LEGAL NOTICE:

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

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

NOTICE TO NON-MEMBERS OF NVM EXPRESS, INC.: If you are not a Member of NVM Express, Inc. and you have obtained a copy of this document, you only have a right to review this document or make reference to or cite this document. Any such references or citations to this document must acknowledge NVM Express, Inc. copyright ownership of this document. The proper copyright citation or reference is as follows: **"© 2007 - 2016 NVM Express, Inc. ALL RIGHTS RESERVED."** When making any such citations or references to this document you are not permitted to revise, alter, modify, make any derivatives of, or otherwise amend the referenced portion of this document in any way without the prior express written permission of NVM Express, Inc. Nothing contained in this document shall be deemed as granting you any kind of license to implement or use this document or the specification described therein, or any of its contents, either expressly or impliedly, or to any intellectual property owned or controlled by NVM Express, Inc., including, without limitation, any trademarks of NVM Express, Inc.

LEGAL DISCLAIMER:

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

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

NVM Express Workgroup
c/o Virtual, Inc.
401 Edgewater Place, Suite 600
Wakefield, MA 01880
info@nvmexpress.org

NVM Express™ Technical Errata

Errata ID	**003**
Revision Date	**10/16/2017**
Affected Spec Ver.	**NVM Express™ 1.3**
Corrected Spec Ver.	

Errata Author(s)

Name	Company
Judy Brock, Bill Martin	Samsung
Fred Knight	NetApp Inc.
Michael Allison	SK Hynix Inc
James Smart	Broadcom
Anthony Constantine	Intel
Christoph Hellwig, Yoni Shternhell	WDC
David Peterson	Brocade
Tom Friend	Toshiba
Paul Suhler	Micron Technology
David Black	Dell EMC

Errata Overview

Clarifications on Write Zeroes
Clarifications on Abort command
Clarifications on Changed Namespace List Log Page
Clarifications on Virtualization
Clarify requirement for shared namespace ID types and uniqueness when NS Management is not supported
Clarify the error returned when a Reservation Report command conflicts with current NS state
Incorporate changes from 1.2.1 ECN that were not incorporated into 1.3
Clarify host handling of unrecognized identifier descriptor types in the Identify command
Correct some references that were pointing to the wrong place
Move a “shall” requirement from the intro into the main body (section 5.15)
Clarification on PRP error case
Clarification on exclusive Stream Resources
Clarification on the additive nature of Telemetry area 1/2/3

Revision History

Revision Date	Change Description
06/09/2017	Incorporate clarifications for Write Zeroes.
06/14/2017	Incorporate clarifications for Abort.
06/20/2017	Incorporate clarifications for Changed Namespace List and Virtualization
06/21/2017	Address Reservation Report conflict.
07/18/2017	Incorporate Fabric errors, HMB updates, Stream clarifications, Telemetry, and NS ID type clarifications.
07/27/2017	Add NVMe-oF transport error alternatives, Log Page Scope clarification, and resolve ABORT command notes.
08/01/2017	Remove NVMe-oF transport errors to their own TP.
08/25/2017	Further clarifications via e-mail and con-call.
08/31/2017	Move HMB clarification to ECN-004
10/05/2017	Move LOG scope clarification to ECN-004
10/16/2017	Ratified

Incompatible Changes

Reservation Report commands that conflict with the existing Host ID state now report a specific error, the specification had been silent as to the handling of this conflicting situation.

In the case of an Abort command that did not successfully abort the specified command, the specification described the contents of the CQE, but was vague about requirements to complete the Abort command. This text has been clarified to include an explicit requirement to complete the Abort command with the described CQE contents.

Description of Specification Changes

Modify a portion of sections 1.1 (Overview) and 1.1.1 (NVMe over PCIe and NVMe over Fabrics) as shown below:

1.1 Overview

…

For an overview of changes from revision 1.2.1 to revision 1.3, refer to http://nvmexpress.org/changes for a document that describes the new features, including mandatory requirements for a controller to comply with revision 1.3.

1.1.1 NVMe over PCIe and NVMe over Fabrics

…

In this specification, a requirement/feature may be documented as specific to NVMe over Fabrics or to a particular NVMe Transport binding. In addition, support requirements for features and functionality may differ between NVMe over PCIe and NVMe over Fabrics.

~~To comply with NVM Express 1.2.1, a controller shall support the NVM Subsystem NVMe Qualified Name in the Identify Controller data structure in Figure 109.~~

Modify a portion of section 4.3 (Physical Region Page Entry and List) as shown below (changes to

Figure 14 are based on changes published in ECN-001):

4.3 Physical Region Page Entry and List

…

Figure 1: PRP Entry – Page Base Address and Offset

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

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

5.1 Abort command

The Abort command is used to abort a specific command previously submitted to the Admin Submission Queue or an I/O Submission Queue. An Abort command is a best effort command; the command to abort may have already completed, currently be in execution, or may be deeply queued. ~~It is implementation specific when a controller chooses to complete the Abort command when the command to abort is not found.~~

…

5.1.1 Command Completion

~~A completion queue entry is posted to the Admin Completion Queue if the command has been completed and a corresponding completion queue entry has been posted to the appropriate Admin or I/O Completion Queue. Dword 0 of the completion queue entry indicates whether the command was aborted. If the command was successfully aborted, then bit 0 of Dword 0 is cleared to '0'. If the command was not aborted, then bit 0 of Dword 0 is set to '1'.~~

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 and indicating whether the command to abort was aborted. Dword 0 of the completion queue entry indicates whether the command to abort was aborted.

If the command to abort was successfully aborted, then a completion queue entry for the aborted command shall be posted to the appropriate Admin or I/O Completion Queue with a status of Command Abort Requested before the completion queue entry for the Abort command is posted to the Admin Completion Queue, and bit 0 of Dword 0 shall be cleared to '0' in the completion queue entry for the Abort command. If the command to abort was not aborted for any reason, then bit 0 of Dword 0 shall be set to '1' in the completion queue entry for the Abort command.

Command specific status values associated with the Abort command are defined in Figure 44.

Modify Section 5.14.2 (Get Log Page command -> Command Completion) as shown below:

5.14 Get Log Page command

…

5.14.2 Command Completion

A completion queue entry is posted to the Admin Completion Queue if the log has been transferred to the memory buffer indicated in PRP Entry 1. Get Log Page command specific status values are defined in Figure 105.

Figure 2: Get Log Page – Command Specific Status Values

Value	Description
9h	**Invalid Log Page:** 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 versions 1.3 and earlier of the specification may return Invalid Field in Command for this condition.

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

5.15 Identify command

…

The Identify Controller data structure and Identify Namespace data structure include several identifiers. The format and layout of these identifiers is described in section 7.10.

Figure 3: Identify – Data Structure Returned

CNS Value	O/M	Definition
03h	M	A list of Namespace Identification Descriptor structures (refer to Figure 116) is returned to the host for the namespace specified in the Namespace Identifier (CDW1.NSID) field if it is an active NSID. The controller may return any number of variable length Namespace Identification Descriptor structures that fit into the 4096 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. If, while processing these descriptors, the host~~s sees an unknown~~ detects a descriptor type, that it does not recognize, then it should skip the unrecognized descriptor type and continue parsing the structure. A controller shall not return multiple descriptors with the same Namespace Identification Descriptor Type (NIDT). A controller shall return at least one descriptor identifying the namespace.

Figure 4: Identify – Identify Controller Data Structure

Bytes	O/M	Description
…		.
767:540		Reserved
1023:768	M	**NVM Subsystem NVMe Qualified Name (SUBNQN):** This field specifies the NVM Subsystem NVMe Qualified Name as a UTF-8 null-terminated string. Refer to section 7.9 for the definition of NVMe Qualified Name. Support for this field is mandatory if the controller supports revision 1.2.1 or later as indicated in the Version register (refer to section 3.1.2).

Bytes	O/M	Description
1791:1024		Reserved

Modify a portion of section 5.22 (Virtualization Management command) as shown below:

5.22 Virtualization Management command

The Virtualization Management command is supported by primary controllers that support the Virtualization Enhancements capability.

…

Figure 5: Virtualization Management – Command Dword 10

Bit	Description
…	
03:00	**Action (ACT):** This field indicates the operation for the command to perform as described below. **Value** / **Description** 0h – Reserved 1h – **Primary Controller Flexible Allocation:** Set the number of Flexible Resources allocated to this primary controller following the next Controller Level Reset other than a Controller Reset (i.e., CC.EN transitions from '1' to '0'). If the Controller Identifier field does not correspond to this primary controller then an error of Invalid Controller Identifier is returned. This value is persistent across power cycles and resets. 2h – 6h – Reserved

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

6.1 Namespaces

6.1.1 Namespace Overview

A namespace is a collection of logical blocks ~~that~~ whose logical block addresses range from 0 to the capacity of the namespace – 1. A namespace ID (NSID) is an identifier used by a controller to provide access to a namespace.

6.1.2 Valid and Invalid NSIDs

Valid NSIDs are the range of possible NSIDs that ~~correspond to a namespace that~~ may be used to refer to namespaces that exist in the NVM subsystem. Any NSID is valid, except if it is zero or greater than the Number of Namespaces field reported in the Identify Controller data structure (refer to Figure 109). 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 ~~or~~ 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.

~~Active NSIDs are valid NSIDs that are attached to the specific controller. Valid NSIDs that are not attached to the specific controller are called inactive. An active NSID becomes inactive when the associated namespace is detached from the specific controller or is deleted.~~

~~Allocated NSIDs are valid NSIDs that refer to namespaces that currently exist within an NVM subsystem. An allocated NSID may not be attached to any controller. An allocated NSID shall be attached to a controller before host software may submit I/O commands for that namespace on that controller. An allocated NSID becomes unallocated when the associated namespace is deleted.~~

6.1.3 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.

6.1.4 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 8.12 for actions associated with a namespace being detached or deleted.

6.1.5 NSID and Namespace Relationships

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

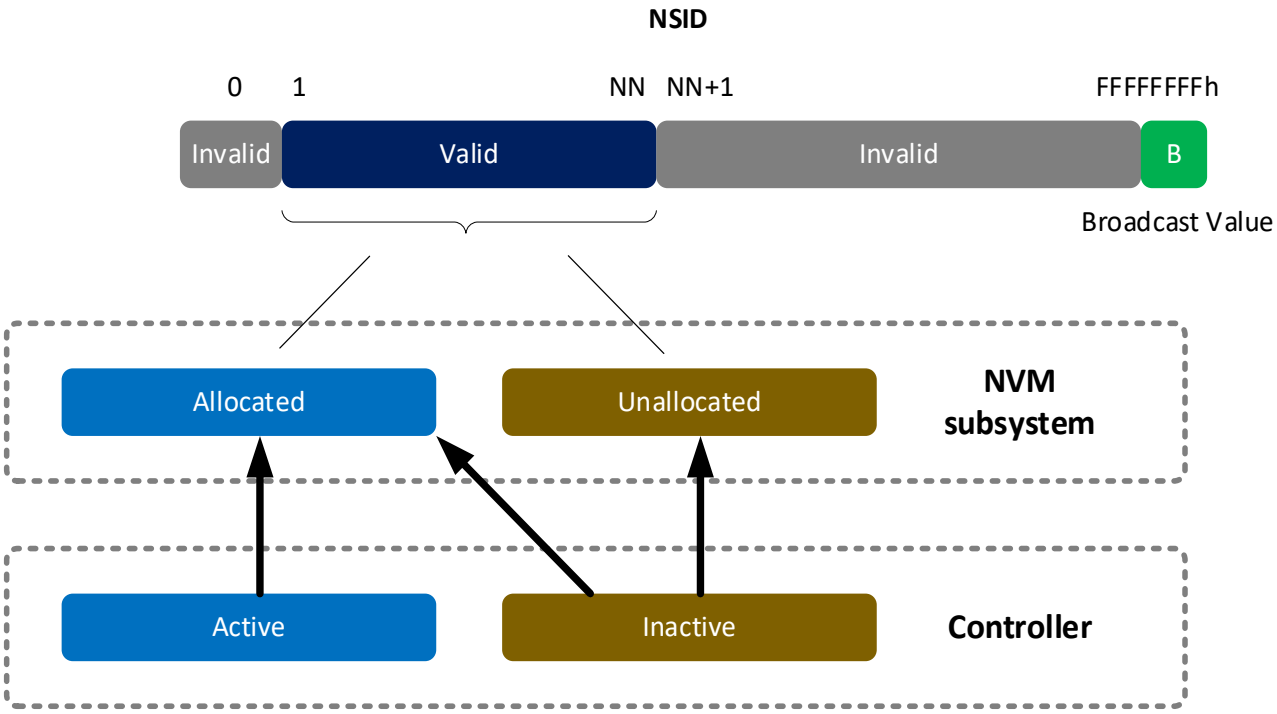
The following table summarizes the valid NSID types and Figure 189 visually shows the NSID types and how they relate.

~~Valid NSID Type~~	~~The associated namespace~~
~~Active~~	~~is attached to this controller~~
~~Inactive~~	~~is not attached to this controller~~
~~Allocated~~	~~exists in the NVM subsystem~~
~~Unallocated~~	~~does not exist in the NVM subsystem~~

Valid NSID Type	NSID relationship to namespace	Reference
Unallocated	Does not refer to any namespace that exists in the NVM subsystem	6.1.2
Allocated	Refers to a namespace that exists in the NVM subsystem	6.1.2
Inactive	Does not refer to a namespace that is attached to this controller [1]	6.1.3

Valid NSID Type	NSID relationship to namespace	Reference
Active	Refers to a namespace that is attached to this controller	6.1.3
NOTES: 1. If allocated, refers to a namespace that is not attached to this controller. If unallocated, does not refer to any namespace.		

Figure 6: NSID Types



6.1.6 NSID and Namespace Usage

If Namespace Management is supported (refer to the OACS field in Figure 109) then ~~Namespace~~ NSIDs shall be unique within the NVM subsystem (e.g., ~~namespace~~ NSID of 3 shall refer to the same physical namespace regardless of the accessing controller). If Namespace Management is not supported then ~~Namespace~~ NSIDs:

a) for shared namespaces shall be unique; and
b) for private namespaces are not required to be unique.

The Identify command (refer to section 5.15) may be used to determine the active NSIDs for a controller and the allocated NSIDs in the NVM subsystem.

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 set to 00h for each valid NSID (based on the Number of Namespaces value in Identify Controller). If a non-zero data structure is returned for a particular NSID, then that is an active NSID.
2. Issue an Identify command with a CNS field set to 02h to retrieve a list of up to 1024 active NSIDs. If there are more than 1024 active NSIDs, continue to issue Identify commands with a CNS field set to 02h until all active NSIDs are retrieved.

To determine the allocated NSIDs in the NVM subsystem, the host may ~~Issue~~ issue an Identify command with the CNS field set to 10h to retrieve a list of up to 1024 allocated NSIDs. If there are more than 1024

allocated NSIDs, continue to issue Identify commands with a CNS field set to 10h until all allocated NSIDs are retrieved.

Modify a portion of section 6.13 (Reservation Report command) as shown below:

6.13 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.

The size of the Reservation Status data structure is a function of the number of controllers in the NVM subsystem that are associated with hosts that are registrants of the namespace (i.e., there is a Registered Controller data structure and/or Registered Controller extended data structure for each such controller). The controller returns the data structure in Figure 231 if the host has selected a 64-bit Host Identifier and the data structure in Figure 232 if the host has selected a 128-bit Host Identifier (refer to section 5.21.1.19).

If a 64-bit Host Identifier has been specified and the Extended Data Structure field 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 field 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 7: Reservation Report – Command Dword 11

Bit	Description
31:01	Reserved
00	**Extended Data Structure (EDS):** If set to '1' then the controller returns the extended data structure defined in Figure 232. If cleared to '0' then the controller returns the data structure defined in Figure 231.

Modify a portion of section 6.16 (Write Zeroes command) as shown below:

6.16 Write Zeroes command

The Write Zeroes command is used to set a range of logical blocks to zero. ~~After successful completion of this command, the value returned by subsequent reads of logical blocks in this range shall be all bytes set to 00h until a write occurs to this LBA range. The~~ Non-PI related metadata for this command, if any, shall be all bytes set to 00h. The ~~and the~~ protection information for logical blocks written to the media is updated based on CDW12.PRINFO. If the Protection Information Action field (PRACT) is cleared to '0', then the protection information ~~metadata~~ for this command shall be all zeroes. If the Protection Information Action field (PRACT) is set to '1', then the protection information shall be based on the End-to-end Data Protection Type Settings (DPS) field in the Identify Namespace data structure (refer to Figure 114) and the CDW15.EILBRT, CDW15.ELBATM, and CDW15.ELBAT fields in the Write Zeroes command ~~any non-PI related metadata, if it exists, shall be all bytes set to 00h~~.

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

If the Deallocate bit (CDW12.DEAC) is set to '1' in a Write Zeroes command, and …

Modify a portion of section 7.5.1 (Pin Based, Single MSI, and Multiple MSI Behavior) as shown below:

7.5.1 Pin Based, Single MSI, and Multiple MSI Behavior

This is the mode of interrupt operation if any of the following conditions are met:

- Pin based interrupts are being used – MSI (MSICAP.MC.MSIE='0') and MSI-X are disabled
- Single MSI is being used – MSI is enabled (MSICAP.MC.MSIE='1'), MSICAP.MC.MME=0h, and MSI-X is disabled
- Multiple MSI is being used – Multiple-message MSI is enabled (MSICAP.MC.MSIE='1'), MSICAP.MC.MME is set to a value between 001b and 101b inclusive, and ~~(MSICAP.MC.MME=1h) and~~ MSI-X is disabled.

Modify a portion of Figure 254 in section 7.11 (Unique Identifier) as shown below:

7.11 Unique Identifier

…

Figure 8: NQN Construction for Older NVM Subsystems

Bytes	Description
26:00	**NQN Starting String (NSS):** Contains the 27 letter ASCII string ~~"nqn.2014.08.org.nvmexpress:""~~nqn.2014-08.org.nvmexpress:".
…	

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

8.4 Power Management

…

Associated with each power state is a Power State Descriptor in the Identify Controller data structure (refer to Figure 113). The descriptors for all implemented power states may be viewed as forming a table as shown in Figure 262 for a controller with seven implemented power states. Note that Figure 262 is illustrative and does not include all fields in the power state descriptor. The Maximum Power (MP) field indicates the maximum power that may be consumed in that state. Refer to the appropriate form factor specification for power measurement methodologies for that form factor. The controller may employ autonomous power management techniques to reduce power consumption below this level, but under no circumstances is power allowed to exceed this level except for non-operational power states as described in section 8.4.1.

Modify a portion of section 8.5 (Virtualization Enhancements) as shown below:

8.5 Virtualization Enhancements (Optional)

…

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 (i.e., CC.EN transitions from '1' to '0'). A secondary controller only supports having Flexible Resources assigned or removed when it is in the Offline state.

…

For each controller resource type supported, the Primary Controller Capabilities Structure (refer to Figure 110) 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 allocated to the primary controller using the Virtualization Management command;~~
- 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.

Modify a portion of section 8.10.4 (Authenticated Device Configuration Block Read) as shown below:

8.10.4 Authenticated Device Configuration Block Read

…

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 278~~ Figure 279.

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

8.12 Namespace Management (Optional)

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. Namespace management is intended for use during manufacturing or by a system administrator.

If Namespace Management is supported, then the controller should support the Namespace Attribute Changed asynchronous event (refer to Figure 49 and section 5.21.1.11).

If ~~When~~ a namespace is detached from a controller, ~~or deleted it~~ then the NSID that referred to that namespace becomes an inactive ~~namespace~~ NSID (refer to section 6.1.4) 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 6.1.3) in the NVM subsystem. Previously submitted but uncompleted or subsequently submitted commands to the affected ~~namespace~~ NSID are handled by the controller as if they were issued to an inactive ~~namespace~~ NSID (refer to Figure 11).

The size of a namespace is based on the number of logical blocks requested in a create operation, the format of the namespace, and any characteristics (e.g., endurance).

…

Modify a portion of section 8.14 (Telemetry) as shown below:

8.14 Telemetry (Optional)

...

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 ~~in order~~ to identify the source of the problem. For rare issues that are not root caused by a small or medium sized telemetry data collection, a third phase may be employed to collect the largest and most complete payload to diagnose the issue.

~~The telemetry data is returned in Telemetry Data Blocks. Each Telemetry Data Block is 512 bytes in size. A set of Telemetry Data Blocks forms a Telemetry Data Area. Each Telemetry Data Area represents the controller's internal state when the telemetry data was captured. There are three Telemetry Data Areas defined in the host-initiated and controller-initiated logs, all starting at Telemetry Data Block 1.~~

There are two telemetry data logs (i.e., Host-Initiated log and Controller-Initiated log) 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 5.14.1.7 and section 5.14.1.8) in units of Telemetry Data Blocks. Each telemetry data log is segmented into three Telemetry Data Areas (i.e., small, medium, and large). 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 A~~a~~rea 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). The size of each Telemetry Data A~~a~~rea is vendor specific and may change on each data collection. When possible, the host should retrieve the payload for all three 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 prepares ~~host memory buffer(s)~~ an area to store telemetry data if needed.
3. 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 5.21.1.11).
4. 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 ~~in~~ from the host-initiated log (refer to section 5.14.1.7) or the controller-initiated log (refer to section 5.14.1.8). If possible, the host should collect Telemetry Data Area 1, 2, and 3. The host reads the log in 512 byte Telemetry Data Block units. As part of the last read for a controller-initiated log, the host clears the Retain Asynchronous Event bit to '0'.
    b. If it is a controller-initiated log, the host re-reads the header of the log page and ensures that the Telemetry Controller-Initiated Data Generation Number matches the original value read. If it does not match, then the data captured is not consistent and needs to be re-read.
    c. When all telemetry data has been saved, the data should be forwarded to the manufacturer of the controller.

Modify a portion of section 8.14.1 (Telemetry Data Collection Examples) as shown below:

8.14.1 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:

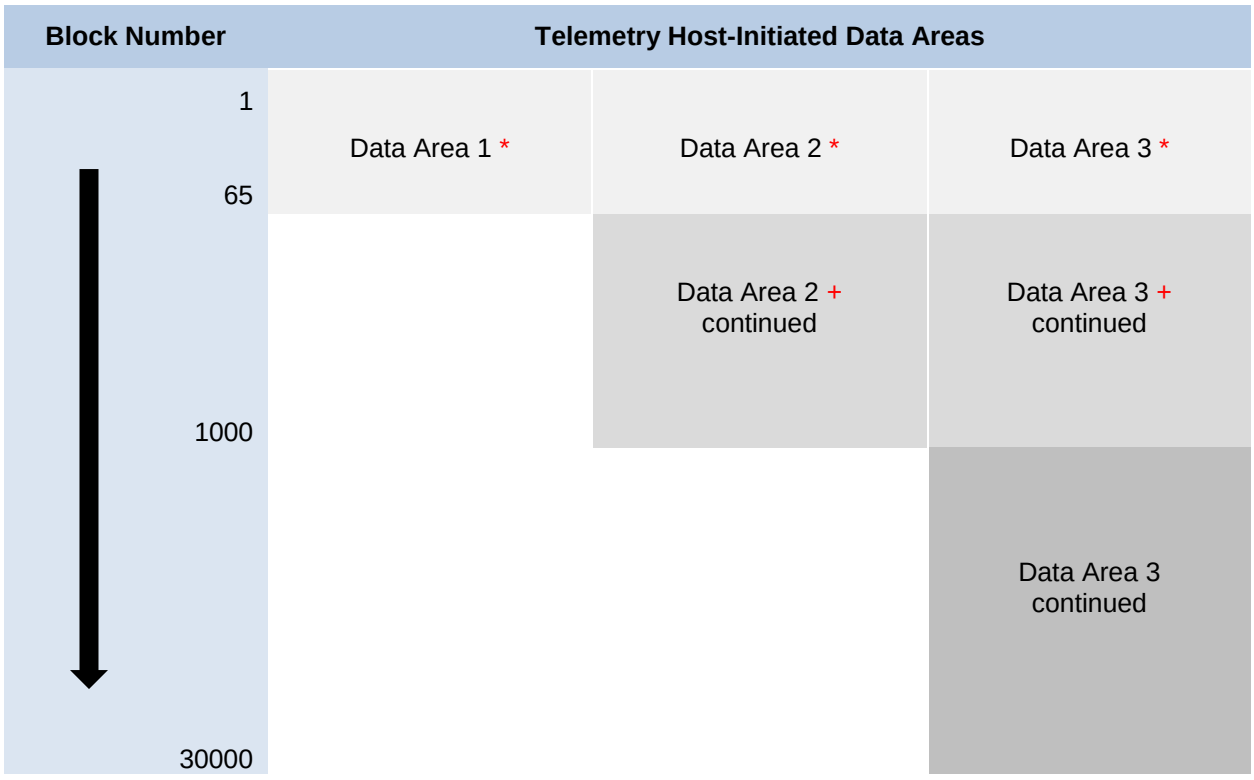
- Telemetry Host-Initiated Data Area 1 Last Block = 0,
- Telemetry Host-Initiated Data Area 2 Last Block = 0,
- Telemetry Host-Initiated Data Area 3 Last Block = 0.

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. ~~As an~~For example, the following values correspond to the layout shown in Figure 283:

- Telemetry Host-Initiated Data Area 1 Last Block = 65,
- Telemetry Host-Initiated Data Area 2 Last Block = 1000,
- Telemetry Host-Initiated Data Area 3 Last Block = 30000.

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 1000 of data area 2 and data area 3 is the same.

Figure 9: Telemetry Log Example – All Data Areas Populated



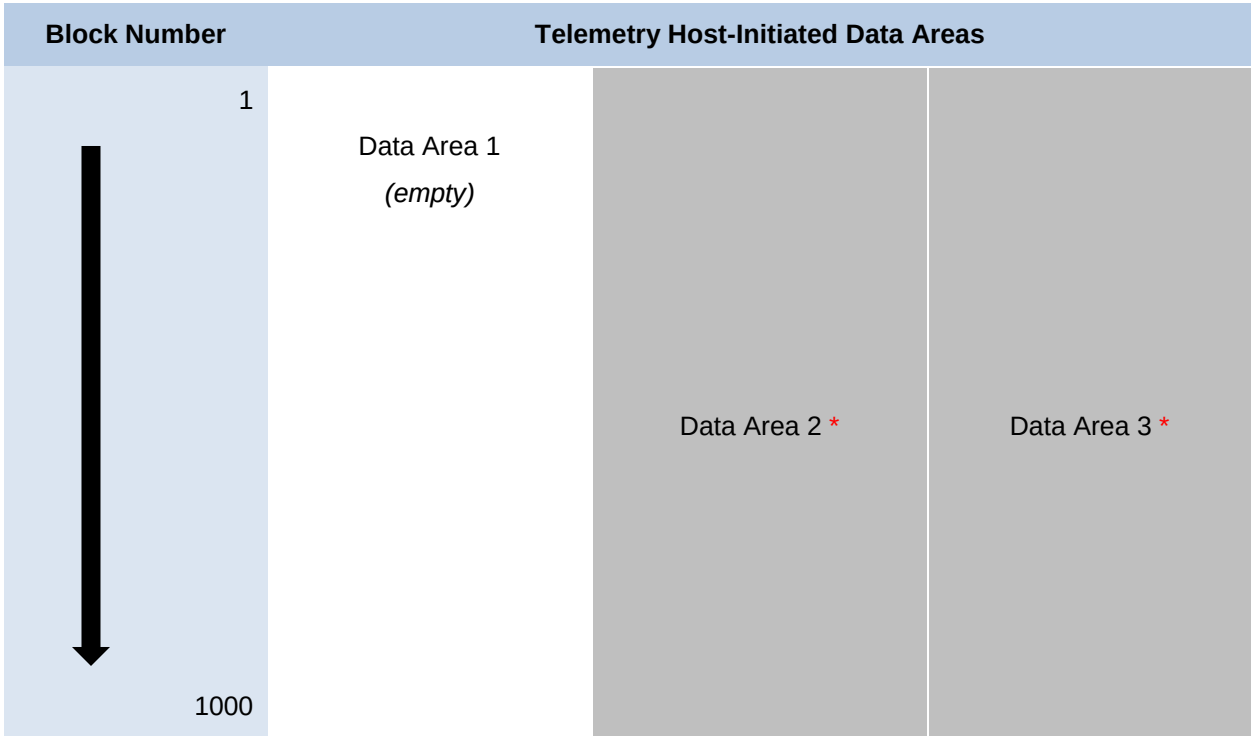
* Data Area 1, Data Area 2, and Data Area 3 contain the same telemetry data in blocks 1 through 65.
\+ Data Area 2 and Data Area 3 contain the same telemetry data in blocks 66 through 1000.

When only the second data areas 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. ~~As an~~For example, the following values correspond to the layout shown in Figure 284:

- Telemetry Host-Initiated Data Area 1 Last Block = 0,
- Telemetry Host-Initiated Data Area 2 Last Block = 1000,
- Telemetry Host-Initiated Data Area 3 Last Block = 1000.

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 1000 of data area of data area 2 and data area 3 is the same.

Figure 10: Telemetry Log Example – Data Area 2 Populated



* Data Area 2, and Data Area 3 contain the same telemetry data in blocks 1 through 1000.

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

9.3 Streams (Directive Type 01h, Optional)

…

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. ~~Stream resources may be allocated for the exclusive use of a specified namespace associated with a particular Host Identifier using the Allocate Resources operation. Stream resources that are not allocated for the exclusive use of any namespace are available NVM subsystem stream resources as reported in NVM Subsystem Streams Available (NSSA) and may be used by any namespace that has the Streams Directive enabled and has not been allocated exclusive stream resources in response to an Allocate Resources operation. As stream resources are allocated for the exclusive use of a specified namespace, the available NVM subsystem stream resources reported in the NSSA field are reduced.~~

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 and may be used by any host in any namespace that has the Streams Directive enabled and has not been allocated exclusive stream resources by that host. 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 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.

The Directive operations that shall be supported if the Streams Directive is supported are listed in Figure 294. 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.

...

The value of Namespace Streams Allocated (NSA) indicates how many resources for individual stream identifiers have been allocated for exclusive use ~~of~~ 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 ~~of~~ 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~~ 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~~ 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 ~~streams~~ stream resources allocated, if the host submits a ~~write~~ Write 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 ~~of~~ for specific namespaces, then the ~~write~~ Write command is treated as a normal ~~write~~ 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 a host in a namespace, then all stream resources and stream identifiers are released for the ~~affected~~ host in that namespace. If the host issues a Format NVM command, or deletes a namespace, then all stream identifiers for all open streams for affected namespaces are released. Streams Directive defines the command specific status values specified in Figure 295.

Modify a portion of section 9.3.1.1 (Return Parameters) as shown below:

9.3.1.1 Return Parameters (Directive Operation 01h)

…

Figure 11: Streams Directive – Return Parameters Data Structure

Bytes	Description
	NVM Subsystem Specific Fields
1:0	**Max Streams Limit (MSL):** 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.
3:2	**NVM Subsystem Streams Available (NSSA)**: 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 ~~of~~ any specific namespace. This field returns the same value independent of specified namespace.
5:4	**NVM Subsystem Streams Open (NSSO)**: 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~~ are not associated with a namespace with allocated stream resources~~. This field returns the same value independent of specified namespace.
15:6	Reserved
	Namespace Specific Fields
19:16	**Stream Write Size (SWS):** This field indicates the alignment and size of the optimal stream write as a number of logical blocks for ~~this~~ the specified namespace. 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 LBA format. If the NSID value is set to FFFFFFFFh then this field may be cleared to 0h if a single logical block size cannot be indicated.
21:20	**Stream Granularity Size (SGS):** This field indicates the stream granularity size for ~~this~~ 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.
	Namespace and Host Identifier Specific Fields
23:22	**Namespace Streams Allocated (NSA):** This field indicates the number of stream resources allocated for exclusive use ~~with this~~ of the specified namespace by the controller processing the Return Parameters operation and by all other controllers which share the same non-zero Host Identifier and which are attached to the specified namespace. 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 zero, then no stream resources are currently allocated to this namespace and the namespace may have up to NSSA number of concurrently open streams.
25:24	**Namespace Streams Open (NSO)**: This field indicates the number of open streams in the specified namespace that were opened by the controller processing the Return Parameters operation and by all other controllers which share the same non-zero Host Identifier and which are attached to this namespace.~~ This field is cleared to zero if no stream resources are currently allocated to this namespace.~~ 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.
31:26	Reserved

Modify a portion of section 9.3.1.2 (Get Status) as shown below:

9.3.1.2 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 an NSID value of FFFFFFFFh is specified, then the controller shall return information about the status of currently open streams ~~for~~ in the NVM subsystem that ~~are not associated with any namespace that has allocated stream resources for its exclusive~~ use resources which are not allocated for the exclusive use of any namespace.

Modify a portion of section 9.3.1.3 (Allocate Resources) as shown below:

9.3.1.3 Allocate Resources (Directive Operation 03h)

The Allocate Resources operation indicates the number of streams that the host requests for the exclusive use ~~of~~ by the host for the specified namespace. 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 ~~of this~~ by the host 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 set NSA to 0000h 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~~s~~ resources allocated for ~~its~~ the exclusive use of the host issuing the Allocate Resources operation, then the controller shall return a status value of Invalid Field in Command. To allocate additional streams resources, the host should release resources and then request a complete set of resources.