LEGAL NOTICE:

This erratum to the NVM Express revision 1.2 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 1.2 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	**001**
Revision Date	**8/30/2016**
Affected Spec Ver.	**NVM Express™ 1.2.1**
Corrected Spec Ver.	

Errata Author(s)

Name	Company
Judy Brock	Samsung
David Black	EMC
James Smart	Broadcom
Fred Knight	NetApp
Dave Landsman	SanDisk
Nadesan Narenthiran	WD

Errata Overview

The erratum includes clarifications to support NVMe over Fabrics 1.0.

The erratum clarifies several items including:

- The Number of Namespaces field in Identify Controller.
- The Format NVM Attributes field in Identify Controller.
- The fields that are command specific in the SQE and CQE.

The erratum clarifies that if BAR2 and BAR3 are configured as memory, then they are vendor specific.

Revision History

Revision Date	Change Description
7/14/2016	Initial draft
7/20/2016	Added clarification for CNS=11h and NSID=FFFFFFFFh for Identify. Added clarification for "Number of Namespaces" in Identify. Added clarification for Format NVM Attributes. Added BAR2 and BAR3 clarification for discussion.
7/27/2016	Added proposal for Number of Namespaces and Format NVM Attributes. Added context in comments to consider for BAR2 and BAR3. Added proposed clarification on wording for resets to discuss. Added LBA Format clarification for currently unavailable formats.
8/4/2016	Added numerous items that were sent to the reflector. Will update the overview as the team decides what to keep.
8/19/2016	Various updates to finalize ECN.
8/30/2016	Removed various items that the team decided to defer to ECN 002 or place in a host guidance document.

Description of Specification Changes

Modify section 1.6.6 as shown below:

1.6.6 command submission

For NVMe over PCIe, a ~~A~~ command is submitted when a Submission Queue Tail Doorbell write has completed that moves the Submission Queue Tail Pointer value past the corresponding Submission Queue entry for the associated command.

For NVMe over Fabrics, refer to section 1.4.14 in the NVMe over Fabrics 1.0 specification.

Modify the beginning of section 4 as shown below:

4 ~~Memory~~ Data Structures

This section describes ~~memory~~ data structures used by NVM Express. ~~The memory structures are supported in host memory. If the Controller Memory Buffer feature is supported, then memory structures may be supported in the controller's memory; refer to section 4.7.~~

Modify the beginning of section 4.1 as shown below:

Sections 4.1, 4.1.1 and 4.1.2 apply to NVMe over PCIe only. For NVMe over Fabrics, refer to sections 2.4, 2.4.1 and 2.4.2 in the NVMe over Fabrics 1.0 specification.

The submitter of entries to a 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 4.1.2).

…

Modify Figure 90 (Identify Controller data structure) as shown below:

767:540	M	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.
~~1792:1024~~ 1791:1024		Reserved
2047:1792		Refer to the NVMe over Fabrics specification.

Modify section 1.1.1 as shown below:

NVM Express 1.2.1 and prior revisions define a register level interface for host software to communicate with a non-volatile memory subsystem over PCI Express (NVMe over PCIe). The NVMe over Fabrics specification defines a protocol interface and related extensions to NVMe that enable operation over other interconnects (e.g., Ethernet, InfiniBand™, Fibre Channel). The NVMe over Fabrics specification has an NVMe Transport binding for each NVMe Transport (either within that specification or by reference).

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 90.

Modify a portion of Figure 11 (Command Format data structure) as shown below:

<table>
<tr><td>39:24</td><td>
Data Pointer (DPTR): This field specifies the data used in the command.

If CDW0[15:14] is set to 00b, then the definition of this field is:

<table>
<tr><td>39:32</td><td>PRP Entry 2 (PRP2): This field:
a) is reserved if the data transfer does not cross a memory page boundary.
b) specifies the Page Base Address of the second memory page if the data transfer crosses exactly one memory page boundary. E.g.,:
i. 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
ii. the Offset portion of the PBAO field of PRP1 is equal to zero and the command data transfer length is greater than one memory page and less than or equal to two memory pages in size.
c) is a PRP List pointer if the data transfer crosses more than one memory page boundary. E.g.,:
i. 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
ii. 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 zero.</td></tr>
<tr><td>31:24</td><td>PRP Entry 1 (PRP1): This field contains the first PRP entry for the command or a PRP List pointer depending on the command.</td></tr>
</table>
If CDW0[15:14] is set to 01b or 10b, then the definition of this field is:

<table>
<tr><td>39:24</td><td>SGL Entry 1 (SGL1): This field contains the first SGL segment for the command. If the SGL segment is an SGL Data Block or Keyed SGL ~~a (Keyed)~~ 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 4.4 for the definition of SGL segments and descriptor types.

The NVMe Transport may support a subset of SGL Descriptor types and features as defined in the NVMe Transport binding specification.</td></tr>
</table>
</td></tr>
</table>

Modify a portion of Figure 87 as shown below:

11h	The Identify Namespace data structure is returned to the host for the namespace specified in the Namespace Identifier (CDW1.NSID) field if it is an allocated NSID. If the specified namespace is an unallocated NSID then the controller returns a zero filled data structure. If the specified namespace is an invalid NSID then the controller shall fail the command with a status code of Invalid Namespace or Format. If CDW1.NSID is set to FFFFFFFFh then the controller should fail the command with a status code of Invalid Namespace or Format.

Modify a portion of Figure 90 as shown below:

519:516	M	**Number of Namespaces (NN):** This field defines the maximum number of ~~valid~~ namespaces ~~present for~~ supported by the controller. This field also represents the maximum value of a valid NSID for the controller.

Modify a portion of the table in section 2.1 as shown below:

Start	End	Symbol	Name

00h	03h	ID	Identifiers
04h	05h	CMD	Command Register
06h	07h	STS	Device Status
08h	08h	RID	Revision ID
09h	0Bh	CC	Class Codes
0Ch	0Ch	CLS	Cache Line Size
0Dh	0Dh	MLT	Master Latency Timer
0Eh	0Eh	HTYPE	Header Type
0Fh	0Fh	BIST	Built In Self Test (Optional)
10h	13h	MLBAR (BAR0)	Memory Register Base Address, lower 32-bits <BAR0>
14h	17h	MUBAR (BAR1)	Memory Register Base Address, upper 32-bits <BAR1>
18h	1Bh	~~IDBAR (BAR2)~~ BAR2	~~Index/Data Pair Register Base Address <BAR2> (Optional)~~ Refer to section 2.1.12.
1Ch	1Fh	BAR3	~~Reserved <BAR3>~~ Vendor Specific
20h	23h	BAR4	Vendor Specific
24h	27h	BAR5	Vendor Specific
28h	2Bh	CCPTR	CardBus CIS Pointer
2Ch	2Fh	SS	Subsystem Identifiers
30h	33h	EROM	Expansion ROM Base Address (Optional)
34h	34h	CAP	Capabilities Pointer
35h	3Bh	R	Reserved
3Ch	3Dh	INTR	Interrupt Information
3Eh	3Eh	MGNT	Minimum Grant (Optional)
3Fh	3Fh	MLAT	Maximum Latency (Optional)

Modify section 2.1.12 and 2.1.13 as shown below:

2.1.12 Offset 18h: ~~IDBAR (~~BAR2~~)~~ – Index/Data Pair Register Base Address or Vendor Specific (Optional)

If this register is configured as I/O space, then this ~~This~~ register specifies the Index/Data Pair base address and is configured as shown in the table below. These registers are used to access the memory registers defined in section **Error! Reference source not found.** using I/O based accesses. ~~If Index/Data Pair is not supported, then the IDBAR shall be read only 0h.~~

Bit	Type	Reset	Description
31:03	RW	0	**Base Address (BA):** Base address of Index/Data Pair registers that is 8 bytes in size.
02:01	RO	0	Reserved
00	RO	1	**Resource Type Indicator (RTE):** Indicates a request for register I/O space.

If this register is configured as memory space (Resource Type Indicator is cleared to '0'), then the BAR2 register is vendor specific. Vendor specific space may also be allocated at the end of the memory registers defined in section **Error! Reference source not found.**.

2.1.13 Offset 1Ch – 20h: BAR3 – ~~Reserved~~ Vendor Specific

~~The BAR3 register allocates memory or an I/O space. BAR3 is reserved for future use.~~

The BAR3 register is vendor specific. Vendor specific space may also be allocated at the end of the memory registers defined in section **Error! Reference source not found.**.

Modify a portion of Figure 90 as shown below:

<table>
<tr>
<td>524</td>
<td>M</td>
<td>
Format NVM Attributes (FNA): This field indicates attributes for the Format NVM command.

Bits 7:3 are reserved.

Bit 2 indicates whether cryptographic erase is supported as part of the secure erase functionality. If set to ‘1’, then cryptographic erase is supported. If cleared to ‘0’, then cryptographic erase is not supported.

~~Bit 1 indicates whether cryptographic erase and user data erase functionality apply to all namespaces or is specific to a particular namespace. If set to‘1’, then a cryptographic erase of a particular namespace as part of a format results in a cryptographic erase of all namespaces, and a user data erase of a particular namespace as part of a format results in a user data erase of all namespaces. If cleared to ‘0’, then a cryptographic erase or user data erase as part of a format is performed on a per namespace basis.~~

~~Bit 0 indicates whether the format operation applies to all namespaces or is specific to a particular namespace. If set to ‘1’, then all namespaces shall be configured with the same attributes and a format of any namespace results in a format of all namespaces. If cleared to ‘0’, then the controller supports format on a per namespace basis.~~

Bits 1:0 indicate whether the format operation may be specified per namespace or if it impacts the entire NVM subsystem. It is defined as shown in the table below.
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>00b</td><td>Format (including the cryptographic erase and user data erase options) may be performed on a per namespace basis or the entire NVM subsystem may be specified.</td></tr>
<tr><td>01b</td><td>This value should not be specified by the controller. If it is specified, the host should treat it as equivalent to value 11b.</td></tr>
<tr><td>10b</td><td>Format (excluding the cryptographic erase and user data erase options) may be performed on a per namespace basis or the entire NVM subsystem may be specified. The cryptographic erase and user data erase options affect the entire NVM subsystem.</td></tr>
<tr><td>11b</td><td>Format (including cryptographic erase and user data erase options) affect the entire NVM subsystem.</td></tr>
</table>
</td>
</tr>
</table>

Modify Figure 92 as shown below:

25	M	**Number of LBA Formats (NLBAF):** This field defines the number of supported LBA data size and metadata size combinations supported by the namespace. LBA formats shall be allocated in order (starting with 0) and packed sequentially. This is a 0's based value. The maximum number of LBA formats that may be indicated as supported is 16. The supported LBA formats are indicated in bytes 128 – 191 in this data structure.The LBA Format fields with an index beyond the value set in this field are invalid and not supported. LBA Formats that are valid, but not currently available may be indicated by setting the LBA Data Size for that LBA Format to 0h. The metadata may be either transferred as part of the LBA (creating an extended LBA which is a larger LBA size that is exposed to the application) or it may be transferred as a separate contiguous buffer of data. The metadata shall not be split between the LBA and a separate metadata buffer. It is recommended that software and controllers transition to an LBA size that is 4KB or larger for ECC efficiency at the controller. If providing metadata, it is recommended that at least 8 bytes are provided per logical block to enable use with end-to-end data protection, refer to section 8.2.

Modify Figure 93 as shown below:

23:16	**LBA Data Size (LBADS):** This field indicates the LBA data size supported. The value is reported in terms of a power of two (2^*n*). A value smaller than 9 (i.e. 512 bytes) is not supported. If the value reported is 0h then the LBA format is not supported / used or is currently not available.

Modify a portion of section 1.4.1 (Multi-Path I/O and Namespace Sharing) as shown below:

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 ~~mapping~~ of a controller's namespace IDs ~~to namespaces~~ is outside the scope of this specification.

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

~~The metadata for this command shall be all zeroes and t~~The protection information is updated based on CDW12.PRINFO. If the Protection Information Action field (PRACT) is cleared to '0', the metadata for this command shall be all zeroes. If the Protection Information Action field (PRACT) is set to '1', any non-PI related metadata, if it exists, shall be all zeroes.

Modify a portion of section 5.13.1 as shown below:

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 96. For failures, the byte offset ~~location~~ 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.

Modify a portion of section 3.1.5 as shown below:

<table>
<tr>
<td>15:14</td>
<td>RW</td>
<td>0h</td>
<td>Shutdown Notification (SHN): This field is used to initiate shutdown processing when a shutdown is occurring, (i.e., a power down condition is expected.) For a normal shutdown notification, it is expected that the controller is given time to process the shutdown notification. For an abrupt shutdown notification, the host may not wait for shutdown processing to complete before power is lost.

The shutdown notification values are defined as:

<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>00b</td><td>No notification; 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>
</table>

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 reboot. To determine when shutdown processing is complete, refer to CSTS.SHST. Refer to section Error! Reference source not found. for additional shutdown processing details.

Other fields in the CC register (including the EN bit) may be modified as part of updating this field to 01b or 10b.</td>
</tr>
</table>

Modify a portion of section 5 as shown below:

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

Modify a portion of section 6 as shown below:

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

Modify a portion of section 8.4 as shown below:

A lower relative performance value indicates better performance (e.g., higher throughput or lower latency). For example, in ~~Figure 219~~ Figure 220 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.