NVM Express™ Technical Errata

Errata ID	**013**
Revision Date	**8/21/2014**
Affected Spec Ver.	**NVM Express™ 1.0 and NVM Express 1.1b**
Corrected Spec Ver.	

Errata Author(s)

Name	Company
Neal Galbo	Micron
Ken Okin, Jason Gao	HGST
Edward Hsieh	SMI
Judy Brock	Samsung
Peter Onufryk, Kwok Kong	PMC-Sierra

Errata Overview

The definition of PRP Entry 2 is clarified throughout the specification.

The Get Log Page and Security Send & Receive commands were modified to allow a pointer to a PRP List to accommodate a larger data transfer.

The specification is made consistent that the MLBAR (BAR0) may be 32-bit address space to be consistent that only 32-bit addresses may be assigned behind a bridge.

Clarifications are included for when it is safe to power off the controller related to shutdown operations.

Write Uncorrectable is added as a command that may cause the “Attempted Write to Read Only Range” error.

Various editorial updates are also made.

Revision History

Revision Date	Change Description
5/15/2014	First draft. Captured PRP2 changes and allowing Get Log Page and Security Send & Receive to point to a PRP List for PRP2.
6/4/2014	Added Write Uncorrectable as a command that may cause the error "Attempted Write to Read Only Range".
7/22/2014	Added atomic clarifications, async event clarifications, and KB clarification.
8/12/2014	Added clarification on write zeroes
8/15/2014	Edited AER clarifications, added autonomous transitions, SGL, editorial changes, minimum queue size, and identify clarifications
8/19/2014	Added clarifications and updates based on review of pages 140 to 172 of the specification, clarified Controller Level Reset, various other updates.
8/21/2014	Rejected change to Figure 46.

Description of Specification Changes

Modify the PRP Entry 2 field of Figure 11 as shown below:

Figure 11: Command Format – Admin Command Set

Bytes	Description
43:40	**Command Dword 10 (CDW10):** This field is command specific Dword 10.
39:32	~~**PRP Entry 2 (PRP2):** This field contains the second PRP entry for the command or if the data transfer spans more than two memory pages, then this field is a PRP List pointer.~~ **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.
31:24	**PRP Entry 1 (PRP1):** This field contains the first PRP entry for the command or a PRP List pointer depending on the command.

Modify the PRP Entry 2 field of Figure 12 as shown below:

Figure 12: Command Format – NVM Command Set

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

If CDW0[15] is cleared to ‘0’, then the definition of this field is:

<table>
<tr><td>39:32</td><td><s>PRP Entry 2 (PRP2):</s> <s>This field contains the second PRP entry for the command or if the data transfer spans more than two memory pages, then this field is a PRP List pointer.</s>

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.,:
iii. 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
iv. 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] is set to ‘1’, 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 a 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 Error! Reference source not found. for the definition of SGL segments and descriptor types.</td></tr>
</table>
</td></tr>
</table>

Modify Figure 62 as shown below:

Figure 62: Firmware Image Download – PRP Entry 2

Bit	Description
63:00	**PRP Entry 2 (PRP2):** This field contains the second PRP entry. Refer to Figure 11 for the definition of this field. <s>If the data transfer is satisfied with PRP Entry 1, then this field is reserved. If the data transfer may be satisfied with two PRP entries total, then this entry specifies the location where data should be transferred from. If the data transfer requires more than two PRP entries, then this field contains a pointer to a PRP List.</s>

Modify Figure 67 as shown below:

Figure 67: Get Features – PRP Entry 2

Bit	Description
63:00	**PRP Entry 2 (PRP2):** This field contains the second PRP entry. Refer to Figure 11 for the definition of this field. ~~If PRP Entry 1 specifies enough space for the data structure, then this field is reserved. Otherwise, it specifies the remainder of the data buffer. 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 ignored.~~

Modify Figure 71 as shown below:

Figure 71: Get Log Page – PRP Entry 2

Bit	Description
63:00	**PRP Entry 2 (PRP2):** This field contains the second PRP entry. Refer to Figure 11 for the definition of this field. ~~If PRP Entry 1 specifies enough space for the data structure, then this field is reserved. Otherwise, it specifies the remainder of the data buffer. This field shall not be a pointer to a PRP List as the data buffer may not cross more than one page boundary.This field contains the second PRP entry.~~

Modify Figure 81 as shown below:

Figure 81: Identify – PRP Entry 2

Bit	Description
63:00	**PRP Entry 2 (PRP2):** This field contains the second PRP entry. Refer to Figure 11 for the definition of this field. ~~If PRP Entry 1 specifies enough space for the data structure, then this field is reserved. Otherwise, it specifies the remainder of the data buffer.~~ This field shall not be a pointer to a PRP List as the data buffer may not cross more than one page boundary.

Modify Figure 88 as shown below:

Figure 881: Set Features – PRP Entry 2

Bit	Description
63:00	**PRP Entry 2 (PRP2):** This field contains the second PRP entry. Refer to Figure 11 for the definition of this field. ~~If PRP Entry 1 specifies enough space for the data structure, then this field is reserved. Otherwise, it specifies the remainder of the data buffer.~~ 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.

Modify Figure 115 as shown below:

Figure 115: Security Receive – PRP Entry 2

Bit	Description
63:00	**PRP Entry 2 (PRP2):** This field contains the second PRP entry. Refer to Figure 11 for the definition of this field. ~~If PRP Entry 1 specifies enough space for the data structure, then this field is reserved. Otherwise, it specifies the remainder of the data buffer. This field shall not be a pointer to a PRP List as the data buffer may not cross more than one page boundary.~~

Modify Figure 119 as shown below:

Figure 119: Security Send – PRP Entry 2

Bit	Description
63:00	**PRP Entry 2 (PRP2):** This field contains the second PRP entry. Refer to Figure 11 for the definition of this field. ~~If PRP Entry 1 specifies enough space for the data structure, then this field is reserved. Otherwise, it specifies the remainder of the data buffer. This field shall not be a pointer to a PRP List as the data buffer may not cross more than one page boundary.~~

Modify the last paragraph of section 4.3 as shown below:

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 ~~page is a pointer to the~~ contains the Page Base Address of the next PRP List page. The total number of PRP entries required by a command is implied by the command parameters and memory page size.

Modify Figure 15 as shown below:

Figure 15: PRP Entry – Page Base Address and Offset

Bit	Description
63:02	**Page Base Address and Offset (PBAO):** This field indicates the 64-bit physical memory page address. The lower bits (*n*:2) of this field indicate the offset within the memory page. If the memory page size is 4KB, then bits 11:02 form the Offset; if the memory page size is 8KB, then bits 12:02 form the Offset, etc. If this entry is not the first PRP entry in the command or ~~in the PRP List~~ a PRP list pointer in a command then the Offset portion of this field shall be cleared to 0h.
01:00	Reserved

Modify section 2.1.10 as shown below:

2.1.10 Offset 10h: MLBAR (BAR0) – Memory Register Base Address, lower 32-bits

This register allocates space for the memory registers defined in section 3.

Bit	Type	Reset	Description
31:14	RW	0	**Base Address (BA):** Base address of register memory space. For controllers that support a larger number of doorbell registers or have vendor specific space following the doorbell registers, more bits are allowed to be RO such that more memory space is consumed.
13:04	RO	0	Reserved
03	RO	0	**Prefetchable (PF):** Indicates that this range is not pre-fetchable
02:01	RO	~~10~~ Impl Spec	**Type (TP):** ~~Indicates that this range may be mapped anywhere in 64-bit address space and that the register is 64-bits wide.~~ Indicates where this range may be mapped. It is recommended to support mapping anywhere in 64-bit address space.

00	RO	0	**Resource Type Indicator (RTE):** Indicates a request for register memory space.

Modify the first paragraph of section 6.14 as shown below:

The Write Uncorrectable command is used to mark a ~~logical block~~ range of logical blocks as invalid. When the specified logical block(s) are read after this operation, a failure is returned with Unrecovered Read Error status. To clear the invalid logical block status, a write operation is performed on those logical blocks.

Modify Figure 43 as shown below:

Figure 43: Asynchronous Event Request – Completion Queue Entry Dword 0

<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>31:24</td><td>Reserved</td></tr>
<tr><td>23:16</td><td><s>Associated</s> Log Page Identifier: Indicates the log page associated with the asynchronous event. This log page needs to be read by the host to clear the event.</td></tr>
<tr><td>15:08</td><td>Asynchronous Event Information: Refer to Figure 44, <s>and</s> Figure 45, and Figure 46 for detailed information regarding the asynchronous event.</td></tr>
<tr><td>07:03</td><td>Reserved</td></tr>
<tr><td>02:00</td><td>Asynchronous Event Type: Indicates the type of the asynchronous event. More specific information on the event is provided in the Asynchronous Event Information field.
<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>0h</td><td>Error status</td></tr>
<tr><td>1h</td><td>SMART / Health status</td></tr>
<tr><td>2h – 5h</td><td>Reserved</td></tr>
<tr><td>6h</td><td>I/O Command Set specific status</td></tr>
<tr><td>7h</td><td>Vendor specific</td></tr>
</table></td></tr>
</table>

Modify the last three paragraphs of section 7.6.2 on Shutdown as shown below:

It is recommended that the host wait a minimum of one second for the shutdown operations to complete. It is not recommended to disable the controller via the CC.EN field. This causes a controller reset condition which may impact the time required to complete shutdown processing.

It is safe to power off the controller when CSTS.SHST indicates shutdown processing is complete (regardless of the value of CC.EN). It remains safe to power off the controller until CC.EN transitions from '0' to '1'.

To start executing commands on the controller after a shutdown operation, a reset (CC.EN cleared from '1' to '0') is required. The initialization sequence should then be executed.

It is an implementation choice whether the host aborts all outstanding commands to the Admin Queue prior to the shutdown. The only commands that should be outstanding to the Admin Queue at shutdown are Asynchronous Event Request commands.

Modify Figure 33 as shown below:

Figure 33: Status Code – Command Specific Status Values, NVM Command Set

Value	Description	Commands Affected
80h	Conflicting Attributes	Dataset Management, Read, Write
81h	Invalid Protection Information	Compare, Read, Write, Write Zeroes
82h	Attempted Write to Read Only Range	Dataset Management, Write, Write Uncorrectable, Write Zeroes
83h - BFh	Reserved	

Modify section 6.7.2 as shown below:

6.7.2 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.

Dataset Management command specific status values are defined in Figure 140.

Figure 140: Dataset Management – Command Specific Status Values

Value	Description
80h	**Conflicting Attributes:** The attributes specified in the command are conflicting.
82h	**Attempted Write to Read Only Range:** The controller may optionally report this status if a Deallocate is attempted for a read only range.

Modify a portion of Figure 83 as shown below:

533:532	O	**Atomic Compare & Write Unit (ACWU):** This field indicates the atomic write size for the controller for a Compare and Write fused command. This field shall be supported if the Compare and Write fused command is supported. This field is specified in logical blocks and is a 0's based value. If a Compare and Write is submitted that requests a transfer size larger than this value, then the controller ~~shall~~ may fail the command with a status code of Invalid Field in Command. If Compare and Write is not a supported fused command, then this field shall be 0h.

Modify a portion of section 6.4.2.1 as shown below:

6.4.2.1 AWUPF Example (Informative)

In this example, AWUPF has a value of 1K (equivalent to two 512 byte logical blocks), AWUN has a value of ~~4K~~ 2K (equivalent to four 512 byte logical blocks). Command A writes LBAs 0-1. Figure 125 shows the initial state of the NVM.

Modify a portion of section 2.1.3 as shown below:

Offset 06h: STS - Device Status

Bit	Type	Reset	Description
15	RWC	0	**Detected Parity Error (DPE):** Set to '1' by hardware when the controller detects a parity error on its interface.
14	RWC/RO	0	**Signaled System Error (SSE):** ~~Not supported by NVM Express~~. Refer to the PCI SIG specifications.

Modify the last two paragraphs of section 5.2 as shown below:

Asynchronous events are reported due to a new entry being added to a log page (e.g., Error Information log) or a status update (e.g., status in the SMART / Health log). A status change may be permanent (e.g., the media has become read only) or transient (e.g., the temperature exceeded a threshold for a period of time). Host software should modify the event threshold or mask the event for transient and permanent status changes before issuing another Asynchronous Event Request command to avoid repeated reporting of asynchronous events.

If the controller needs to report an event and there are no outstanding Asynchronous Event Request commands, the controller should send a single notification of that Asynchronous Event Type when an Asynchronous Event Request command is received. If a Get Log Page command clears the event prior to receiving the Asynchronous Event Request command, then a notification is not sent.

~~Asynchronous events may be reported due to a single instance (e.g., Invalid Doorbell Write Value) or a persistent condition (e.g., Temperature Above Threshold). If the asynchronous event is triggered due to a persistent condition, the host should modify the event threshold or mask the event before issuing another Asynchronous Event Request command. If the host clears the event without taking these recommended actions for a persistent condition, then the persistent condition may cause repeated reporting of asynchronous events.~~

~~When the controller needs to report an event and there are no outstanding Asynchronous Event Request commands, the controller queues the event internal to the controller and reports it when an Asynchronous Event Request command is received.~~

Modify section 1.5 as shown below:

1.5 Conventions

Hardware shall return '0' for all bits and registers that are marked as reserved, and host software shall write all reserved bits and registers with the value of '0'.

Inside the register section, the following abbreviations are used:

RO	Read Only
RW	Read Write
R/W	Read Write. The value read may not be the last value written.
RWC	Read/Write '1' to clear
RWS	Read/Write '1' to set
Impl Spec	Implementation Specific – the controller has the freedom to choose its implementation.
HwInit	The default state is dependent on NVM Express controller and system configuration. The value is initialized at reset, for example by an expansion ROM, or in the case of integrated devices, by a platform BIOS.

For some register 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 register bit is referred to in the document, the convention used is “Register Symbol.Field Symbol”. For example, the PCI command register parity error response enable bit is referred to by the name CMD.PEE. If the register field is an array of bits, the field is referred to as “Register Symbol.Field Symbol (array offset)”.

When a memory field is referred to in the document, the convention used is “Register Name [Offset Symbol]”.

~~Some fields or registers are 0’s based values. In a 0’s based value, the value of 0h corresponds to 1; other values similarly correspond to the value+1.~~

A 0-based value is a numbering scheme for which the number 0h actually corresponds to a value of 1h and thus produces the pattern of 0h = 1h, 1h = 2h, 2h = 3h, etc. In this numbering scheme, there is not a method for specifying the value of 0h.

When a size is stated in the document as KB, the convention used is 1KB = 1024 bytes.

Some parameters are defined as a string of ASCII characters. ASCII data fields shall contain only code values 20h through 7Eh. For the string “Copyright”, the character “C” is the first byte, the character “o” is the second byte, etc. The string is left justified and shall be padded with spaces (ASCII character 20h) to the right if necessary.

Delete section 1.8 as shown below:

~~**1.8 Conventions**~~

~~A 0-based value is a numbering scheme for which the number 0h actually corresponds to a value of 1h and thus produces the pattern of 0h = 1h, 1h = 2h, 2h = 3h, etc. In this numbering scheme, there is not a method for specifying the value of 0h.~~

~~Some parameters are defined as a string of ASCII characters. ASCII data fields shall contain only code values 20h through 7Eh. For the string “Copyright”, the character “C” is the first byte, the character “o” is the second byte, etc. The string is left justified and shall be padded with spaces (ASCII character 20h) to the right if necessary.~~

***Modify Section* 7.7 from NVM Express ECN 009 as shown below:**

7.7 Asynchronous Event Request Host Software Recommendations (Informative)

This section describes the recommended host software procedure for Asynchronous Event Requests.

The host sends *n* Asynchronous Event Request commands (refer to section 7.6.1, step 11). When an Asynchronous Event Request completes (providing Asynchronous Event Type, Event Information, and Log Page details):

1. If the event(s) in the reported Log Page may be disabled with the Asynchronous Event Configuration feature (refer to section 5.12.1.11), then h~~H~~ost software issues a Set Features command for the Asynchronous Event Configuration feature disabling the reporting of all events that utilize the Log Page reported. Host software should wait for the Set Features command to complete.
2. 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.

3. 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.
4. Host software should issue an Asynchronous Event Request command to the controller (restoring to *n* the number of these commands outstanding).
5. If the reporting of event(s) was disabled, host software should enable reporting of the event(s) using Asynchronous Event Configuration feature. If the condition reported may persist ~~was persistent~~, host software should continue to monitor the condition (e.g., ~~over temperature~~ spare below threshold)~~,~~ via Get Log Page, to determine if reporting of the event should be re-enabled.

Modify section 5.9.2 as shown below:

A completion queue entry is posted to the Admin Completion Queue if the controller has completed returning any attributes associated with the Feature. Depending on the Feature Identifier, Dword 0 of the completion queue entry may contain feature information (refer to section ~~0~~ 5.12.1).

Modify a portion of Figure 85 as shown below:

30	O	**Namespace Multi-path I/O and Namespace Sharing Capabilities (NMIC):** This field specifies multi-path I/O and namespace sharing capabilities of the namespace. Bits 7:1 are reserved Bit 0: If set to '1' then the ~~NVM~~ namespace may be accessible by two or more controllers in the NVM subsystem (i.e., may be a shared namespace). If cleared to '0' then the ~~NVM~~ namespace is a private namespace and may only be accessed by the controller that returned this namespace data structure.

***Modify Figure 93* as shown below:**

Figure 2: Power Management – Command Dword 11

Bit	Description
31:05	Reserved
04:00	**Power State (PS):** This field indicates the new power state into which the controller should transition. This power state shall be one supported by the controller as indicated in the Number of Power States Supported (NPSS) field in the ~~Indentify~~ Identify Controller data structure. The behavior of transitioning to a power state not supported by the controller is undefined.

Modify section 6.16 as shown below:

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 zeroes until a write occurs to this LBA range. The metadata for this command shall be all zeroes and the protection information is updated based on CDW12.PRINFO.

***Modify Figure 174* as shown below:**

Figure 174: Write Zeroes – Command Dword 12

Bit	Description
31	**Limited Retry (LR):** If set to '1', the controller should apply limited retry efforts. If cleared to '0', the controller should apply all available error recovery means to write the data to the NVM.
30	**Force Unit Access (FUA):** This field indicates that the data shall be written to non-volatile media before indicating command completion. There is no implied ordering with other commands.
29:26	**Protection Information Field (PRINFO):** Specifies the protection information action and check field, as defined in Figure 127. The Protection Information Check field (PRCHK) shall be 000b.
25:16	Reserved
15:00	**Number of Logical Blocks (NLB):** This field indicates the number of logical blocks to be written. This is a 0's based value.

***Modify Figure 30* as shown below:**

Figure 3: Status Code – Generic Command Status Values

Value	Description
00h	**Successful Completion:** The command completed successfully.
01h	**Invalid Command Opcode:** The associated command opcode field is not valid.
02h	**Invalid Field in Command:** An invalid or unsupported field specified in the command parameters.

***Modify a portion of section 5.12.1.12* as shown below:**

This feature configures the settings for autonomous power state transitions, refer to section 8.4.2.

The Autonomous Power State Transition uses Command Dword 11 and specifies the ~~type and~~ attribute information in the data structure indicated in Figure 105. ~~The data structure is 256 bytes in size and shall be physically contiguous.~~

If a Get Features command is issued for this Feature, the attributes specified in Figure 105 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 106 is returned in the data buffer for that command.

Figure 105: Autonomous Power State Transition – Command Dword 11

Bit	Description
31:01	Reserved
00	**Autonomous Power State Transition Enable (APSTE):** This field specifies whether autonomous power state transition is enabled. If this field is set to '1', then autonomous power state transitions are enabled. If this field is cleared to '0', then autonomous power state transitions are disabled. This field is cleared to '0' by default.

Each entry in the Autonomous Power State Transition data structure is defined in Figure 106. 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.

***Modify a portion of section 8.2* as shown below:**

The second mechanism for transferring the metadata is as a separate ~~contiguous~~ buffer of data. This mechanism is illustrated in **Error! Reference source not found.**. In this case, the metadata is pointed to with the Metadata Pointer ~~(or Metadata SGL Segment Pointer if SGLs are used)~~, while the logical block data is pointed to by the Data Pointer ~~PRP1 and PRP2 pointers (or SGL Entry 1 if SGLs are used)~~. When a command uses PRPs for the metadata in the command, ~~T~~the metadata is required to be physically contiguous ~~in this case since there is only one Metadata Pointer~~. When a command uses SGLs for the metadata in the command, the metadata is not required to be physically contiguous.

***Modify a portion of section 4.5* as shown below:**

Metadata may be supported for a namespace as either part of the logical block (creating an extended logical block which is a larger logical block 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 logical block and a separate metadata buffer. For writes, the metadata shall be written atomically with its associated logical block. Refer to section **Error! Reference source not found.**.

In the case where the namespace is formatted to transfer the metadata as a separate ~~contiguous~~ buffer of data, then the Metadata Region is used. In this case, the location of the Metadata Region is indicated by the Metadata Pointer within the command. The Metadata ~~(SGL Segment)~~ Pointer within the command shall be Dword aligned.

***Modify a portion of Figure 68* as shown below:**

Figure 4: Get Features – Command Dword 10

<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>31:11</td><td>Reserved</td></tr>
<tr><td>10:08</td><td>Select (SEL): This field specifies which value of the attributes to return in the provided data:

<table>
<tr><th>Select</th><th>Description</th></tr>
<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 – 111b</td><td>Reserved</td></tr>
</table>

Refer to section ~~Error! Reference source not found.~~ 5.9.1 for details on the value returned in each case.

The controller indicates in bit 4 of the Optional NVM Command Support field of the Identify Controller ~~D~~data structure in Figure 83 whether this field is supported.

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 treat the Select field as though it was set to 001b (i.e., default.)</td></tr>
</table>

***Modify a portion of Figure 89* as shown below:**

Figure 5: Set Features – Command Dword 10

Bit	Description
31	**Save (SV):** This field specifies that the controller shall save the attribute so that the attribute persists ~~thru~~ through all power states and resets. The controller indicates in bit 4 of the Optional NVM Command Support field of the Identify Controller ~~D~~data structure in Figure 83 whether this field is supported. If the Feature Identifer specified in the Set Features command is not saveable by the controller and the controller recieves a Set Features command with the Save bit set to one, then the command shall be aborted with a status of Feature Identifier Not Saveable.

***Modify a portion of Figure 90* as shown below:**

This column is only valid if bit 4 in the Optional NVM Command Support field of the Identify Controller ~~D~~data structure in Figure 83 is cleared to '0'.

Modify a portion of Figure 91 as shown below:

This column is only valid if bit 4 in the Optional NVM Command Support field of the Identify Controller ~~D~~data structure in Figure 83 is cleared to '0'.

Modify a portion of Figure 83 as shown below:

Figure 83: Identify – Identify Controller Data Structure

Bytes	O/M	Description
259	M	**Asynchronous Event Request Limit (AERL):** This field is used to convey the maximum number of concurrently outstanding Asynchronous Event Request commands supported by the controller (see section **Error! Reference source not found.**). This is a 0's based value. It is recommended that implementations support a minimum of four Asynchronous Event Request Limit commands outstanding simultaneously.

Modify a portion of Figure 109 as shown below:

Figure 109: Reservation Notification Configuration – Command Dword 11

Bit	Description
31:04	Reserved
03	**Mask Reservation Preempted Notification (RESPRE):** If set to '1', then mask the reporting of reservation preempted notification by the controller. If cleared to '0', then the notification is not masked and a Reservation Notification log page is created whenever notification occurs.
02	**Mask Reservation Released Notification (RESREL):** If set to '1', then mask the reporting of reservation released notification by the controller. If cleared to '0', then the notification is not masked and a Reservation Notification log page is created whenever the notification occurs.
01	**Mask Registration Preempted Notification (REGPRE):** If set to '1', then mask the reporting of registration preempted notification by the controller. If cleared to '0', then the notification is not masked and a Reservation Notification log page is created whenever the ~~notificaiton~~ notification occurs.
00	Reserved

***Modify a portion of section 3.1.8* as shown below:**

3.1.8 Offset 24h: AQA – Admin Queue Attributes

Bit	Type	Reset	Description
31:28	RO	0h	Reserved
27:16	RW	0h	**Admin Completion Queue Size (ACQS):** Defines the size of the Admin Completion Queue in entries. Refer to section 4.1.3. Enabling a controller while this field is cleared to 00h produces undefined results. The minimum size of the Admin Completion Queue is two entries. The maximum size of the Admin Completion Queue is 4096 entries. This is a 0's based value.
15:12	RO	0h	Reserved
11:00	RW	0h	**Admin Submission Queue Size (ASQS):** Defines the size of the Admin Submission Queue in entries. Refer to section 4.1.3. Enabling a controller while this field is cleared to 00h produces undefined results. The minimum size of the Admin Submission Queue is two entries. The maximum size of the Admin Submission Queue is 4096 entries. This is a 0's based value.

Modify a portion of Figure 32 as shown below:

Figure 32: Status Code – Command Specific Status Values

Value	Description	Commands Affected
00h	Completion Queue Invalid	Create I/O Submission Queue
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	~~Maximum~~ Invalid Queue Size ~~Exceeded~~	Create I/O Submission Queue, Create I/O Completion Queue

Modify a portion of Figure 51 as shown below:

Figure 51: Create I/O Completion Queue – Command Specific Status Values

Value	Description
1h	**Invalid Queue Identifier:** 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 is currently in use or one that is outside the range supported by the controller.
2h	**~~Maximum~~ Invalid Queue Size ~~Exceeded~~:** The host attempted to create an I/O Completion Queue with an invalid number of entries (e.g., ~~that~~ a value of zero or a value which exceeds the maximum supported by the controller, specified in CAP.MQES).

Modify a portion of Figure 55 as shown below:

Figure 55: Create I/O Submission Queue – Command Specific Status Values

Value	Description
0h	**Completion Queue Invalid:** The Completion Queue identifier specified in the command does not exist.
1h	**Invalid Queue Identifier:** 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 is currently in use or one that is outside the range supported by the controller.
2h	**~~Maximum~~ Invalid Queue Size ~~Exceeded~~:** Host software attempted to create an I/O Submission Queue with an invalid number of entries (e.g., ~~that~~ a value of zero or a value which exceeds the maximum supported by the controller, specified in CAP.MQES)

Modify the paragraph following Figure 180 as shown below:

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.CDW1.NSID is set to the appropriate namespace. This value shall be the same as CMD0.CDW1.NSID.
- If metadata is being used in a separate buffer, then the location of that buffer is specified.
    - If a command uses PRPs then CMD1.MPTR is set to the address of the metadata buffer.
    - If a command uses SGLs then CMD1.MSGLP is set to an SGL segment that describes the metadata buffer.
- 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.

    - If the command uses SGLs, ~~CMD0.SGL1~~ 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 compare against. Note that this field also spans Command Dword 11. This value shall be the same as CMD0.CDW10.SLBA.
- CMD1.CDW12.LR is set 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 DRAM 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 4KB 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.
- CMD1.CDW15 is cleared to 0h since end-to-end protection is not enabled.

Modify the sixth paragraph of section 7.8 as shown below:

If the controller supports the Save field in the Set Features command and the Select field in the Get Features command, then any Feature Identifier may be namespace specific as a value may be saved per namespace. ~~If bit 4 is set to '1' in the Optional NVM Command Support field of the Identify Controller Data structure in Figure 83, then any Feature Identifier may be namespace specific depending on the implementation. Host software may discover if a Feature Identifier is namespace specific by using the 'supported capabilities' value in the Select field in Get Features. If bit 4 is cleared to '0' in the Optional NVM Command Support field of the Identify Controller Data structure in Figure 83, then LBA Range Type is the only Feature Identifier that is namespace specific.~~

Modify the second paragraph of section 8.3 as shown below (adding a parenthesis after "Figure 181"):

The most commonly used data protection mechanisms in Enterprise implementations are Data Integrity Field (DIF), defined in the SCSI Block Commands – 3 reference (SBC-3), and the Data Integrity Extension (DIX). The primary difference between these two mechanisms is the location of the protection information. In DIF the protection information is contiguous with the logical block data and creates an extended logical block, while in DIX the protection information is stored in a separate buffer. The end-to-end data protection mechanism defined by this specification is functionally compatible with both DIF and DIX. DIF functionality is achieved by configuring the metadata to be contiguous with logical block data (as shown in Figure 181), while DIX functionality is achieved by configuring the metadata and data to be in separate buffers (as shown in Figure 182).

Modify section 7.3.3 as shown below:

7.3.3 Queue Level

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 reset, 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 if desired.

The host should ensure that the appropriate I/O Submission Queue or I/O Completion Queue is idle before deleting it. 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). 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 reset as part of the same operation. The behavior of an I/O Submission Queue without a corresponding I/O Completion Queue is undefined.

Modify section 7.4.3 as shown below:

7.4.3 Queue Abort

To abort a large number of commands, the recommended procedure is to delete and recreate the I/O Submission Queue. Specifically, to abort all commands that are submitted to the I/O Submission Queue host software should issue a Delete I/O Submission Queue command for that queue. After the 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 any commands desired to the associated I/O Submission Queue.

Modify the first paragraph of section 1.4 as shown below:

NVM Express is a scalable host controller interface designed to address the needs of Enterprise and Client systems that utilize PCI Express based solid state drives. 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 64K 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.

Modify paragraphs two and three of section 8.3 as shown below:

The most commonly used data protection mechanisms in Enterprise implementations are SCSI Protection Information, commonly known as Data Integrity Field (DIF), ~~defined in the SCSI Block Commands – 3 reference (SBC-3),~~ and the Data Integrity Extension (DIX). The primary difference between these two mechanisms is the location of the protection information. In SCSI Protection Information ~~DIF~~ the protection information is contiguous with the logical block data and creates an extended logical block, while in DIX the protection information is stored in a separate buffer. The end-to-end data protection mechanism defined by this specification is functionally compatible with both SCSI Protection Information ~~DIF~~ and DIX. SCSI Protection Information ~~DIF~~ functionality is achieved by configuring the metadata to be contiguous with logical block data (as shown in Figure 181, while DIX functionality is achieved by configuring the metadata and data to be in separate buffers (as shown in Figure 182).

NVM Express supports the same end-to-end protection types as SCSI Protection Information ~~DIF~~. The type of end-to-end data protection (Type 1, Type 2, or Type 3) is selected when a namespace is formatted and is reported in the Identify Namespace data structure.

Modify the second paragraph following Figure 185 as shown below:

Checking of protection information consists of the following operations performed by the controller. If bit 2 of the Protection Information Check (PRCHK) field of the command is set to ‘1’, then the controller compares the protection information Guard field to the CRC-16 computed over the logical block data. If bit 1 of the PRCHK field is set to ‘1’, then the controller compares unmasked bits in the protection information Application Tag field to the Logical Block Application Tag (LBAT) field in the command. A bit in the protection information Application Tag field is masked if the corresponding bit is cleared to ‘0’ in the Logical Block Application Tag Mask (LBATM) field of the command. If bit 0 of the PRCHK field is set to ‘1’, then the controller compares the

protection information Reference Tag field to the computed reference tag. The value of the computed reference tag for the first LBA of the command is the value contained in the Initial Logical Block Reference Tag (ILBRT) or Expected Initial Logical Block Reference Tag (EILBRT) field in the command, for writes and reads respectively. The computed reference tag is incremented for each subsequent logical block. Unlike SCSI Protection Information ~~DIF~~ Type 1 protection which implicitly uses the least significant four bytes of the LBA, The controller always uses the ILBRT or EILBRT field and requires host software to initialize the ILBRT or EILBRT field to the least significant four bytes of the LBA when Type 1 protection is used.

Modify the third step in section 7.2.1 as shown below:

3. The controller fetches the command(s) in the Submission Queue from host memory for future execution. Arbitration is the method used to determine the Submission Queue from which the controller starts processing the next candidate command, refer to section 4.9.

Modify the first paragraph of section 7.2.5.2 as shown below:

This example describes how host software creates and executes a fused command, specifically Compare and Write for a total of 16KB 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 ~~LBA~~ logical block is 4KB.

Modify the Shutdown Status field in section 3.1.6 as shown below:

<table>
<tr>
<td>03:02</td>
<td>RO</td>
<td>0</td>
<td>Shutdown Status (SHST): This field indicates the status of shutdown processing that is initiated by the host setting the CC.SHN field.

The shutdown status values are defined as:

<table>
<tr><th>Value</th><th>Definition</th></tr>
<tr><td>00b</td><td>Normal operation (no shutdown has been requested)</td></tr>
<tr><td>01b</td><td>Shutdown processing occurring</td></tr>
<tr><td>10b</td><td>Shutdown processing complete</td></tr>
<tr><td>11b</td><td>Reserved</td></tr>
</table>

To start executing commands on the controller after a shutdown operation (CSTS.SHST set to 10b), a Controller Reset ~~reset~~ (CC.EN cleared to ‘0’) is required. If host software submits commands to the controller without issuing a reset, the behavior is undefined.</td>
</tr>
</table>

Modify the first paragraph of section 5.7 as shown below:

The Firmware Activate command is used to verify that a valid firmware image has been downloaded and to commit that revision to a specific firmware slot. The host may select the firmware image to activate on the next Controller Level Reset ~~controller reset (CC.EN transitions from ‘1’ to ‘0’, a PCI function level reset, and/or other Controller or NVM Subsystem Reset)~~ as part of this command. The currently executing firmware revision may be determined from the Firmware Revision field of the Identify Controller data structure in Figure 83 or as indicated in the Firmware Slot Information log page.

Modify the first paragraph of section 5.7.1 as shown below:

A completion queue entry is posted to the Admin Completion Queue if the controller has completed the requested action (specified in the Activate Action field). For requests that specify activation of a new firmware image and return with status code value of 00h, any ~~controller level reset~~ Controller Level Reset defined in

section ~~7.3.1~~ 7.3.2 activates the specified firmware. Firmware Activate command specific status values are defined in Figure 61.

Modify the title and first paragraph of section 7.3.2 as shown below:

7.3.2 Controller Level Reset

There are five primary ~~controller level reset~~ Controller Level Reset mechanisms:

Modify the fifth and sixth paragraphs of section 7.6.2 as shown below:

It is recommended that the host wait a minimum of one second for the shutdown operations to complete. It is not recommended to disable the controller via the CC.EN field. This causes a Controller Reset ~~controller reset condition~~ which may impact the time required to complete shutdown processing.

To start executing commands on the controller after a shutdown operation, a Controller Reset ~~reset~~ (CC.EN cleared from '1' to '0') is required. The initialization sequence should then be executed.

Modify the second paragraph of section 9.1 as shown below:

In the case of serious error conditions for Admin commands, the entire controller should be reset using a ~~Controller Reset~~ 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.