3.4.6.3).

Use of the AVE service by an NVMe entity is optional and is determined by configuration of the NVMe entity. If an NVMe entity uses the AVE, then provisioning of DH-HMAC-CHAP information on that entity is reduced to only that entity’s DH-HMAC-CHAP secret (refer to section 8.3.4.5.8) and the parameters (refer to section 8.3.4.6.2) for accessing the AVE (i.e., no DH-HMAC-CHAP keys are required to be provisioned for verification of responses received from other NVMe entities).

An AVE is required to maintain the following information for each NVMe entity:

- The NQN of that entity (i.e.,  $NQN_e$ ),
- the DH-HMAC-CHAP key associated with that entity (i.e.,  $K_e$ ), and
- the PSK shared between that entity and the AVE (i.e.,  $PSK_{ea}$ ).

$K_e$  is used to perform the authentication verification function (refer to section 8.3.4.6.3) and  $PSK_{ea}$  is used to establish a secure connection with the AVE (refer to section 8.3.4.6.2). An AVE shall support all hash functions defined for DH-HMAC-CHAP (refer to section 8.3.4.5.2).

To facilitate dynamic discovery of the transport addresses of an AVE through a Discovery Controller (refer to section 8.3.4.6.4) and to simplify establishing a secure connection to an AVE (refer to section 8.3.4.6.2), an AVE is identified at by an NQN ( $NQN_{AVE}$ ).

### 8.3.4.6.2 AVE Connections

An NVMe entity (i.e., a host or a controller) connection with a DH-HMAC-CHAP AVE shall use TLS version 1.3 (refer to RFC 8446) with pre-shared key (PSK) authentication, as specified for NVMe/TCP (refer to the NVM Express TCP Transport Specification).

In order to establish a TLS connection with an AVE, an NVMe entity requires a PSK shared between that entity and the AVE (i.e.,  $PSK_{ea}$ ) for authentication of the TLS connection.  $PSK_{ea}$  shall be either derived from the DH-HMAC-CHAP secret, if the DH-HMAC-CHAP secret representation specifies a hash function (refer to section 8.3.4.5.8), or provisioned on the NVMe entity through a configured PSK, as specified for NVMe/TCP.

Derivation of  $PSK_{ea}$  from a DH-HMAC-CHAP secret shall use the HKDF-Extract and HKDF-Expand-Label operations (refer to RFC 5869 and RFC 8446) in the following order:

1.  $PRK = \text{HKDF-Extract}(0, \text{DH-HMAC-CHAP secret});$  and
2.  $PSK_{ea} = \text{HKDF-Expand-Label}(PRK, \text{"A-V-Entity"} \parallel NQN_{AVE}, NQN_e, \text{Length}(\text{DH-HMAC-CHAP secret})),$

where  $NQN_{AVE}$  is the NQN of the AVE and  $NQN_e$  is the NQN of the NVMe entity. The hash function used with HKDF shall be the one specified in the DH-HMAC-CHAP secret representation (refer to section 8.3.4.5.8). This transform requires that the NVMe entity knows  $NQN_{AVE}$ .

Derivation of  $PSK_{ea}$  from a DH-HMAC-CHAP secret is not possible if the DH-HMAC-CHAP secret representation does not specify a hash function. In this case  $PSK_{ea}$  shall be provisioned on the NVMe entity through a configured PSK, as specified for NVMe/TCP, and that configured PSK should be different than the DH-HMAC-CHAP secret for that entity.

Derivation of  $PSK_{ea}$  from a configured PSK shall use the HKDF-Extract and HKDF-Expand-Label operations in the following order:

1.  $PRK = \text{HKDF-Extract}(0, \text{Configured PSK});$  and
2.  $PSK_{ea} = \text{HKDF-Expand-Label}(PRK, \text{"A-V-Entity"} \parallel NQN_{AVE}, NQN_e, \text{Length}(\text{Configured PSK})),$

where  $NQN_{AVE}$  is the NQN of the AVE and  $NQN_e$  is the NQN of the NVMe entity. The hash function used with HKDF shall be the one specified in the PSK interchange format (refer to the NVM Express TCP Transport Specification). If no hash function is specified in the PSK interchange format, then the configured PSK shall be used as  $PSK_{ea}$ . This transform requires that the NVMe entity knows  $NQN_{AVE}$ .

The TLS connection with the AVE shall be performed as specified in the TLS PSK and PSK Identity Derivation section of the NVM Express TCP Transport Specification, with  $PSK_{ea}$  used as the Retained PSK,  $NQN_e$  used as the host NQN, and  $NQN_{AVE}$  used as the controller NQN. Mandatory and recommended cipher suites for this TLS connection are specified in the Mandatory and Recommended Cipher Suites section of the NVM Express TCP Transport Specification. This TLS connection may be used for multiple authentication verifications. An NVMe entity may terminate this TLS connection and re-establish it as required. An AVE may terminate this TLS connection after some period of inactivity (e.g., 10 minutes). An NVMe entity may avoid termination of this TLS connection by using the TLS heartbeat extension (refer to RFC 6520).

### 8.3.4.6.3 AVE Access Protocol

Communication with a DH-HMAC-CHAP AVE uses two PDUs, DH-HMAC-CHAP\_Access-Request and DH-HMAC-CHAP\_Access-Result, that are sent directly over the TLS connection with the AVE (refer to section 8.3.4.6.2). The format of the DH-HMAC-CHAP\_Access-Request PDU is shown in Figure 755.

**Figure 755: DH-HMAC-CHAP\_Access-Request PDU format**

Bytes	Description
00	<b>PDU Type (PDU-Type):</b> AEh for DH-HMAC-CHAP_Access-Request
01	<b>Flags (FLAGS):</b> Reserved
02	<b>Header Length (HLEN):</b> Fixed length of 8 bytes (08h)

Figure 755: DH-HMAC-CHAP\_Access-Request PDU format

Bytes	Description
03	<b>PDU Data Offset (PDO):</b> Reserved
07:04	<b>PDU Length (PLEN):</b> total length of the PDU in bytes
15:08	<b>Identifier (ID):</b> 64-bit identifier used to match Access-Request and Access-Result PDUs
16	<b>Hash Length (HL):</b> Length in bytes of the selected hash function
17	<b>Hash Identifier (HashID):</b> Identifier of selected hash function
19:18	<b>Transaction Identifier (T_ID):</b> 16-bit authentication transaction identifier
20	<b>Secure Channel Concatenation (SC_C):</b> Secure Channel concatenation indication
21	<b>Responder's Role (RESPR):</b> 'H' if host, 'C' if controller
22	<b>NQN Responder Length (NQNRIen):</b> Length of the responder's NQN
23	Reserved
27:24	<b>Sequence Number (SEQN):</b> Sequence number S
27+HL:28	<b>Augmented Challenge Value (ACV):</b> Challenge C <sub>a</sub>
27+2*HL:28+HL	<b>Response Value (RESPV):</b> Response R
NQNRIen+27+2*HL:28+2*HL	<b>NQN of Responder (NQNR):</b> Responder's NQN

The DH-HMAC-CHAP\_Access-Request PDU contains the parameters exchanged by the host and the controller during a DH-HMAC-CHAP authentication transaction. The responder is the entity that replied to a DH-HMAC-CHAP challenge sent by an authenticator.

Referring to Figure 754, when the controller transmits the DH-HMAC-CHAP\_Access-Request PDU, the parameters are instantiated as follows:

- Responder's Role: 'H'
- Sequence Number: S<sub>1</sub>
- Augmented Challenge Value: C<sub>a1</sub>
- Response Value: R<sub>1</sub>
- NQNR: NQN<sub>h</sub>
- HashID, T\_ID, SC\_C: the correspondent DH-HMAC-CHAP parameters

When the host transmits the DH-HMAC-CHAP\_Access-Request PDU, the parameters are instantiated as follows:

- Responder's Role: 'C'
- Sequence Number: S<sub>2</sub>
- Augmented Challenge Value: C<sub>a2</sub>
- Response Value: R<sub>2</sub>
- NQNR: NQN<sub>c</sub>
- HashID, T\_ID, SC\_C: the correspondent DH-HMAC-CHAP parameters

Upon receiving a DH-HMAC-CHAP\_Access-Request PDU, the AVE shall perform the following steps in order:

1. Lookup the DH-HMAC-CHAP key of the responder (i.e., K<sub>r</sub>) from NQNR;
2. If the Responder's Role is 'H', compute the expected response R' as:  
R' = HMAC(K<sub>r</sub>, C<sub>a</sub> || S || T\_ID || SC\_C || "HostHost" || NQN<sub>r</sub> || 00h || NQN<sub>a</sub>)  
where NQN<sub>a</sub> is the NQN of the authenticator;
3. If the Responder's Role is 'C', compute the expected response R' as:  
R' = HMAC(K<sub>r</sub>, C<sub>a</sub> || S || T\_ID || SC\_C || "Controller" || NQN<sub>r</sub> || 00h || NQN<sub>a</sub>)  
where NQN<sub>a</sub> is the NQN of the authenticator; and
4. Compare the expected response R' with the response value R received in the DH-HMAC-CHAP\_Access-Request PDU. If R' = R then the authentication is successful; if R' ≠ R then the authentication has failed.

The NQN of the authenticator (i.e., NQN<sub>a</sub>) is retrieved from the TLS identity associated to the TLS connection with the AVE (refer to section 8.3.4.6.2).

The result of an authentication verification is returned to the NVMe entity in a DH-HMAC-CHAP\_Access-Result PDU. The format of the DH-HMAC-CHAP\_Access-Result PDU is shown in Figure 756.

**Figure 756: DH-HMAC-CHAP\_Access-Result PDU format**

Bytes	Description										
00	<b>PDU Type (PDU-Type):</b> AFh for DH-HMAC-CHAP_Access-Result										
01	<b>Flags (FLAGS):</b> Reserved										
02	<b>Header Length (HLEN):</b> Fixed length of 8 bytes (08h)										
03	<b>PDU Data Offset (PDO):</b> Reserved										
07:04	<b>PDU Length (PLEN):</b> Fixed length of 20 bytes (14h)										
15:08	<b>Identifier (ID):</b> 64-bit identifier used to match Access-Request and Access-Result PDUs										
16	<b>Authentication Verification Result (AuthRes):</b> <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>01h</td> <td>Authentication Verification Successful</td> </tr> <tr> <td>02h</td> <td>Authentication Verification Failed</td> </tr> <tr> <td>All other values</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	01h	Authentication Verification Successful	02h	Authentication Verification Failed	All other values	Reserved		
		Value	Definition								
		01h	Authentication Verification Successful								
		02h	Authentication Verification Failed								
All other values	Reserved										
17	<b>Reason Code (RCODE):</b> Additional explanation when authentication verification failed <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>Value</th> <th>Definition</th> </tr> </thead> <tbody> <tr> <td>00h</td> <td>No additional explanation</td> </tr> <tr> <td>01h</td> <td>Authentication failure</td> </tr> <tr> <td>02h</td> <td>Selected hash function unusable</td> </tr> <tr> <td>All other values</td> <td>Reserved</td> </tr> </tbody> </table>	Value	Definition	00h	No additional explanation	01h	Authentication failure	02h	Selected hash function unusable	All other values	Reserved
		Value	Definition								
		00h	No additional explanation								
		01h	Authentication failure								
02h	Selected hash function unusable										
All other values	Reserved										
19:18	Reserved										

#### 8.3.4.6.4 AVE Discovery

The AVE transport addresses may be configured on an NVMe entity or may be discovered by interacting with a Discovery Controller (e.g., a CDC). An NVMe entity should randomly select any of the discovered AVE transport addresses to connect to the AVE. The AVE Discovery log page is defined to facilitate this discovery (refer to section 5.1.12.3.3).

## 9 Error Reporting and Recovery

### 9.1 Command and Queue Error Handling

In the case of serious error conditions, like Completion Queue Invalid, the operation of the associated Submission Queue or Completion Queue may be compromised. In this case, host software should delete the associated Completion Queue and/or Submission Queue. The delete of a Submission Queue aborts all outstanding commands, and deletion of either queue type releases resources associated with that queue. Host software should recreate the Completion Queue and/or Submission Queue to then continue with operation.

In the case of serious error conditions for Admin commands, the entire controller should be reset using a Controller Level Reset. The entire controller should also be reset if a completion is not received for the deletion of a Submission Queue or Completion Queue.

For most command errors, there is not an issue with the Submission Queue and/or Completion Queue itself. Thus, host software and the controller should continue to process commands. It is at the discretion of host software whether to retry the failed command; the Retry bit in the completion queue entry indicates whether a retry of the failed command may succeed.

### 9.2 Media and Data Error Handling

In the event that the requested operation could not be performed to the NVM media, the particular command is completed with a media error indicating the type of failure using the appropriate status code.

If a read error occurs during the processing of a command, (e.g., End-to-end Guard Check Error, Unrecovered Read Error), the controller may either stop the DMA transfer into the memory or transfer the erroneous data to the memory. The host shall ignore the data in the memory locations for commands that complete with such error conditions.

If a write error occurs during the processing of a command, (e.g., an internal error, End-to-end Guard Check Error, End-to-end Application Tag Check Error), the controller may either stop or complete the DMA transfer.

Additional I/O Command Set specific error handling is described within applicable I/O Command Set specifications.

### 9.3 Memory Error Handling

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

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

### 9.4 Internal Controller Error Handling

If a controller level failure (e.g., a DRAM failure) occurs during the processing of a command, then the controller should abort the command with a status code of Internal Error. Any data transfer associated with the command may be incomplete or incorrect, and therefore any transferred data should not be used for any purpose other than error reporting or diagnosis. The host may choose to re-submit the command or indicate an error to the higher-level software.

### 9.5 Controller Fatal Status Condition

If the controller has a serious error condition and is unable to communicate with host software via completion queue entries in the Admin Completion Queue or I/O Completion Queues, then the controller may set the Controller Fatal Status (CSTS.CFS) bit to '1' (refer to section 3.1.4.6). This indicates to host

software that a serious error condition has occurred. When this condition occurs, host software should attempt to reset and then re-initialize the controller.

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

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

If that Controller Reset does not clear the Controller Fatal Status condition, the host should initiate an NVM Subsystem Reset (refer to section 3.7.1), if supported.

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

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

## 9.6 Communication Loss Handling

If the host loses communication with a controller, then the host is unable to receive a completion (CQE) for any outstanding command that has been submitted to that controller (refer to section 3.4.5). If the host is able to use another controller to access the same NVM subsystem or re-establish communication with the original controller, then it is strongly recommended that any host use of that controller to recover from communication loss follow the procedures and requirements in this section in order to avoid possible corruption of user data and unintended changes to NVM subsystem state.

Host recovery from communication loss with a controller consists of three functional components:

- Host determination that communication with a controller has been lost is described in section 9.6.1.
- Host determination that no further processing of outstanding commands is possible on that controller is described in section 9.6.2.
- Host retry, if any, of outstanding commands after communication loss is described in section 9.6.3.

These functional components interact with each other. Host detection of communication loss is necessary before the host is able to determine when no further controller processing of outstanding commands is possible. Host retries of outstanding commands that modify user data or NVM subsystem state are able to corrupt user data or make unintended changes to NVM subsystem state unless the host determines that no further controller processing of the original commands is possible as described in section 9.6.2.

### 9.6.1 Host Communication Loss with a Controller

A host determines that communication has been lost with a controller if:

- the host detects a Keep Alive Timeout (refer to section 3.9);
- for message-based transports, the host or the controller terminates the NVMe Transport connection on which the command was sent (refer to section 3.3.2.4); or
- the host detects a transport connection loss using methods outside the scope of this specification (e.g., the transport notifies the host of a loss of communication either with the controller to which the command was submitted or with the queue on which the command was sent).

A controller may detect a loss of communication at a different time (e.g., later) than the host detects that loss of communication. As explained in section 9.6.2, additional time may be required for the controller to stop processing commands after the controller detects a loss of communication.

### 9.6.2 End of Controller Processing of Outstanding Commands

This section describes how a host determines that no further controller processing of an outstanding command is possible after a loss of communication happens. At the time when a host detects a communication loss with a controller, the outstanding commands, if any, are commands for which the host is unable to receive a CQE as a result of the communication loss (refer to section 3.4.5).

Some commands (e.g., Sanitize) initiate background operations. These background operations are able to continue after a host loss of communication with the controller that started the background operation. After such a loss of communication, additional measures (e.g., commands submitted to a different controller) are necessary for the host to track progress and completion of such a background operation.

A host that is unable to communicate with a controller should perform the following steps in order to determine that no further controller processing of outstanding commands is able to occur:

1. For message-based transports, terminate the association and the associated transport connections. This step is skipped for memory-based transports.
2. Wait for sufficient time to ensure that the controller has detected a loss of communication using at least one of the following:
  - a. If the controller uses Command Based Keep Alive (refer to section 3.9.3.1), wait at least until  $2 * KATT$  (refer to section 3.9) from the time the host submitted the most recent Keep Alive Command to the controller;
  - b. If the controller uses Traffic Based Keep Alive (refer to section 3.9.4.1), wait at least until  $3 * KATT$  from the time the host submitted the most recent command to the controller; or
  - c. Receive a transport-specific notification for determining that the controller has terminated an NVMe Transport connection or detected a loss of communication (e.g., a fabric notification or a PCIe surprise link down error notification for a PCIe link that directly connects a host to an NVM subsystem (e.g., an SSD)).
3. Wait for additional sufficient time to ensure that the controller has stopped processing commands using one of the following:
  - a. If the CQT field (refer to Figure 312) is non-zero, wait for the amount of time indicated in the CQT field to elapse; or
  - b. If the CQT field is cleared to 0h, wait for an implementation specific amount of time (e.g., 10 seconds). The host should allow this value to be administratively configured.

The specification of the times to wait to ensure that the controller has detected a Keep Alive Timeout described in this section (i.e.,  $2 * KATT$  and  $3 * KATT$ ) assumes that the transport delays any command by at most one KATT. Once the last command is fetched by the controller, the controller is required to detect a Keep Alive Timeout after at most a further  $1 * KATT$  for Command Based Keep Alive and at most  $2 * KATT$  for Traffic Based Keep Alive (refer to Figure 90). The sum of the two delays (i.e., the transport delay and the delay to detect the Keep Alive timeout) is  $2 * KATT$  for Command Based Keep Alive and  $3 * KATT$  for Traffic Based Keep Alive.

### 9.6.3 Command Retry After Communication Loss

If the host loses communication with a controller, then the host is unable to receive a completion (CQE) for any outstanding command (refer to section 3.4.5) that has been submitted to that controller. If the host is able to use another controller to access the same NVM subsystem or re-establish communication with the original controller, the host may be able to use that controller to recover from the communication loss by retrying outstanding commands. It is strongly recommended that any host retry of any outstanding commands after communication loss follow the procedures and requirements in this section in order to avoid possible corruption of user data and unintended changes to NVM subsystem state (e.g., Reservation state (refer to section 8.1.22)).

For command retry purposes, every outstanding command falls into one of three command retry categories, Unrestricted Retry, Delayed Retry, or State-Dependent Retry, based on whether the command is idempotent (refer to section 9.6.3.1), and whether the command modifies user data or NVM subsystem



state. Section 9.6.3.2 defines these categories and describes requirements and restrictions on retrying outstanding commands in each category.

### 9.6.3.1 Idempotent Commands

Controller processing of an idempotent command produces the same end state on the NVM subsystem and returns the same results if that command is resubmitted one or more times with no intervening commands. All commands tend to modify some ancillary state on the controller (e.g., tracking statistics); these ancillary changes to state do not prevent a command from being considered idempotent. The results of the command include the status code (excluding transient status codes or error conditions, e.g., due to a loss of communication), any data returned to the host and any NVM subsystem changes to user data or state (e.g., reservation state, feature contents).

For example, a read command addressed to a specific location (e.g., LBA) in a namespace is an idempotent command. The read command addressed to a valid location in a namespace returns the same data with a successful completion status code if that command is submitted repeatedly. Similarly, a write command addressed to a valid location in a namespace writes the same data to that location if submitted repeatedly. This command is also an idempotent command.

On the other hand, a Namespace Management command (refer to section 5.1.21) that creates a namespace is not idempotent (i.e., is a non-idempotent command), as repeating the Namespace Management command creates additional namespaces with different namespace identifiers. Similarly, a Reservation Register command that unregisters a host (refer to section 7.6) is also not idempotent because repeating the command attempts to unregister a host that is no longer registered and returns an error status code.

### 9.6.3.2 Command Retry Categories and Requirements

For command retry purposes, an outstanding command is in one of three categories:

- **Unrestricted Retry:** The outstanding command is an idempotent command (refer to section 9.6.3.1) that is able to be retried without restrictions because that outstanding command has no effect on user data or NVM subsystem state (e.g., an NVM Command Set Read command). Refer to section 9.6.3.2.1.
- **Delayed Retry:** The outstanding command is an idempotent command for which any retry and/or reporting of the result of that retry is required to be delayed until no further controller processing is possible of that outstanding command because the outstanding command modifies user data or NVM subsystem state (e.g., an NVM Command Set Write command, except as described in section 9.6.3.2.2). Refer to section 9.6.3.2.2.
- **State-Dependent Retry:** The outstanding command is not an idempotent command (e.g., a Namespace Management command that creates a namespace) or is an idempotent command that is able to affect behavior of other hosts. The procedures for recovery from such an outstanding command depend on the extent, if any, to which that outstanding command has been processed by the controller. Refer to section 9.6.3.2.3.

Sections 9.6.3.2.1, 9.6.3.2.2, and 9.6.3.2.3 define each command retry category and describe host requirements and restrictions that prevent retry of outstanding commands from corrupting user data or making unintended changes to NVM subsystem state.

#### 9.6.3.2.1 Unrestricted Retry Commands

An outstanding command is an Unrestricted Retry command if that command:

- a) is an idempotent command (refer to section 9.6.3.1); and
- b) does not change more than ancillary state in the NVM subsystem (e.g., statistics such as the value of the Data Units Read field in the SMART / Health Information log page (refer to Figure 206)).

For an Unrestricted Retry command:

- the Controller Capability Change (CCC) bit;
- the Namespace Inventory Change (NIC) bit;

- the Namespace Capability Change (NCC) bit; and
- the Logical Block Content Change (LBCC) bit

are all cleared to '0' in the Commands Supported and Effects data structure (refer to Figure 210) in the Commands Supported and Effects log page (refer to section 5.1.12.1.6). If any of these four bits is set to '1', then that command is not an Unrestricted Retry command.

A host may retry any outstanding command that is an Unrestricted Retry command immediately after communication loss without determining whether further controller processing of that outstanding command is possible.

For recovery purposes, a host may treat any outstanding command that is an Unrestricted Retry command as if that command were a Delayed Retry command or a State-Dependent Retry command.

#### 9.6.3.2.2 Delayed Retry Commands

An outstanding command is a Delayed Retry command if that command:

- is an idempotent command (refer to section 9.6.3.1); and
- changes user data or NVM subsystem state (e.g., Read Recovery Level (refer to section 8.1.20) or Reservation state (refer to section 8.1.22)),

unless the changes to user data or NVM subsystem state are able to affect the behavior of any other host (e.g., refer to the example in Annex B.6.4). As explained further in Annex B.6.4, use of individual Delayed Retry commands (e.g., an NVM Command Set Write command) that are not part of a fused operation to affect the behavior of other hosts is strongly discouraged.

A host should treat an outstanding command that is a Delayed Retry command as having command ordering requirements with respect to other commands where those command ordering requirements are enforced by higher-level software (refer to section 3.4.1). Hence, for any such command, a host should not report completion or error status, including errors caused by communication loss, to higher-level software (e.g., an associated application, filesystem or database) until the host has determined that no further controller processing of that command and retries, if any, of that command is possible. If a host violates this recommendation, corruption of user data and unintended changes to NVM subsystem state are possible; refer to Annex B.6.1 for an example where user data is corrupted.

A host is able to comply with this "should not report" recommendation by delaying submission of a retry of any outstanding command that is a Delayed Retry command until no further controller processing is possible of the original outstanding command and any previously submitted retries of that command. This avoids host delays in reporting completion of any command upon receiving the CQE for that command.

A host may treat any outstanding command that is a Delayed Retry command as if that command were a State-Dependent Retry command.

A host that does not adhere to the recommendations in this section for handling outstanding commands that are Delayed Retry commands risks causing corruption of user data. It is strongly recommended that host NVMe implementations adhere to these recommendations to avoid data corruption.

#### 9.6.3.2.3 State-Dependent Retry Commands

An outstanding command is a State-Dependent Retry command if the command changes user data or NVM subsystem state, and the command:

- is not an idempotent command (refer to section 9.6.3.1); or
- changes user data or NVM subsystem state in a way that is able to affect the behavior of other hosts.

A host should not retry an outstanding command that is a State-Dependent Retry command without first determining that command retry is the appropriate recovery action. This is because retrying such a command may have different results than the original command, duplicate the results of the original command, or affect the behavior of other hosts in a different manner than the original command. In general, determination of the appropriate recovery action is only able to be performed by higher-level software (e.g.,

an associated application, filesystem or database) that is able to determine the extent, if any, to which the outstanding command has been processed and enforce ordering requirements among commands (refer to section 3.4.1).

A host should treat an outstanding command that is a State-Dependent Retry command as having command ordering requirements enforced by higher-level software with respect to other commands (refer to section 3.4.1). Hence, for any such command, a host should not report completion or error status, including errors caused by communication loss, to higher-level software (e.g., an associated application, a filesystem or database), until the host has determined that no further controller processing of the outstanding command and retries, if any, of that command is possible. If a host violates this recommendation, corruption of user data or unintended changes to NVM subsystem state are possible; refer to Annex B.6.2 for an example where unintended changes occur to NVM subsystem state.

A host that does not adhere to the recommendations in this section for handling outstanding commands that are State-Dependent Retry commands risks causing corruption of user data and unintended changes to NVM subsystem state. It is strongly recommended that host NVMe implementations adhere to these recommendations to avoid these outcomes.

## Annex A. Sanitize Operation Considerations (Informative)

### A.1 Overview

The Sanitize command starts a sanitize operation that makes all user data previously written to the sanitization target inaccessible (i.e., all user data has been purged, as defined in IEEE Std 2883-2022). It is very difficult to prove that the sanitize operation successfully purged all user data. This annex provides some context and considerations for understanding the result of the operation and the practical limitations for auditing the result of the sanitize operation.

### A.2 Hidden Storage (Overprovisioning)

Sanitize operations purge all physical storage in the sanitization target that is able to hold user data. Many NVMe SSDs contain more physical storage than is addressable through the interface (e.g., overprovisioning). That physical storage is used for vendor specific purposes which may include providing increasing endurance, improving performance, and providing extra capacity to allow retiring bad or worn-out storage without affecting capacity. This excess capacity as well as any retired storage are not accessible through the interface. Vendor specific innovative use of this extra capacity supports advantages to the end user, but the lack of observability makes it difficult to ensure that all storage within the sanitization target was correctly purged. Only the accessible storage is able to be audited for the results of a sanitization operation.

### A.3 Integrity checks and No-Deallocate After Sanitize

Another issue is availability of the data returned through the interface. Some of the sanitize operations (e.g., Block Erase) affect the physical devices in such a way that directly reading the accessible storage may trigger internal integrity checks resulting in error responses instead of returning the contents of the storage. Other sanitize operations (e.g., Crypto Erase) may scramble the vendor specific internal format of the data, also resulting in error responses instead of returning the contents of the storage.

To compensate for these issues, a controller may perform additional internal write operations to media that is able to be allocated for user data (i.e., additional media modification, refer to section 8.1.24.1) so that all media that is allocated for user data is readable without error. However, this has the side effect of potentially significant additional wear on the media as well as the side effect of obscuring the results of the initial sanitize operation (i.e., the writes destroy the ability to forensically audit the result of the initial sanitize operation). Given this side effect, process audits of sanitize behavior only provide effective results when the No-Deallocate After Sanitize bit is set the same way (e.g., set to '1') for both process audits and the individual forensic device audits.

The Sanitize command introduced in NVM Express Base Specification, Revision 1.3 included a mechanism to specify that sanitized media allocated for user data not be deallocated, thereby allowing observations of the results of the sanitization operation. However, some architectures and products (e.g., integrity checking circuitry) interact with this capability in such a way as to defeat the sanitize result observability purpose. New features were added to NVM Express Base Specification, Revision 1.4 that include extended information about the sanitization capabilities of devices, a new asynchronous event, and configuration of the response to No-Deallocate After Sanitize requests. These features are intended to both support new systems that understand the new capabilities, as well to help manage legacy systems that do not understand the new capabilities without losing the ability to sanitize as requested.

These issues in returning the contents of accessible storage do not apply if the sanitization target is in the Media Verification state (refer to section 8.1.24.3.6). In that state, failure of internal integrity checks do not cause error responses to Read commands (refer to the Media Verification section of the NVM Command Set Specification). Because the Sanitize command that caused entry to the Media Verification state specified the Enter Media Verification State (EMVS) bit set to '1', the controller does not perform the additional media modification described in this section.

#### **A.4 Bad Media and Vendor Specific NAND Use**

Another audit capability that is not supported by NVM Express is checking that any media that could not be sanitized (e.g., bad physical blocks) has been removed from the pool of storage that is able to be used as addressable storage.

An approach that is performed under some circumstances is removing the physical storage components from the NVM Express device after a sanitize operation and reading the contents in laboratory conditions. However, this approach also has multiple difficulties. When physical storage components are removed from an NVM Express device, much context is lost. This includes:

- a) any encoding for zero's/one's balance;
- b) identification of which components contain device firmware or other non-data information; and
- c) which media has been retired and cannot be sanitized.

## Annex B. Host Considerations (Informative)

### B.1 Basic Steps when Building a Command

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

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

### B.2 Creating an I/O Submission Queue

This example describes how host software creates an I/O Submission Queue that utilizes non-contiguous PRP entries. Creating an I/O Submission Queue that utilizes a PRP List is only valid if the controller supports non-contiguous queues as indicated in CAP.CQR.

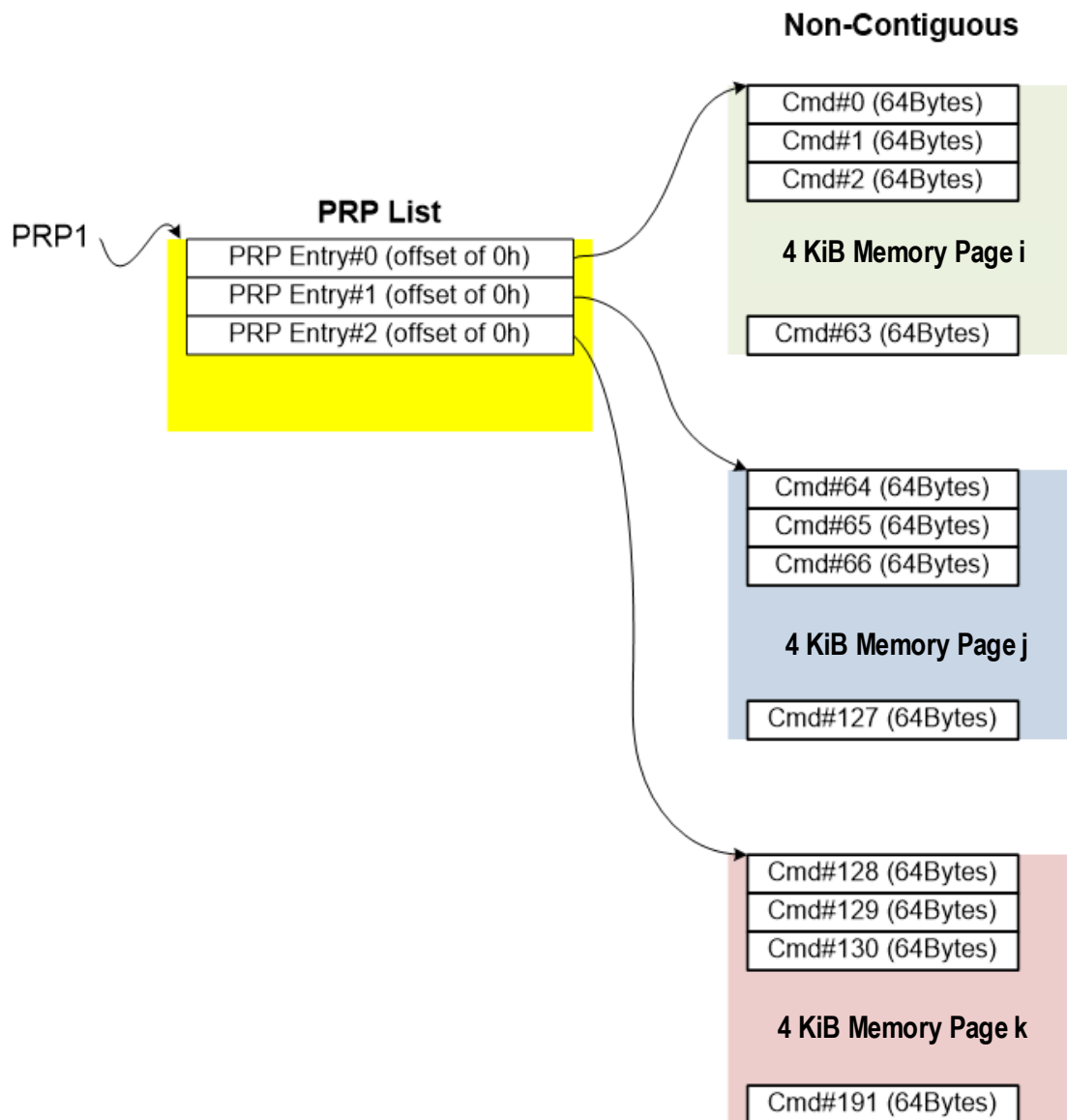
Prior to creating an I/O Submission Queue, host software shall create the I/O Completion Queue that the SQ uses with the Create I/O Completion Queue command.

To create an I/O Submission Queue, host software builds a Create I/O Submission Queue command for the Admin Submission Queue. Host software builds the Create I/O Submission Queue command in the next free Admin Submission Queue command location. The attributes of the command are:

- CDW0.OPC is set to 01h;
- CDW0.FUSE is cleared to 00b indicating that this is not a fused operation;
- CDW0.CID is set to a free command identifier;
- The NSID field is cleared to 0h; Submission Queues are not specific to a namespace;
- MPTR is cleared to 0h; metadata is not used for this command;
- PRP1 is set to the physical address of the PRP List. The PRP List is shown in Figure 757 for a PRP List with three entries;
- PRP2 is cleared to 0h; PRP Entry 2 is not used for this command;
- CDW10.QSIZE is set to the size of queue to create. In this case, the value is set to 191, indicating a queue size of 192 entries. The queue size shall not exceed the maximum queue entries supported, indicated in the CAP.MQES field;
- CDW10.QID is set to the Submission Queue identifier;
- CDW11.CQID is set to the I/O Completion Queue identifier where command completions are posted;
- CDW11.QPRIO is set to 10b, indicating a Medium priority queue; and
- CDW11.PC is cleared to '0' indicating that the data buffer indicated by PRP1 is not physically contiguously.

Host software then completes a transport specific action in order to submit the command for processing. Host software shall maintain the PRP List unmodified in host memory until the Submission Queue is deleted.

**Figure 757: PRP List Describing I/O Submission Queue**



### B.3 Executing a Fused Operation

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

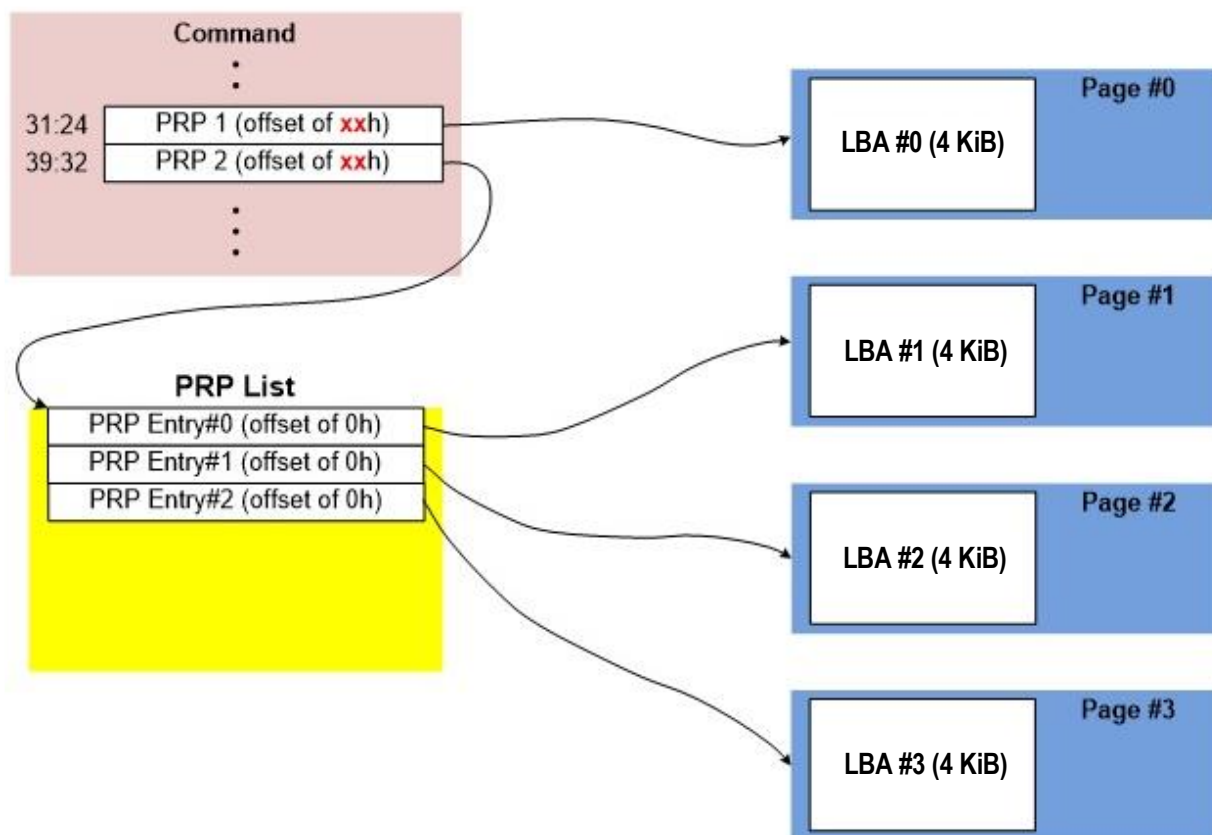
To build commands for a fused operation, host software utilizes two available adjacent command locations in the appropriate I/O Submission Queue as is described in section 3.4.2.

The attributes of the Compare command are:

- CMD0.CDW0.OPC is set to 05h for Compare;
- CMD0.CDW0.FUSE is set to 01b indicating that this is the first command of a fused operation;
- CMD0.CDW0.CID is set to a free command identifier;
- CMD0.NSID is set to identify the appropriate namespace;

- If metadata is being used in a separate buffer, then the location of that buffer is specified in the CMD0.MPTR field;
- The physical address of the first page of the data to compare:
  - If PRPs are used, CMD0.PRP1 is set to the physical address of the first page of the data to compare and CMD0.PRP2 is set to the physical address of the PRP List. The PRP List is shown in Figure 758 for a PRP List with three entries; or
  - If the command uses SGLs, CMD0.SGL1 is set to an appropriate SGL segment descriptor depending on whether more than one descriptor is needed;
- CMD0.CDW10.SLBA is set to the first LBA to compare against. Note that this field also spans Command Dword 11;
- CMD0.CDW12.LR is cleared to '0' to indicate that the controller should apply all available error recovery means to retrieve the data for comparison;
- CMD0.CDW12.FUA is cleared to '0', indicating that the data may be read from any location, including a volatile cache, in the NVM subsystem;
- CMD0.CDW12.PRINFO is cleared to 0h since end-to-end protection is not enabled;
- CMD0.CDW12.NLB is set to 3h, indicating that four logical blocks of a size of 4 KiB each are to be compared against;
- CMD0.CDW14 is cleared to 0h since end-to-end protection is not enabled; and
- CMD0.CDW15 is cleared to 0h since end-to-end protection is not enabled.

**Figure 758: PRP List Describing Data to Compare**



The attributes of the Write command are:

- CMD1.CDW0.OPC is set to 01h for Write;
- CMD1.CDW0.FUSE is set to 10b indicating that this is the second command of a fused operation;
- CMD1.CDW0.CID is set to a free command identifier;



- CMD1.NSID is set to identify the appropriate namespace. This value shall be the same as CMD0.NSID;
- If metadata is being used in a separate buffer, then the location of that buffer is specified in the CMD1.MPTR field;
- The physical address of the first page of data to write is identified:
  - If the command uses PRPs, then CMD1.PRP1 is set to the physical address of the first page of the data to write and CMD1.PRP2 is set to the physical address of the PRP List. The PRP List includes three entries; or
  - If the command uses SGLs, CMD1.SGL1 is set to an appropriate SGL segment descriptor depending on whether more than one descriptor is needed;
- CMD1.CDW10.SLBA is set to the first LBA to write. Note that this field also spans Command Dword 11. This value shall be the same as CMD0.CDW10.SLBA;
- CMD1.CDW12.LR is cleared to '0' to indicate that the controller should apply all available error recovery means to write the data to the NVM;
- CMD1.CDW12.FUA is cleared to '0', indicating that the data may be written to any location, including a volatile cache, in the NVM subsystem;
- CMD1.CDW12.PRINFO is cleared to 0h since end-to-end protection is not enabled;
- CMD1.CDW12.NLB is set to 3h, indicating that four logical blocks of a size of 4 KiB each are to be compared against. This value shall be the same as CMD0.CDW12.NLB;
- CMD1.CDW14 is cleared to 0h since end-to-end protection is not enabled; and
- CMD1.CDW15 is cleared to 0h since end-to-end protection is not enabled.

Host software then completes a transport specific action in order to submit the command for processing. Note that the transport specific submit action shall indicate both commands have been submitted at one time.

#### **B.4 Asynchronous Event Request Host Software Recommendations**

This section describes the recommended host software procedure for Asynchronous Event Requests.

The host sends  $n$  Asynchronous Event Request commands (refer to section 3.5.1, step 12). When an Asynchronous Event Request completes (providing Event Type, Event Information, and Log Page details):

- If the event(s) in the reported Log Page may be disabled with the Asynchronous Event Configuration feature (refer to section 5.1.25.1.5), then host software issues a Set Features command for the Asynchronous Event Configuration feature specifying to disable reporting of all events that utilize the Log Page reported. Host software should wait for the Set Features command to complete;
- Host software issues a Get Log Page command requesting the Log Page reported as part of the Asynchronous Event Command completion. Host software should wait for the Get Log Page command to complete;
- Host software parses the returned Log Page. If the condition is not persistent, then host software should re-enable all asynchronous events that utilize the Log Page. If the condition is persistent, then host software should re-enable all asynchronous events that utilize the Log Page except for the one(s) reported in the Log Page. The host re-enables events by issuing a Set Features command for the Asynchronous Event Configuration feature;
- Host software should issue an Asynchronous Event Request command to the controller (restoring to  $n$  the number of these commands outstanding); and
- If the reporting of event(s) was disabled, host software should enable reporting of the event(s) using the Asynchronous Event Configuration feature. If the condition reported may persist, host software should continue to monitor the event (e.g., spare below threshold) to determine if reporting of the event should be re-enabled.

## B.5 Updating Controller Doorbell Properties using a Shadow Doorbell Buffer

### B.5.1. Shadow Doorbell Buffer Overview

Controllers that support the Doorbell Buffer Config command are typically emulated controllers where this feature is used to enhance the performance of host software running in Virtual Machines. If supported by the controller, host software may enable Shadow Doorbell buffers by submitting the Doorbell Buffer Config command (refer to section 5.2.5).

After the completion of the Doorbell Buffer Config command, host software shall submit commands by updating the appropriate entry in the Shadow Doorbell buffer instead of updating the controller's corresponding doorbell property. If updating an entry in the Shadow Doorbell buffer changes the value from being less than or equal to the value of the corresponding EventIdx buffer entry to being greater than that value, then the host shall also update the controller's corresponding doorbell property to match the value of that entry in the Shadow Doorbell buffer. Queue wrap conditions shall be taken into account in all comparisons in this paragraph.

The controller may read from the Shadow Doorbell buffer and update the EventIdx buffer at any time (e.g., before the host writes to the controller's doorbell property).

### B.5.2. Example Algorithm for Controller Doorbell Property Updates

Host software may use modular arithmetic where the modulus is the queue depth to decide if the controller doorbell property should be updated, specifically:

- Compute  $X$  as the new doorbell value minus the corresponding EventIdx value, modulo queue depth; and
- Compute  $Y$  as the new doorbell value minus the old doorbell value in the shadow doorbell buffer, also modulo queue depth.

If  $X$  is less than or equal to  $Y$ , the controller doorbell property should be updated with the new doorbell value.

## B.6 Examples of Incorrect Command Retry Handling After Communication Loss

Section 9.6.3 describes requirements for host retry of outstanding commands after communication loss. In this situation, the response of a command is unknown and hence the host has no information about the extent, if any, to which the controller has processed that command. Many commands are not safe to unconditionally retry if they have been processed in part or completely. This annex describes examples of problematic situations caused by retrying an outstanding command without regard to the consequences of that retry.

### B.6.1. Write after Write

In the example shown in Figure 759, the host loses communication with Controller 1 and does not receive a response from Controller 1 for an idempotent command that changes user data at location  $X$  to  $A$  (e.g., an NVM Command Set Write command). The following events occur:

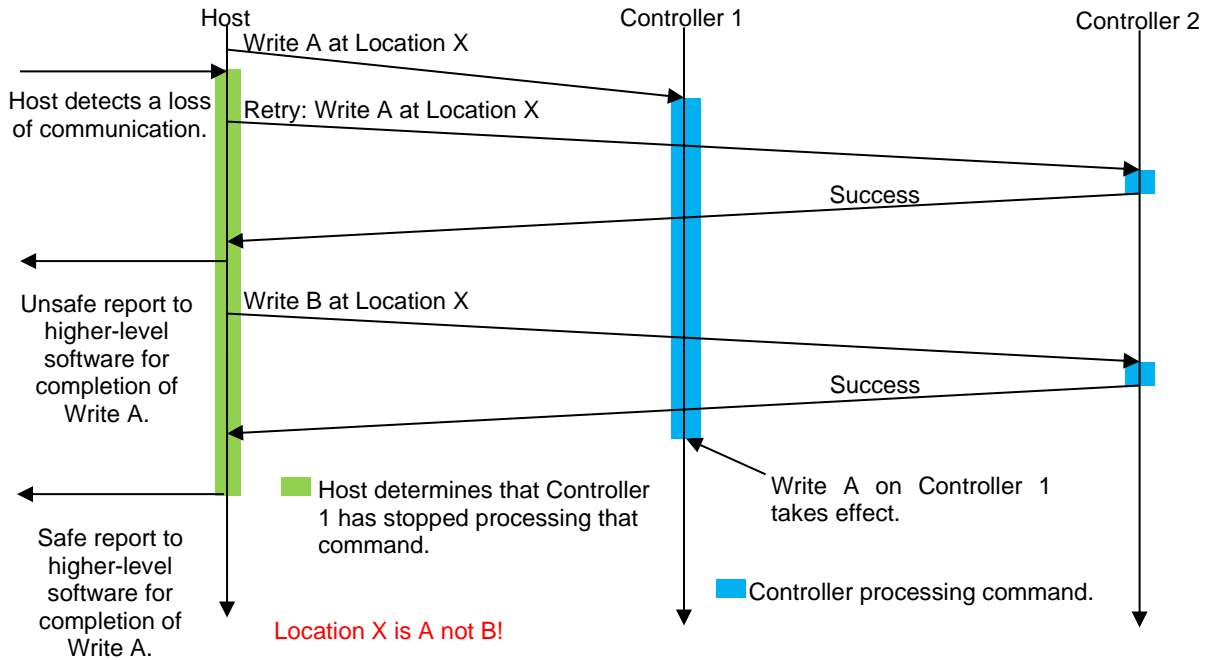
- The host retries that command on Controller 2 (Retry: Write  $A$  at Location  $X$ ), and the retry succeeds quickly.
- The completion of that retry leads to the host subsequently submitting a command that changes the user data at the same location to  $B$  (Write  $B$  at Location  $X$ ).
- During this time, Controller 1 has been processing the original outstanding command (Write  $A$  at Location  $X$ ), and that command's change of user data at location  $X$  to  $A$  finally takes effect after the user data at location  $X$  has already been changed to  $B$ .

The final outcome is that the user data at location  $X$  is  $A$ , which is incorrect and an example of data corruption.

For an idempotent command that changes user data or NVM subsystem state, this example shows why the host should not report the results of that command, including any retry of that command, to higher-level

software until the host is able to determine that no further controller processing of that command and any retry of that command is possible (refer to section 9.6.2).

**Figure 759: Write after Write**

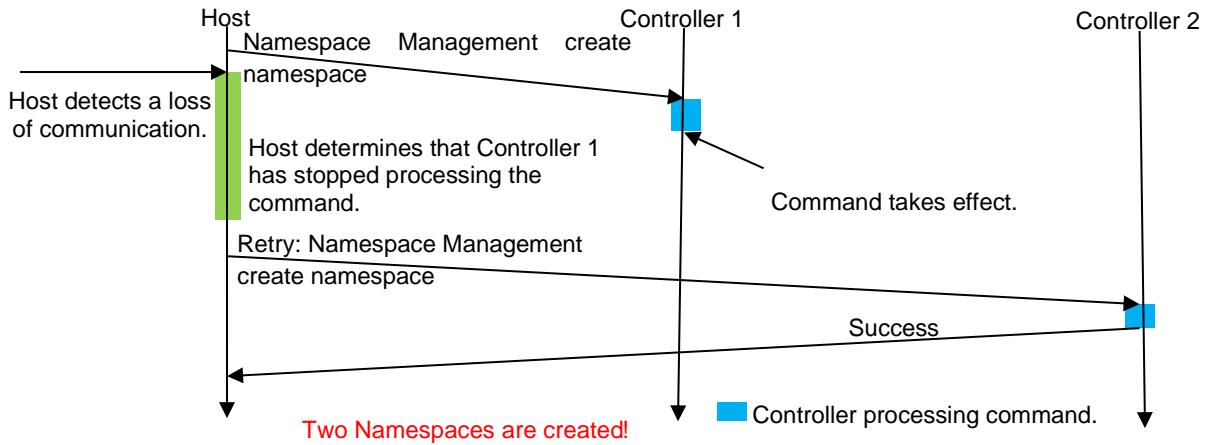


### B.6.2. Non-Idempotent Command

In the example shown in Figure 760, the host loses communication with Controller 1 and does not receive a response from Controller 1 for a Namespace Management command that creates a namespace (refer to section 5.1.21). The host ensures that no further controller processing of that command is possible (refer to section 9.6.2), and then retries that command on Controller 2, which creates a second namespace.

This example shows why higher-level software (e.g., an associated application, filesystem or database) should take steps to determine that a retry of a non-idempotent command does not cause unintended changes to NVM subsystem state (e.g., number of namespaces).

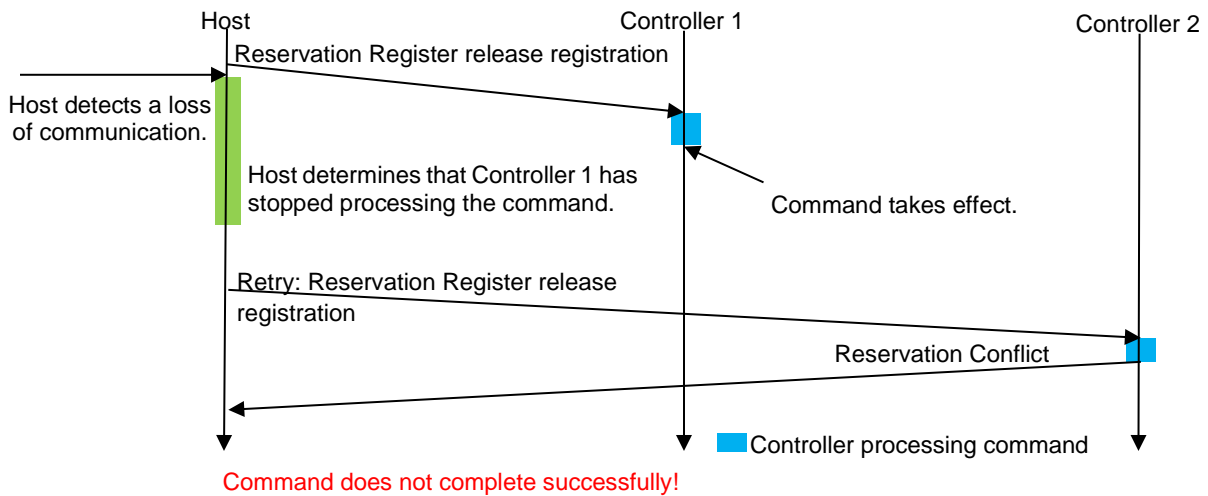
**Figure 760: Non-Idempotent Command**



**B.6.3. Retried Command Does Not Succeed**

In the example shown in Figure 761, the host loses communication with Controller 1 and does not receive a response to a Reservation Register command that unregisters the host (refer to section 7.6). The host ensures that no further controller processing of that command is possible (refer to section 9.6.2), and then retries that command on Controller 2. As a result of the original command unregistering the host, the host is no longer a registrant, and for that reason, the controller returns a status code of Reservation Conflict (refer to section 8.1.22.4).

**Figure 761: Retried Command Does Not Succeed**



This example shows why an error status code is able to be returned if a non-idempotent command is retried after the original command has been processed. An analogous example is possible for the Compare and Write fused operation (refer to the Fused Operation section of the NVM Command Set Specification) because that fused operation is not idempotent

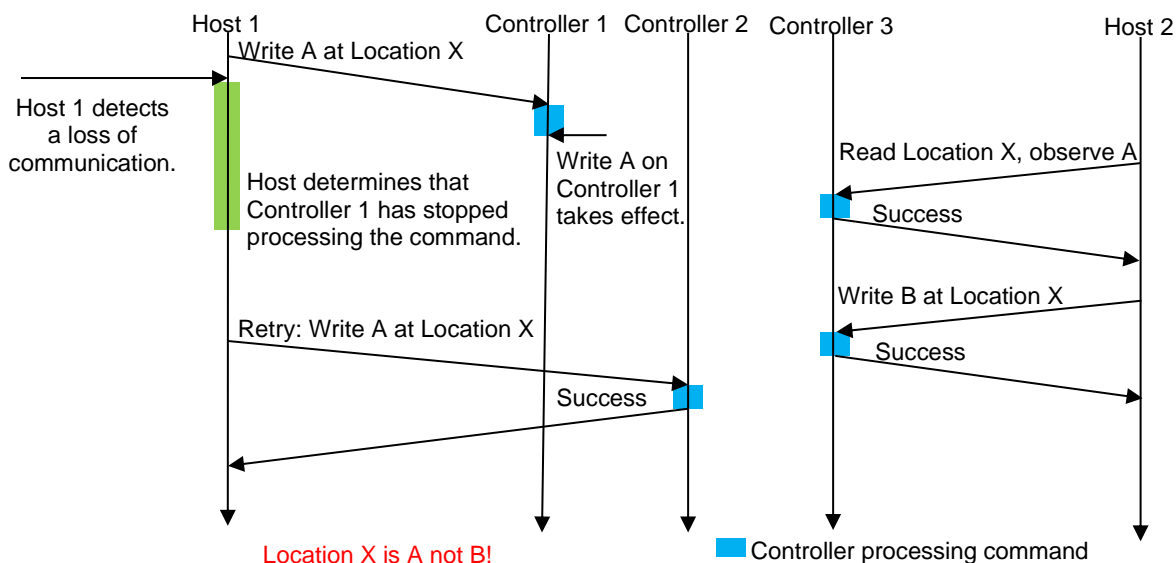
### B.6.4. Retried Command Affects Another Host

In the example shown in Figure 762, two hosts use Location X in a namespace for coordination. Writing the value A to Location X indicates that step A in a processing sequence has been completed, and writing the value B indicates that step B in the processing sequence has been completed, where higher-level software requires that step B follow step A.

Host 1 indicates completion of step A by writing the value A to Location X, but loss of communication prevents Host 1 from receiving the completion of that command. Host 2 observes that step A is complete, quickly performs step B, and indicates completion of step B by writing the value B to Location X.

In the absence of receiving a completion for the original command, Host 1 retries writing the value A to Location X, overwriting the completion of step B reported by Host 2. This example shows that retry of commands that are able to affect the behavior of other hosts is problematic. In this example, higher-level software needs a mechanism to indicate that the writes to Location X are not safe to retry after a delay.

**Figure 762: Retried Command Affects Another Host**



This sort of higher-level software usage of ordinary NVMe commands (e.g., NVM Command Set Write commands) for coordination and synchronization among multiple hosts is strongly discouraged because retry of these commands after communication loss is problematic. Higher-level software should instead use mechanisms intended for coordination among multiple hosts. Two examples of such mechanisms are:

- Reservations (refer to section 8.1.22); and
- Compare and Write fused operations (refer to the Fused Operation section of the NVM Command Set Specification).

In addition, command retries that modify NVM subsystem state (e.g., a Set Features command that modifies a feature that has any scope that is visible to other hosts as described in Figure 385) are able to affect the behavior of other hosts. Use of commands that modify NVM subsystem state for coordination and synchronization among multiple hosts is likewise strongly discouraged.

## Annex C. Power Management and Consumption (Informative)

NVM Express power management capabilities allow the host to manage power for a controller. Power management includes both control and reporting mechanisms.

For information on transport power management (e.g., PCIe, RDMA), refer to the applicable NVM Express transport specification.

The scope of NVM Express power management is the controller (refer to section 5.1.25.1.2).

NVM Express power management uses the following functionality:

- a) Features:
  - Power Management (refer to section 5.1.25.1.2 and section 8.1.17);
  - Autonomous Power State Transition (refer to section 5.1.25.1.6 and section 8.1.17.2);
  - Non-Operational Power State Configuration (refer to section 5.1.25.1.10 and section 8.1.17.1); and
  - Spinup Control (refer to section 5.1.25.1.18);
- b) NVM subsystem workloads (refer to 8.1.17.3); and
- c) Runtime D3 transitions (refer to section 8.1.17.4).

Controller thermal management may cause a transition to a lower power state, interacting with these Features:

- a) Temperature Threshold (refer to section 5.1.25.1.3);
- b) Host Controlled Thermal Management (refer to section 5.1.25.1.9 and section 8.1.17.5); and
- c) access to host memory buffer (refer to section 5.1.25.2.4) may be prohibited in non-operational power state.

NVM Express power management uses these reporting mechanisms:

- a) properties:
  - Controller Power Scope (CAP.CPS) (refer to Figure 36);
- b) fields in the Identify Controller data structure (refer to Figure 312):
  - RTD3 Resume Latency (RTD3R);
  - RTD3 Entry Latency (RTD3E);
  - Non-Operational Power State Permissive Mode;
  - Number of Power States Support (NPSS);
  - Autonomous Power State Transition Attributes (APSTA); and
  - Power State 0 Descriptor (PSD0) through Power State 31 Descriptor (PSD31) (refer to Figure 313);
- c) Features:
  - Power Management (refer to section 5.1.25.1.2);
  - Temperature Threshold (refer to section 5.1.25.1.3);
  - Autonomous Power State Transition (refer to section 5.1.25.1.6 and section 8.1.17.2);
  - Non-Operational Power State Configuration (refer to section 5.1.25.1.10 and section 8.1.17.1);
  - Host Controlled Thermal Management (refer to section 5.1.25.1.9 and section 8.1.17.5);
  - Host Memory Buffer (refer to section 5.1.25.2.4); and
  - Spinup Control (refer to section 5.1.25.1.18);

and
- d) log pages:
  - SMART / Health Information log page fields (refer to section 5.1.12.1.3):
    - Thermal Management Temperature [1-2] Transition Count; and

- Total Time For Thermal Management Temperature [1-2];
- and
- Persistent Event Log fields (refer to section 5.1.12.1.14):
    - Power On Hours (POH) (refer to Figure 226);
    - Power Cycle Count (refer to Figure 226);
    - Controller Power Cycle (refer to Figure 234); and
    - Power on milliseconds (refer to Figure 234).